Carlos Augusto Bossolani

Representações Distribuídas de Texto Aplicadas em Análise de Sentimento de Mensagens Curtas e Ruidosas

Sorocaba, SP

14 de Dezembro de 2018

Carlos Augusto Bossolani

Representações Distribuídas de Texto Aplicadas em Análise de Sentimento de Mensagens Curtas e Ruidosas

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação (PPGCC-So) da Universidade Federal de São Carlos como parte dos requisitos exigidos para a obtenção do título de Mestre em Ciência da Computação. Linha de pesquisa: Computação Científica e Inteligência Computacional.

Universidade Federal de São Carlos – UFSCar

Centro de Ciências em Gestão e Tecnologia – CCGT

Programa de Pós-Graduação em Ciência da Computação – PPGCC-So

Orientador: Prof. Dr. Tiago Agostinho de Almeida

Sorocaba, SP

14 de Dezembro de 2018

Bossolani, Carlos Augusto

Representações Distribuídas de Texto Aplicadas em Análise de Sentimento de Mensagens Curtas e Ruidosas / Carlos Augusto Bossolani. -- 2018.
98 f. : 30 cm.

Dissertação (mestrado)-Universidade Federal de São Carlos, campus Sorocaba, Sorocaba
Orientador: Tiago Agostinho de Almeida
Banca examinadora: Akebo Yamakami, Evandro Eduardo Seron Ruiz, Sahudy Montenegro Gonzalez, Tiemi Christine Sakata
Bibliografia

1. Análise de sentimento. 2. Processamento de linguagem natural. 3. Aprendizado de máquina. I. Orientador. II. Universidade Federal de São Carlos. III. Título.

Ficha catalográfica elaborada pelo Programa de Geração Automática da Secretaria Geral de Informática (SIn).
DADOS FORNECIDOS PELO(A) AUTOR(A)
Bibliotecário(a) Responsável: Maria Aparecida de Lourdes Mariano – CRB/8 6979

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências em Gestão e Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Folha de Aprovação

Assinaturas dos membros da comissão examinadora que avaliou e aprovou a Defesa de Dissertação de Mestrado do candidato Carlos Augusto Bossolani, realizada em 14/12/2018:

Prof. Dr. Tiago Agostinho de Almeida
UFSCar

Prof. Dr. Evandro Eduardo Seron Ruiz
USP

Profa. Dra. Sahudy Montenegro González
UFSCar

Certifico que a defesa realizou-se com a participação à distância do(s) membro(s) Evandro Eduardo Seron Ruiz e, depois das arguições e deliberações realizadas, o(s) participante(s) à distância está(ao) de acordo com o conteúdo do parecer da banca examinadora redigido neste relatório de defesa.

Prof. Dr. Tiago Agostinho de Almeida

Aos meus pais.

Agradecimentos

Agradeço,

aos meus pais pelo incentivo e apoio,

ao meu orientador por me proibir de desistir.

"If you tell me precisely what it is a machine cannot do,
then I can always make a machine which will do just that."
(John von Neumann)

Resumo

A evolução da Internet e da Web proporcionou o surgimento de uma quantidade vasta de mensagens de texto contendo opiniões. Embora a importância da análise de sentimento tenha crescido proporcionalmente, o uso da tradicional *bag of words* como forma de representar computacionalmente essas mensagens impõe sérias limitações: a quantidade de dimensões das amostras pode ser muito alta; a informação sobre a posição relativa das palavras no texto é perdida; não é capturada a relação de sinonímia, e não é feita distinção dos diferentes sentidos de palavras ambíguas. Mensagens curtas, como as postadas nas redes sociais e aplicativos de mensagens instantâneas, costumam ser repletas de gírias, abreviaturas, ortografia fonética e *emoticons*, o que agrava o problema da representação computacional. Técnicas de normalização léxica e indexação semântica, tradicionalmente utilizadas para lidar com esses problemas, dependem de dicionários, a manutenção dos quais é inviável dada a velocidade de evolução da língua. Representações distribuídas de texto, que representam cada palavra por um vetor de baixa dimensionalidade, têm o potencial de contornar algumas dessas deficiências, por capturar as relações de similaridades entre as palavras, armazenando informações sobre os contextos da sua ocorrência. Técnicas recentes possibilitaram obter esses vetores a partir dos pesos de uma rede neural artificial, que são otimizados para maximizar a probabilidade dos contextos em que a palavra é observada. Otimizações posteriores possibilitaram gerar esses modelos com corpus bem maiores, fazendo ressurgir o interesse nessas técnicas. Este trabalho de pesquisa investigou e confirmou a hipótese de que o uso de modelos de representação distribuída de texto contornam os problemas e desvantagens do uso de *bag of words* em análise de sentimento em mensagens curtas e ruidosas, dispensando a necessidade de técnicas tradicionais de normalização léxica e indexação semântica, mantendo a qualidade preditiva e reduzindo o esforço computacional.

Palavras-chaves: Análise de sentimento. Processamento de linguagem natural. Aprendizado de máquina.

Abstract

The evolution of the Internet and the Web has given rise to a vast amount of text messages containing opinions. Although the importance of sentiment analysis has grown proportionately, the use of the traditional bag of words as a way to represent these messages computationally imposes serious limitations: the number of dimensions in the samples may be very high; information about the relative position of the words in the text is lost; the relation of synonymy is not captured, and no distinction is made between the different meanings of ambiguous words. Short messages, such as those posted on social media and instant messaging applications, often contain a lot of slang, abbreviations, phonetic spelling and emoticons, which aggravates the problem of computational representation. Lexical normalization techniques and semantic indexing, traditionally used to deal with these problems, depend on dictionaries and their maintenance is impractical given the speed of language evolution. Distributed text representations, which represent each word by a low dimensional vector, have the potential to bypass some of these shortcomings by capturing the similarity relationship among words, storing information about the contexts of their occurrence. Recent techniques have made it possible to obtain these vectors from the weights of an artificial neural network, which are optimized to maximize the probability of the contexts in which the word is observed. Later optimizations made it possible to generate these models with a much larger corpus, thus raising interest in these techniques. This work investigated and proved the hypothesis that the use of distributed text models overcomes the problems and disadvantages of the use bag of words in sentiment analysis in short and noisy messages, making it possible to dispense with the need for traditional lexical normalization techniques and semantic indexing, maintaining predictive power and reducing computational effort.

Key-words: Sentiment analysis. Natural language processing. Machine learning.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

NB-B	Naive Bayes Bernoulli
B.C4.5	Boosted C4.5
C4.5	Árvores de decisão
NB-G	Naive Bayes Gaussiano
k-NN	k-Vizinhos mais próximos
RL	Regressão logística
SVM-L	Máquinas de vetores de suporte com *kernel* linear
SVM-R	Máquinas de vetores de suporte com *kernel* de função de base radial
IMDb	*Internet Movie Database* (Banco de dados de Filmes)
SMS	*Short Message Service* (Serviço de Mensagens Curtas)
NLP	*Natural Language Processing* (Processamento de Linguagem Natural)
SVD	*Singular Value Decomposition* (Decomposição em valores singulares)
CBOW	*Continuous Bag of Words*
PV-DM	*Paragraph Vector - Distributed Memory*
PV-DBOW	*Paragraph Vector - Distributed Bag of Words*
GloVe	*Global Vectors*
NLTK	*Natural Language Toolkit*

Lista de símbolos

$x^{(i)}$	i-ésima amostra do conjunto de treinamento
$P(A\|B)$	Probabilidade condicional de A dado B
$M_{m\times n}$	Matriz com m linhas e n colunas
M^T	Transposta da Matriz M
$M_{a:b,c:d}$	Sub-matriz extraída do recorte da matriz M, considerando as linhas de a até b, e as colunas de c até d
$M_{m,:}$	Linha m da matriz M (todas as colunas)
$M_{:,n}$	Coluna n da matriz M (todas as linhas)
$O(x)$	Limite superior assintótico de x
$\exp(x)$	Função exponencial e^x

Sumário

Introdução

O número de mensagens postadas no Twitter diariamente é de aproximadamente 500 milhões[1]. Somam-se a esse oceano de mensagens os comentários sobre os vídeos no YouTube, as avaliações sobre produtos e serviços em sites como Amazon e Tripadvisor, as opiniões sobre filmes e livros em sites como Rotten Tomatoes[2] e Goodreads[3], e muitas outras. Dessas mensagens, cuja quantidade é gigantesca e crescente, as que manifestam opinião são especialmente valiosas se puderem ser classificadas quanto à polaridade expressada, ou seja, quanto a opinião ser positiva, negativa ou neutra. A área que estuda esse tipo de classificação é chamada de *análise de sentimento*, *mineração de opinião* ou *detecção de polaridade*.

Existem muitas aplicações importantes e lucrativas para análise de sentimento. Por exemplo, sistemas de recomendação que levam em consideração as opiniões sobre os produtos vendidos, podem fazer recomendações muito mais assertivas. Empresas que monitoram as opiniões sobre os produtos que fabricam, podem planejar mudanças para aumentar a satisfação dos consumidores. Candidatos a cargos políticos podem avaliar sua popularidade e traçar estratégias de campanha mais efetivas.

Uma das abordagens mais utilizadas em análise de sentimento é o emprego de métodos de classificação de texto (PANG; LEE, 2008). Em problemas de classificação, métodos de aprendizado de máquina são treinados a partir de um conjunto de amostras rotuladas que são usadas para gerar um modelo capaz de classificar amostras ainda não vistas. Em problemas de classificação de texto, uma amostra corresponde a uma mensagem de texto.

Para representar computacionalmente cada amostra, tradicionalmente usa-se o modelo espaço-vetorial, também chamado de *bag of words*. Nessa representação, a dimensionalidade de cada amostra corresponde ao tamanho de um vocabulário pré-definido. Embora seja simples, ela traz vários problemas: a quantidade de dimensões pode ser muito alta, pois corresponde ao tamanho do vocabulário; a informação sobre a posição relativa das palavras[4] no texto é perdida; relações de sinonímia não são capturadas e, portanto, não é possível distinguir os diferentes sentidos de palavras ambíguas; gírias, abreviações e erros de escrita podem não fazer parte do vocabulário.

---

1 *Twitter Usage Statistics*. Disponível em <http://www.internetlivestats.com/twitter-statistics>. Acessado em 28/09/2018

2 Rotten Tomatoes. Disponível em <http://www.rottentomatoes.com>. Acessado em 28/09/2018

3 Good Reads. Disponível em <https://www.goodreads.com>. Acessado em 28/09/2018

4 No decorrer deste manuscrito, "palavra" é utilizada indistintamente para representar qualquer termo ou *token* presente em uma mensagem de texto, podendo ser um símbolo, um conjunto de símbolos, entre outros

Esses problemas se agravam no caso de mensagens curtas, como as enviadas através de SMS (do inglês, *Short Message Service*), de aplicativos de mensagens instantâneas, e de *microblogs* como o Twitter, que geralmente são enviadas a partir de *smartphones*. A necessidade de escrever rapidamente, usando poucas palavras, e utilizando um teclado diminuto, faz com que os usuários utilizem uma espécie de "nova linguagem", denominada Lingo[5], para se comunicar com maior agilidade. Frequentemente, são usadas gírias, abreviaturas, símbolos, ortografia fonética (palavras com erro de escrita, mas com pronúncia similar a palavra correta) e *emoticons* para expressar sentimentos e emoções.

Tradicionalmente, para lidar com o problema das gírias, abreviações e erros de escrita, são empregadas técnicas de normalização que utilizam dicionários léxicos para traduzir variantes de palavras e expressões para sua forma canônica. Já o problema dos sinônimos e da polissemia é abordado empregando-se técnicas que substituem cada palavra pela representação da classe semântica específica da palavra no contexto, denominadas técnicas de indexação semântica. Ambas as técnicas utilizam dicionários semânticos e algoritmos de desambiguação (*word sense disambiguation*) para determinar o sentido correto da palavra de acordo com o contexto da mensagem.

Com base nessas técnicas, Lochter (2015) propôs um método sofisticado de análise de sentimento que envolve o uso de um comitê de classificadores com abordagens tradicionais de processamento de texto. O autor demonstrou que o uso desse comitê proporcionou um desempenho superior em problemas de detecção de polaridade em mensagens curtas e ruidosas, quando comparado com cada método de classificação individualmente. Seu método foi disponibilizado em uma ferramenta *online*, chamada `SentMiner`[6].

Embora o uso do comitê proporcionou um alto desempenho preditivo, foi constatado que o emprego de técnicas de indexação semântica promoveu uma melhoria em cerca de apenas 36% dos casos analisados (LOCHTER, 2015, p. 73). Além disso, essas técnicas são computacionalmente muito custosas, pois cada palavra da mensagem, junto com seu contexto, devem ser analisados. O maior problema, entretanto, reside na forte dependência de dicionários. As línguas evoluem com o tempo, e cada vez mais frequentemente surgem novas palavras ou novos significados são atribuídos a palavras existentes. Dessa forma, a atualização permanente dos dicionários utilizados representa um custo proibitivo, porém necessário para a manutenção da qualidade preditiva do sistema.

A utilização de técnicas recentes de *representações distribuídas de texto* tem o potencial de contornar algumas dessas deficiências, por capturar as relações de similaridade entre as palavras. Nesse tipo de representação, cada palavra é representada por um vetor de baixa dimensionalidade, que armazena informações sobre os contextos em que a palavra

[5] Linguagem escrita e falada, especialmente jargões e gírias de um determinado campo, grupo ou indivíduo

[6] SentMiner. Disponível em <http://lasid.sor.ufscar.br/sentminer/>. Acessado em 28/09/2018

ocorre. Vetores “próximos” entre si correspondem a palavras similares, por aparecerem em contextos similares.

Bengio et al. (2003) propuseram uma abordagem que obtém vetores de palavras de dimensionalidade reduzida a partir dos pesos de uma rede neural, que são otimizados para maximizar a probabilidade dos contextos em que a palavra é observada no corpus. O problema dessa abordagem é que para capturar as muitas similaridades semânticas é preciso utilizar corpora significativamente grandes. Isso faz com que a rede neural tenha muitos neurônios, tornando proibitivamente longo o tempo de treinamento com corpora grandes o suficiente. Mais recentemente, Mikolov et al. (2013b) propuseram diversas otimizações que possibilitaram gerar esses modelos com corpora bem maiores do que os usados anteriormente, fazendo ressurgir o interesse nesses modelos, e inspirando o surgimento de outros (PENNINGTON; SOCHER; MANNING, 2014; LE; MIKOLOV, 2014).

Hipótese

Os problemas e desvantagens característicos do uso da tradicional representação *bag of words* são potencializados quando as amostras são curtas e ruidosas, cenário no qual são comumente empregadas técnicas de normalização léxica e indexação semântica, que são fortemente dependentes de dicionários e computacionalmente onerosas.

Este trabalho de pesquisa avalia a hipótese de que o uso de modelos de representação distribuída de texto pode contornar esses problemas e desvantagens, dispensando a necessidade de técnicas tradicionais de normalização léxica e indexação semântica, eliminando a dependência da manutenção de dicionários, preservando a qualidade preditiva e reduzindo o esforço computacional na etapa de treinamento dos classificadores.

Objetivos e contribuições

Os principais objetivos e contribuições deste trabalho são:

- detalhar e analisar o funcionamento dos diferentes modelos de representação distribuída de texto;
- analisar como as características do corpo de texto usado no treinamento dos modelos de representação distribuída pode impactar na capacidade preditiva dos classificadores;
- avaliar e comparar os resultados obtidos com os diferentes modelos em problemas de análise de sentimento de mensagens curtas e ruidosas;

Organização

Este manuscrito está organizado da seguinte maneira:

- O **Capítulo 1** apresenta um breve histórico da área de análise de sentimentos de maneira geral, e especificamente em mensagens curtas e ruidosas, com intuito de contextualizar o problema abordado neste trabalho.
- No **Capítulo 2**, são discutidas as principais deficiências do uso da representação *bag of words*, e como essas deficiências são agravadas quando as mensagens são curtas e ruidosas. São apresentadas as abordagens tradicionais para lidar com essas deficiências, e é apresentado o *SentMiner* proposto por Lochter (2015).
- No **Capítulo 3**, são apresentados os principais modelos de representação distribuída de texto. São descritos os detalhes de funcionamento e implementação e evidenciadas as vantagens e desvantagens de cada modelo.
- O **Capítulo 4** descreve os experimentos realizados para avaliar como as características do corpo de texto usado para treinar os modelos distribuídos podem impactar na sua qualidade.
- O **Capítulo 5** descreve os experimentos e resultados obtidos usando representação distribuída na tarefa de análise de sentimento em textos curtos e ruidosos. É apresentada uma análise estatística dos resultados realizada para determinar se a hipótese da pesquisa foi confirmada.
- O **Capítulo 6** oferece uma discussão sobre o que pode ser concluído a partir dos resultados e da análise estatística.

1 Análise de sentimento

As previsões estavam erradas quando em novembro de 2016, Donald Trump contrariando a maioria das expectativas, venceu as eleições presidenciais dos EUA. Esse fato chamou a atenção para a importância e a dificuldade de descobrir a opinião das pessoas. Pesquisas de intenção de voto existem desde 1824, quando *The Harrisburg Pennsylvanian* realizou uma pesquisa que mostrava Andrew Jackson liderando John Quincy Adams por 335 votos a 169 nas eleições para a presidência dos EUA[1]. Essas pesquisas, hoje em dia, são usadas durante todo o período das campanhas eleitorais pelos candidatos e pela mídia, e seus resultados determinam em grande parte onde os fundos de campanha serão gastos e onde os esforços de cada candidato serão concentrados. A pesquisa de mercado, de maneira geral, é um componente essencial do plano de marketing de qualquer empresa. Ela determina a estratégia, informa a alocação de recursos e ajuda as empresas a entenderem e se conectarem com seus clientes de maneiras que, no passado, pareceriam impossíveis.

Saber a opinião das pessoas, entretanto, não é importante apenas para as empresas e candidatos a cargos políticos. O interesse na opinião de outros provavelmente é quase tão antigo quanto a própria comunicação verbal, sendo uma importante peça de informação para a maioria das pessoas durante o processo de tomada de decisão.

Nos últimos anos, a Internet e a Web têm tornado possível descobrir as opiniões e as experiências de um vasto grupo de pessoas que não são conhecidos pessoais, nem críticos profissionais. Não só pessoas estão buscando opiniões na Internet, mas elas também estão disponibilizando suas opiniões para estranhos. Pang e Lee (2008) chamam a atenção para a crescente disponibilidade e popularidade de recursos ricos em opiniões, como sites de críticas *online* e blogs pessoais. Em 2008, os autores apresentaram dados de duas pesquisas que já evidenciavam a importância da Internet com respeito ao crescimento do interesse na análise de sentimento:

- 81% dos usuários da Internet (ou 60% dos americanos) fizeram pesquisas *online* sobre um produto pelo menos uma vez;
- 20% (15% de todos os americanos) fazem isso em um dia típico;
- dos leitores de avaliações *online* de restaurantes, hotéis e vários serviços (por exemplo, agências de viagens ou médicos), entre 73% e 87% relatam que as avaliações tiveram uma influência significativa em sua compra;
- consumidores relataram estar dispostos a pagar de 20% a 99% a mais em um item classificado com 5 estrelas do que um item classificado com 4

[1] *Opinion poll*. Disponível em <https://en.wikipedia.org/wiki/Opinion_poll>. Acessado em 31/5/2018

estrelas (a variância decorre de que tipo de item ou serviço é considerado);

- 32% forneceram uma classificação para um produto, serviço ou pessoa por meio de um sistema de classificação *online*, e 30% (incluindo 18% dos idosos *online*) postaram um comentário *online* ou avaliação referente a um produto ou serviço (HORRIGAN; LIPSMAN, 2008, 2007 apud PANG; LEE, 2008, tradução nossa).

O aumento dos recursos ricos em opiniões tem um reflexo direto no aumento dos trabalhos científicos sobre análise de sentimento (LIU, 2012). Mäntylä, Graziotin e Kuutila (2016) realizaram uma revisão da literatura assistida por computador, com intuito de responder algumas perguntas sobre a pesquisa na área. Eles procuraram artigos usando o Scopus e o Google Scholar e encontraram aproximadamente 7000 artigos, publicados desde 1940 até 2018, sendo que 99% deles surgiram depois de 2004. O resultado do seu trabalho conta a história da evolução da pesquisa na área.

Análise de sentimento consiste na série de métodos, técnicas e ferramentas utilizados para detectar e extrair informações subjetivas, como opinião e atitudes de pessoas, a partir de textos escritos em linguagem natural (LIU, 2010). Tradicionalmente, análise de sentimento está relacionada com detecção de polaridade, ou seja, se alguém tem uma opinião negativa, positiva ou neutra em relação a algo (DAVE et al., 2003). Geralmente, a opinião expressada é sobre um produto ou serviço cuja crítica foi escrita e tornada pública na Internet, o que explica o fato de "detecção de polaridade", "mineração de opinião" e "análise de sentimento" serem frequentemente usados como sinônimos.

Antes da disponibilidade da grande quantidade de opiniões *online*, os estudos se restringiam a analisar resultados de pesquisas de opiniões públicas ou de especialistas, como em Stagner (1940). O trabalho realizado pela *Association for Computational Linguistics*, fundada em 1962, influenciou de maneira importante o nascimento da análise de sentimentos moderna. Nessa comunidade, Wiebe (1990) apresentou métodos para detectar as sentenças subjetivas de uma narrativa, e nove anos depois, ele e outros pesquisadores propuseram um "*gold-standard*" para fazer isso (WIEBE; BRUCE; O'HARA, 1999).

Em meados da década de 1990, computadores passaram a ser usados para realizar análise de sentimento. Kalfsbeek e Dubois (1995), por exemplo, usaram o computador para analisar opiniões de especialistas no domínio de segurança industrial. Em 1997, Hatzivassiloglou e McKeown (1997), usando corpus do Wall Street Journal de 1987, construíram uma lista de adjetivos positivos e negativos e previram se os adjetivos conjuntos são da mesma orientação semântica (por exemplo, "justo" e "legítimo") ou diferentes (por exemplo, "justo" e "brutal").

A partir de meados da década de 2000, a popularização da Internet proporcionou um aumento do número de trabalhos em análise de sentimento (LIU, 2012), que inicial-

mente se concentrou nas críticas de produtos disponíveis na Web.

Em 2002, Pang, Lee e Vaithyanathan (2002) deram início à análise de sentimentos moderna. Em seu artigo, examinaram a eficácia de aplicar técnicas de aprendizado de máquina ao problema de classificação de sentimentos. O objetivo do trabalho, segundo os autores, foi examinar se é possível tratar a classificação de sentimentos como um caso específico de classificação de texto em tópicos (sendo dois: os sentimentos positivo e negativo), ou se precisariam ser desenvolvidas novas técnicas específicas para categorização de sentimentos. Eles coletaram críticas de filmes do *Internet Movie Database* (IMDb) e utilizaram os métodos naive Bayes, máquina de vetores de suporte (do inglês *support-vector machines* ou SVM) e entropia máxima, para classificar essas críticas quanto sendo positivas ou negativas, obtendo uma acurácia máxima de 82,9%. Eles concluíram que, embora os resultados tenham sido bons, estão muito abaixo dos relatados em trabalhos que usam as mesmas técnicas para classificar texto em tópicos, o que indica que classificar sentimento seria um problema particularmente mais desafiador.

No mesmo ano, Turney (2002) descreveu um método não-supervisionado para classificar a polaridade de avaliações *online* de automóveis, bancos, filmes e destinos de viagem. Ele usou a mesma ideia de orientação semântica descrita em Hatzivassiloglou e McKeown (1997), mas desenvolveu um algoritmo mais simples. Esse algoritmo extraía cada frase das avaliações e executava uma busca no motor de busca *Alta Vista*, usando o operador *NEAR*, para encontrar o número de documentos que continha as palavras "*excellent*" e "*poor*" até uma distância de 10 palavras de cada palavra da frase, em qualquer ordem. Com essa estratégia, o autor alcançou uma precisão média de 74% na classificação das recomendações.

Em 2003, Turney e Littman (2003) expandiram a idéia apresentada em Turney (2002), e propuseram um método para inferir automaticamente a orientação semântica de uma palavra a partir do contexto estatístico. A estratégia foi determinar a orientação semântica de uma determinada palavra a partir da força de sua associação com um conjunto de palavras positivas, menos a força de sua associação com um conjunto de palavras negativas. Ao contrário de seu trabalho anterior (TURNEY, 2002), os conjuntos continham sete palavras positivas e sete negativas, em vez de apenas uma. Esta técnica obteve 78% de precisão classificando palavras na lista de palavras contidas em Hatzivassiloglou e McKeown (1997).

O trabalho de Dave et al. (2003) abordou a detecção de polaridade de críticas de produtos do C|net[2]. O corpus estudado consiste em 10 conjuntos aleatoriamente selecionados de 56 críticas positivas e 56 negativas das 4 maiores categorias do C|net, totalizando 448 críticas. A abordagem extraiu n-gramas (unigramas, bigramas e trigramas) que foram

[2] *Website* que contém avaliações de diversos produtos. Disponível em <https://www.cnet.com/>, acessado em 18/10/2018

avaliados como vetores de frequência. Em seguida, foi utilizada um máquina de vetores de suporte, que conseguiu classificar as críticas com precisão de 85,8%, usando validação cruzada de dez partições sem estratificação.

Os trabalhos citados avaliaram sentimento de documentos inteiros, partindo do pressuposto que as opiniões expressadas em cada documento são sobre a mesma entidade. Nos anos que se seguiram, observa-se tanto trabalhos que abordaram diferentes níveis de granularidade, aplicações e aplicabilidade da análise de sentimento. Quanto aos níveis de análise, alguns trabalhos posteriores abordam a detecção polaridade de sentenças (WIEBE et al., 2004), chegando até o nível mais fino de captar sentimento de diferentes aspectos de múltiplas entidades. Por exemplo, a sentença "A qualidade das chamadas do iPhone é boa, mas a duração da bateria é curta" avalia dois aspectos, a qualidade da chamada e a duração da bateria, de uma entidade (iPhone) (LIU, 2012).

Áreas distintas como a previsão dos mercados financeiros (NASSIRTOUSSI et al., 2014) e reações a ataques terroristas (BURNAP et al., 2014) foram abordadas, e tarefas auxiliares como detecção de ironia (REYES; ROSSO, 2014) e suporte multilíngue (HOGENBOOM et al., 2014) também foram estudadas. No que diz respeito às emoções, os esforços avançaram partindo da detecção de polaridade simples e chegando a detecção de nuances mais complexas das emoções, diferenciando emoções negativas como a raiva e o luto (CAMBRIA et al., 2015).

1.1 Mensagens curtas e ruidosas

A popularização da Internet ocasionou o surgimento de novas formas de comunicação mediada por computador, incluindo salas de bate-papo, quadros de avisos e sites de redes sociais. Aliado a isso, o aumento do uso de *smartphones* fez com que uma grande quantidade de mensagens de texto passasse a ser enviada, primeiramente usando SMS, depois usando outros aplicativos. Finalmente, os *microblogs* surgiram, sendo o Twitter o mais popular. O que essas mensagens têm em comum é um estilo repleto de erros de escrita, abreviações, gírias, *emoticons* e frases truncadas (GRINTER; ELDRIDGE, 2003; THURLOW, 2003).

Do ponto de vista da análise de sentimento, as redes sociais e plataformas de *microblogs* forneceram novas fontes valiosas de opiniões. O Twitter foi criado em 2006, e em 2013 o número de mensagens enviadas através da plataforma por dia já era de aproximadamente 500 milhões[3]. Muitas dessas mensagens contém opiniões sobre algo. As Figuras 1 e 2 ilustram críticas negativa e positiva respectivamente, sobre dois filmes diferentes.

[3] *Twitter Usage Statistics*. Disponível em <http://www.internetlivestats.com/twitter-statistics/>. Acessado em 11/08/2018

Figura 1 – Exemplo de opinião negativa.

Fonte: Twitter.

Figura 2 – Exemplo de opinião positiva.

Fonte: Twitter.

Alguns trabalhos abordaram o problema de análise de sentimento em mensagens curtas e ruidosas, sendo que a maioria se concentrou em mensagens do Twitter.

Thelwall et al. (2010) desenvolveram um algoritmo chamado de *SentiStrength*, para extrair a "força" do sentimento a partir de mensagens ruidosas em inglês, usando novos métodos para explorar os estilos de escrita característicos desse tipo de mensagem. Medindo a força do sentimento em uma escala de 1 a 5, e utilizando os comentários do MySpace, o algoritmo foi capaz prever a força das emoções positivas com 60,6% de precisão e das emoções negativas com 72,8% de precisão.

Sitaram e Huberman (2010) usaram mensagens do Twitter para prever os lucros da bilheteria de filmes. Os autores construíram um modelo de regressão linear para prever as receitas de bilheteria dos filmes na pré-estreia. Os resultados dos experimentos superaram em precisão os indicadores da Bolsa de Valores de Hollywood, demonstrando que há uma forte correlação entre a quantidade de atenção que um determinado filme recebe antes do lançamento, e o seu lucro após o lançamento.

Tumasjan et al. (2010) examinaram cerca de 100.000 *tweets* com intuito de avaliar se eles são bons indicadores dos resultados das eleições. Os autores descobriram que mensagens *online* no Twitter refletem o sentimento político *offline*, porém com menos precisão do que as pesquisas de opinião.

O 'Connor et al. (2010) utilizaram dados do Twitter para avaliar a opinião pública em séries temporais. Eles analisaram várias pesquisas sobre a confiança do consumidor e a opinião política durante o período de 2008 a 2009 e compararam com as opiniões expressas em mensagens do Twitter, no mesmo período. Um fato interessante é que eles estavam interessados em capturar o sentimento agregado a respeito de um tópico em um determinado instante, e não em realizar uma análise de sentimento precisa de cada

tweet. Por esse motivo utilizaram uma abordagem muito simples. Uma mensagem foi considerada positiva ou negativa dependendo da presença ou ausência de palavras que indicam o sentimento correspondente. Para isso, foi usado o dicionário de subjetividade OpinionFinder[4], que contém cerca de 1.600 e 1.200 palavras marcadas como positivas e negativas, respectivamente (JAIN; NEMADE, 2010). Com uma abordagem tão simples, era esperada uma alta taxa de erros. Mas, segundo os autores, uma alta taxa de erro apenas implica que o detector de sentimentos é um instrumento de medição ruidoso. Com um número razoavelmente grande de medições, esses erros se cancelam. A conclusão dos autores foi que as opiniões coletadas em pesquisas se correlacionam ao sentimento expressado em mensagens do Twitter que são postadas no mesmo período.

Patodkar e I.R (2010) mostraram como coletar automaticamente um corpus a partir de mensagens do Twitter e o utilizam para construir um classificador de sentimento. Os autores afirmam que a abordagem pode ser adaptada para vários idiomas, embora seu trabalho apenas utilize *tweets* em inglês.

Kiritchenko, Zhu e Mohammad (2014) desenvolveram um sistema que detecta o sentimento de mensagens de texto curtas e informais, como *tweets* e SMS (nível de mensagem) e o sentimento de uma palavra ou uma frase dentro de um mensagem (nível de termo). Eles utilizam um classificador estatístico supervisionado. Para classificar cada amostra utilizaram uma série de atributos baseados em algumas regras da forma superficial do texto levando em consideração as categorias léxicas. Também utilizaram atributos de vários dicionários de sentimentos. Eles empregaram tanto dicionários de propósito geral pré-existentes que foram manualmente criados, quanto dicionários que eles geraram automaticamente a partir de *tweets* com *hashtags* de palavra de sentimento e de *tweets* com *emoticons*. Para capturar adequadamente o sentimento das palavras negadas (por exemplo *"not good"*), um dicionário de sentimento separado foi gerado. Os experimentos mostraram que os novos dicionários específicos de *tweets* são superiores na previsão de sentimentos em ambientes não-supervisionados e supervisionados. O sistema ficou em primeiro lugar na tarefa *Sentiment Analysis in Twitter* (SemEval-2013, Tarefa 2), obtendo F-medida de 69,02% na tarefa em nível de mensagem e 88,93% na tarefa em nível de termo.

Os trabalhos citados neste capítulo foram escolhidos por serem trabalhos importantes que ilustram as duas principais abordagem ao problemas de análise de sentimento: supervisionada e não supervisionada. Na abordagem supervisionada, se faz uso de algoritmos supervisionados de aprendizado de máquina, e são necessárias amostras rotuladas (PANG; LEE; VAITHYANATHAN, 2002; DAVE et al., 2003). Na abordagem não supervisionada, geralmente se faz uso de algum tipo de dicionário de sentimento (TURNEY, 2002; TURNEY; LITTMAN, 2003).

[4] OpinionFinder. Disponível em <http://mpqa.cs.pitt.edu/>. Acessado em 21/09/2018

O uso de representação distribuída em problemas de análise de sentimento é abordado em alguns trabalhos mais recentes, alguns dos quais são apresentados na Seção 3.4. Outros trabalhos recentes abordam o uso de *Deep Learning* nesse tipo de problema. Esse termo se refere aos métodos geralmente baseados em redes neurais artificias, compostas de vários níveis hierárquicos, onde cada nível aprende a transformar os dados de entrada em uma representação um pouco mais abstrata e composta. Embora tenha sido decidido não abordar esse assunto neste trabalho, uma introdução interessante e uma lista de artigos relevantes na área podem ser encontrados em Zhang, Wang e Liu (2018).

Os trabalhos sobre análise de sentimento em mensagens curtas e ruidosas têm em comum o desafio de representar computacionalmente esse tipo de mensagem. As limitações das formas tradicionais de representação de texto em problemas típicos de análise de sentimento são abordadas no próximo capítulo. Nele, é explicado como esses problemas são agravados em mensagens curtas e ruidosas, e como eles são abordados habitualmente.

2 Representação computacional de texto

Em problemas de classificação de texto, uma amostra corresponde a um texto acompanhado de um rótulo. Para ser usado em métodos de classificação, o texto deve ser representado como sendo um conjunto de valores de atributos. Assim, o primeiro passo é extrair os elementos que serão utilizados como atributos, sendo que tradicionalmente o texto é decomposto em palavras. Entretanto, pode-se optar por extrair sequências de duas, três ou n palavras, que na área de linguística computacional são chamadas de bigramas, trigramas e n-gramas respectivamente. Independente da granularidade, esse tipo de representação é conhecido como *bag of words*.

A escolha da granularidade depende do problema. Embora Pang, Lee e Vaithyanathan (2002), por exemplo, tenham relatado que os unigramas superaram os bigramas nos experimentos realizados para detectar polaridade em críticas de filmes. Por outro lado, Dave et al. (2003) verificaram que em alguns cenários, os bigramas e os trigramas produziram melhores resultados na detecção de polaridade em críticas produtos.

No processo de representação computacional de textos, outra decisão que precisa ser tomada é quanto aos valores dos atributos. Para cada atributo, é possível designar um valor que indique a frequência ou a simples ocorrência da palavra (ou n-grama) no texto. Uma medida particularmente popular é o *tf-idf* (abreviação do inglês *term frequency-inverse document frequency*, que significa frequência do termo inverso da frequência nos documentos) (PANG; LEE, 2008). O valor *tf-idf* indica a importância de uma palavra em um documento em relação ao total de documentos. Esse valor aumenta proporcionalmente à medida que aumenta o número de ocorrências da palavra em um documento, ou seja, quanto menor o valor, mais rara a palavra.

Embora o uso de valores baseados em frequências, como o *tf-idf*, seja popular em problemas de classificação de texto, Pang, Lee e Vaithyanathan (2002) obtiveram melhor desempenho usando apenas a presença de unigramas, ao invés da frequência. Esse achado indica uma diferença interessante entre categorização de texto por tópico e classificação de polaridade: embora seja mais provável que um tópico seja enfatizado por ocorrências frequentes de determinadas palavras-chave, o sentimento geral usualmente não é evidenciado pelo uso repetido das mesmas palavras.

Em muitos problemas de classificação de texto, e em análise de sentimento especificamente, a simples ocorrência de palavras costuma ser usada como valores dos atributos. Para ilustrar esse tipo de representação, considere a Tabela 1 que contém uma lista de avaliações de filmes. Após excluir as palavras mais comuns, conhecidas como *stop-words*[1],

[1] Artigos, preposições, pronomes etc.

a representação de cada avaliação utilizando *bag of words* é apresentada na Tabela 2.

Tabela 1 – Exemplos de avaliações de filmes rotuladas como positivas (1) ou não (0).

Avaliação	**Positiva (+)?**
Detestei esse filme. Achei muito chato e parado.	0
Ótimo filme. O ator principal teve uma atuação excelente.	1
Fantástico! Muito bom mesmo.	1

Tabela 2 – Representação computacional das avaliações usando *bag of words* com valores de atributos indicando a ocorrência.

achei	ator	atuação	bom	chato	detestei	excelente	fantástico	filme	parado	principal	ótimo	+
1	0	0	0	1	1	0	0	1	1	0	0	0
0	1	1	0	0	0	1	0	1	0	1	1	1
0	0	0	1	0	0	0	1	0	0	0	0	1

Apesar de ser amplamente utilizado em representação de textos, o modelo *bag of words* traz uma série de deficiências bem conhecidas que serão discutidas a seguir.

A quantidade de dimensões pode ser muito alta

Para a maioria dos algoritmos de aprendizado de máquina, um dos problemas com o uso da *bag of words* é que a quantidade de dimensões pode ser alta, pois corresponde ao tamanho do vocabulário. Isso pode ocasionar super-ajustamento dos dados e/ou aumento considerável no custo computacional de treinamento. Além disso, como cada amostra contém apenas uma fração das palavras do vocabulário, existe o problema da alta esparsidade. No exemplo artificialmente simples da Tabela 2, com 3 amostras e 12 palavras (atributos), isso já pode ser observado. A maioria das palavras não está presente na maioria das amostras. Esse fato se agrava em problemas reais que podem contar com milhões de amostras e com milhares de palavras no vocabulário.

A informação sobre a posição relativa dos termos no texto é perdida

Para alguns problemas de classificação de texto, a simples ocorrência de determinadas palavras pode ser suficiente para classificar uma mensagem corretamente. Contudo, para problemas como análise de sentimento, a ordem das palavras pode ser muito importante. As frases "O vírus destruiu o sistema imunológico do paciente." e "O sistema imunológico do paciente destruiu o vírus." contém exatamente as mesmas palavras, mas têm significados diferentes. Assim, saber somente quais palavras ocorrem em cada frase pode não ser suficiente para que ela seja classificada corretamente.

Sinônimos, gírias, abreviações e erros de escrita

Abreviações e gírias são muito usadas, especialmente no caso de textos curtos, como nas mensagens do Twitter. Por exemplo, a mensagem "*dz ne1 knw h2 ripair dis*

terrible LPT?", traduzida para o inglês formal equivaleria a: "*Does any one know how to repair this terrible printer?*". O grande problema com essas palavras é que elas representam diferentes maneiras de se referir a mesma coisa, o que corresponde ao mesmo problema dos sinônimos. No exemplo acima, "*LTP*" equivale a "*printer*" (impressora). Mesmo que "*LTP*" fizesse parte do vocabulário, o algoritmo de aprendizado não teria como associar "*LTP*" com "*printer*". Um modelo de classificação que tivesse sido treinado com a amostra "Esse filme é sensacional" como sendo uma avaliação positiva, não levaria em consideração o fato de "sensacional" ser sinônimo de "incrível" ao ser usado para classificar "Esse filme é incrível", pois essas duas palavras são atributos diferentes que para o modelo não têm nenhuma relação.

Ambiguidade

Algumas palavras podem ter vários significados diferentes. Nas frases "Meu caro amigo" e "Achei aquele celular muito caro", as ocorrências da palavra "caro" denotam sentimento positivo e negativo respectivamente. Geralmente, o significado correto pode ser determinado pelo contexto onde a palavra ocorre. Entretanto, com o uso de *bag of words*, o contexto é descartado, e essa distinção não pode ser feita.

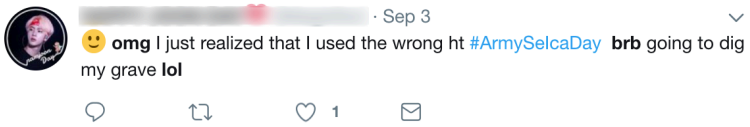
2.1 Representação computacional de mensagens curtas e ruidosas

Na representação computacional de mensagens curtas, como o caso de SMS ou *tweets*, alguns dos problemas citados se agravam. O limite de caracteres imposto para o tamanho dessas mensagens faz com que a tarefa de análise de sentimento seja particularmente mais difícil. Menos caracteres equivale a menos palavras, o que faz com que cada amostra usada para treinar os algoritmos de classificação contenha menos atributos.

O uso dos *smartphones* para acessar as redes sociais, como alternativa ao computador, amplifica esse problema. Além do texto ter que ser reduzido, geralmente as pessoas escrevem rapidamente, utilizando o teclado diminuto do *smartphone* com apenas dois dedos, sendo que as mãos são frequentemente utilizadas para segurar o celular e digitar o texto ao mesmo tempo. Devido a união desses fatores, os usuários criaram uma nova linguagem para comunicar suas mensagens com a maior brevidade possível. Embora essa brevidade permita que as mensagens transmitam mais informações usando menos caracteres, elas dificultam a representação computacional devido à falta de padronização.

Essa linguagem extremamente coloquial contém uma quantidade alta de repetições, acrônimos e de neologismos. Acrônimos são termos que se formam pela junção das primeiras letras ou das sílabas iniciais de um grupo de termos. Na mensagem ilustrada pela Figura 3, *"omg"*, *"brb"* e *"lol"* são usados no lugar de *"oh my god"*, *"be right back"* e *"laugh out loud"*, respectivamente.

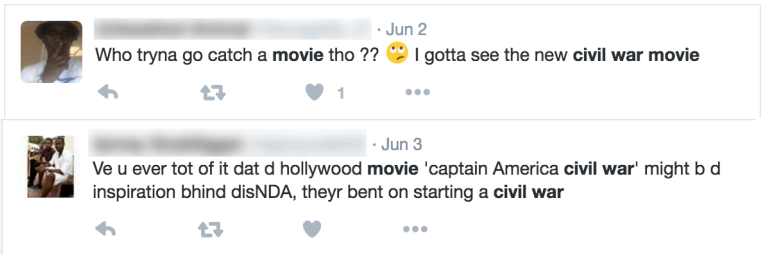
Figura 3 – Exemplo de mensagem com acrônimos no Twitter.



Fonte: Twitter.

Outro recurso muito usado nessa nova linguagem é a ortografia fonética, ou seja, palavras escritas de maneira errada cuja pronuncia é similar a pronuncia da palavra correta. Isso pode ser observado na primeira mensagem ilustrada pela Figura 4, onde o autor usa a palavra *"tryna"* no lugar de *"trying to"*. A segunda mensagem presente na mesma figura contém ainda mais exemplos de ortografia fonética e erros de escrita.

Figura 4 – Exemplo de mensagens ruidosas no Twitter.



Fonte: Twitter.

Nesse tipo de mensagem, também existe pouca consideração pelo uso adequado de letras maiúsculas e pontuações. Letras maiúsculas podem sinalizar um nome próprio ou o fim da sentença, mas também podem ser usadas para algo tão arbitrário quanto enfatizar um determinado segmento. A pontuação pode determinar os limites da sentença, mas também pode ser empregada para criar *emoticons*, como ":(" e ";-)", que são usados para expressar emoções, sentimentos ou ideias.

Diversas técnicas vem sendo propostas e utilizadas para melhorar a representação computacional de textos curtos e ruidosos, sendo que as mais tradicionais envolvem normalização léxica, indexação semântica e desambiguação de sentido das palavras.

2.1.1 Normalização léxica

Tradicionalmente, para lidar com os problemas das gírias, abreviações e erros de escrita, são utilizadas técnicas de normalização léxica para traduzir variantes de palavras e expressões para sua forma canônica. No caso de mensagens curtas e ruidosas, embora haja similaridade com a tarefa de correção ortográfica, a tarefa de normalização léxica é mais desafiadora porque em muitos casos as palavras são intencionalmente escritas de maneira errada, devido ao limite de caracteres, ou para adotar o estilo característico do

meio utilizado. Por exemplo, substituir *"b4"* por *"before"* está além da capacidade de um simples corretor ortográfico, assim como é o caso da repetição de letras para dar ênfase como em *"goooood"*.

Kaufmann (2010) descreve uma abordagem para normalização léxica de *tweets* em duas etapas. Na primeira, os *tweets* são pré-processados. É realizada uma normalização ortográfica que simplesmente identifica e corrige os erros de ortografia intencionais mais comuns, para evitar que palavras escritas intencionalmente de maneira errada (possíveis neologismos) sejam substituídas. Palavras com erros ortográficos que contêm repetições de letras, têm essas letras removidas. Se a remoção dessas letras criar uma palavra correta, a palavra com a ortografia incorreta será substituída no texto. Da mesma forma, a pontuação repetida é encurtada para um sinal de pontuação. Em seguida é feita uma desambiguação sintática, que processa meta-caracteres como "@" e "#", que têm um significado especial em mensagens do Twitter. Na segunda etapa, os *tweets* são processados por um sistema de tradução de máquina estatística, que foi previamente treinado. O autor mostrou que combinando o software de tradução de máquina estatística com um pré-processador, é possível remover a maioria dos ruídos de um *tweet* e aumentar sua legibilidade de forma significativa.

Han e Baldwin (2011) adotaram uma estratégia diferente. Eles criaram um método que usa um classificador pré-treinado para detectar palavras mal-formadas e gerar candidatas à correção baseado na similaridade morfofonêmica. Sua abordagem dispensa dados previamente anotados. Ela utiliza dicionários e leva em consideração tanto a semelhança quanto o contexto da palavra para selecionar a candidata mais provável para a correção. Segundo os autores, seu método obteve resultados em linha com o estado-da-arte em mensagens de SMS e do Twitter.

2.1.2 Dicionários semânticos e desambiguação de sentido da palavra

Para abordar o problema dos sinônimos e da polissemia, tradicionalmente se faz uso de dicionários semânticos e algoritmos de desambiguação do sentido da palavra. Dicionários semânticos como *WordNet* (MILLER, 1995) ou *BabelNet* (NAVIGLI; PONZETTO, 2010), relacionam palavras aos seus diferentes sentidos. Desambiguação de sentido de uma palavra (do inglês, *word sense disambiguation*) é a atividade de encontrar, dada a ocorrência em uma palavra ambígua no texto, o sentido específico daquela ocorrência de acordo com o contexto em que ela ocorre. Por exemplo, a palavra *bank* pode ter (pelo menos) dois sentidos diferentes em inglês, podendo ser uma instituição financeira ou a margem de um rio. A seguir, são apresentados dois exemplos do uso dos diferentes sentidos da palavra *bank*:

*I must go to the **bank** and change some money.*
*We pushed the boat off from the river **bank**.*

Hidalgo, Rodríguez e Pérez (2005) evidenciam o papel de algoritmos de desambiguação de sentido da palavra em problemas de categorização de texto. Eles descrevem dois métodos oriundos da área de Recuperação de Informação que podem ser utilizados em categorização de texto. Na **indexação de conceitos** (também chamada de indexação semântica), as palavras de um documento e das consultas são substituídas por uma representação da classe semântica a qual elas referem. Dessa maneira, apenas o sentido correto da palavra é utilizado nas buscas, evitando resultados errados causados por polissemia. Na **expansão de consulta**, cada palavra de cada documento é substituída por todos os seus sinônimos, fazendo com que isso possa melhorar os resultados da consulta. Os autores realizaram uma série de experimentos aplicados em classificação de textos que sofrem com os mesmos problemas de variabilidade de linguagem. Os resultados comprovaram a eficácia dessas técnicas nesses tipos de problema, e o papel chave da desambiguação de sentido para obter bons resultados.

Para melhoria de desempenho na tarefa de filtrar *spam* em *Instant Messaging* e SMS, Almeida et al. (2016) propuseram um sistema que utiliza normalização léxica e indexação semântica, em uma fase de pré-processamento, e comprovaram, após análise estatística dos resultados, que o uso de tais métodos realmente oferece aumento de desempenho nas técnicas de classificação. As técnicas utilizadas pelos autores foram posteriormente empregadas em um sistema de análise de sentimento chamado SentMiner (LOCHTER, 2015; LOCHTER et al., 2016), que será detalhado na próxima seção.

2.2 O SentMiner

As mesmas técnicas de normalização léxica e indexação semântica utilizadas em Almeida et al. (2016) foram posteriormente aplicadas por Lochter (2015) em problemas de análise de sentimento. Nas palavras do autor:

> O método proposto seleciona automaticamente a melhor combinação entre técnicas de normalização e indexação semântica e métodos de classificação, considerados estado da arte tanto em processamento de linguagem natural quanto em aprendizado de máquina. Com isso, a abordagem proposta é capaz de gerar hipóteses mais robustas e genéricas que podem conduzir a desempenhos superiores aos métodos isolados e disponíveis na literatura (LOCHTER, 2015, p.58).

Sua pesquisa culminou na criação do *SentMiner*, ferramenta disponível online dentro da suite de ferramentas ML-Tools[2]. O SentMiner é um comitê de máquinas de classificação, ou seja, vários métodos de classificação são combinados com técnicas de proces-

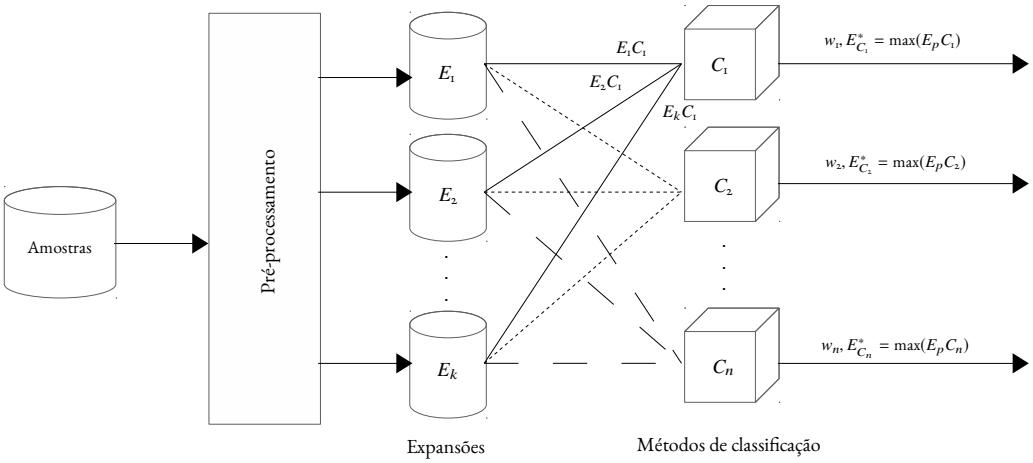
[2] ML-Tools. Disponível em <http://lasid.sor.ufscar.br/ml-tools>. Acessado em 30/07/2018

samento de textos com o intuito de obter um bom desempenho na tarefa de análise de sentimento de mensagens curtas e ruidosas.

Inicialmente, existe uma etapa de seleção do modelo na qual são determinados os parâmetros de cada método de classificação $(C_1, \ldots, C_n)$. Para isso, é feita uma busca em grade com um subconjunto estratificado e aleatório das amostras. Outra atividade realizada nesta etapa é a expansão das amostras. Existem diferentes regras de expansão que podem ser aplicadas às amostras, sendo que cada qual representa uma combinação das técnicas de normalização léxica, geração de conceitos e/ou desambiguação dos sentidos das palavras. Deu-se o nome de expansor para cada possível combinação dessas técnicas.

Na etapa de seleção de modelo, as amostras são processadas com cada expansor $(E_1, \ldots, E_k)$. Em seguida, cada base resultante da expansão é usada no treinamento de cada método de classificação $(C_1, \ldots, C_n)$. O desempenho de cada combinação expansor-classificador $(Ep \rightarrow Cj)$ é avaliado, para definir qual é a combinação mais adequada $(E_{c*} = max(E_p, C_j) \forall p \in \{1, \ldots, k\}, j \in \{1, \ldots, n\})$. Finalmente, um peso w_j (grau de confiança) é calculado para cada combinação j, baseado na sua acurácia individual comparada com aquela que obteve melhor acurácia no sistema. A Figura 5 ilustra a etapa de seleção de modelo.

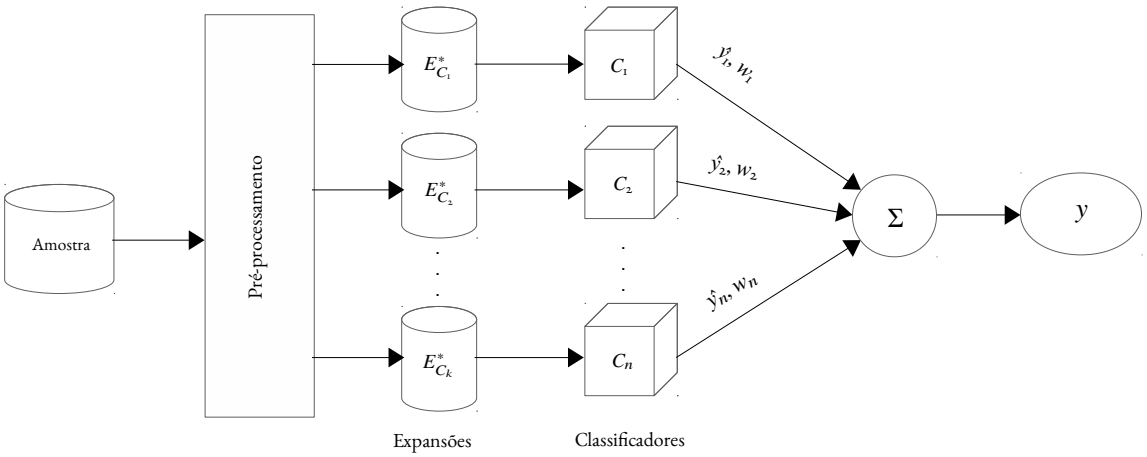
Figura 5 – Etapa de seleção de modelo do SentMiner.



Fonte: Lochter (2015, p.56).

Finalizada a etapa de seleção de modelo, os métodos de classificação são treinados com o resultado das respectivas expansões definidas na etapa anterior. Para predizer o rótulo de uma nova amostra, ela é pré-processada com cada expansor selecionado para cada classificador, e depois classificada com cada um dos métodos, que envia a predição a um concentrador (Σ). Cada método emite uma predição $(\hat{y}_j)$ com um grau de confiança (w_j) e o voto final é computado pelo voto majoritário ponderado, conforme ilustra a Figura 6.

Figura 6 – Etapa de classificação no SentMiner.



Fonte: Lochter (2015, p.57).

Lochter (2015) comparou o desempenho do comitê de classificadores com os métodos de classificação individuais. Ele demonstrou que o comitê foi estatisticamente superior a qualquer outro método. A utilização dos métodos de expansão das amostras com dicionários semânticos e desambiguação promoveu um desempenho superior em 35,8% dos casos analisados (LOCHTER, 2015, p.73).

A indexação semântica e expansão de amostras, embora tenham ajudado em alguns casos, também podem piorar o resultado, quando a desambiguação não é feita, ou é feita de maneira imperfeita (HIDALGO; RODRÍGUEZ; PÉREZ, 2005). Lochter não conseguiu uma melhoria de desempenho na maioria dos casos usando esses métodos (LOCHTER, 2015, p.73). Outro problema é que esses métodos possuem alto custo computacional, sendo que cada palavra da amostra, junto com seu contexto, devem ser analisados. O maior problema, entretanto, é que todos esses métodos são dependentes de dicionários. As línguas evoluem com o tempo, e cada vez mais frequentemente surgem novas palavras ou novos significados são atribuídos a palavras existentes. Nas redes sociais, novos termos surgem e se espalham rapidamente, passando a fazer parte do "dialeto" utilizado. Além disso, para classificar textos em diferentes línguas são necessários diferentes dicionários, sendo que as atualizações constantes podem ter um custo proibitivo em cenários de aplicações reais.

Para contornar essas limitações, o uso de *representações vetoriais distribuídas de palavras e parágrafos* é uma alternativa que dispensa o emprego de dicionários, e tem o potencial de resolver os principais problemas relacionados à *bag of words*. As representações distribuídas são geradas de maneira não supervisionada a partir de um grande volume de texto, sendo que nesses modelos cada palavra corresponde a um vetor de baixa dimensionalidade. Palavras que aparecem em contextos similares no texto corresponderão a vetores próximos nesse espaço. Dessa maneira, a similaridade semântica entre as palavras

é preservada.

Como a similaridade semântica é preservada, algoritmos de aprendizado podem aprender com palavras diferentes mas com significados similares, como no exemplo de "incrível" e "sensacional". Devido ao fato dos modelos serem gerados automaticamente a partir de uma grande quantidade de texto, eles podem incorporar gírias e abreviações sem a constante necessidade de atualizar dicionários. Por exemplo, é esperado que o termo "*hj*" seja representado por um vetor não muito distante do vetor que representa a palavra "hoje". O próximo capítulo descreve os detalhes de funcionamento das principais técnicas existentes para gerar esse tipo de representação.

3 Representação vetorial distribuída de texto

A ideia de que o contexto fornece uma boa aproximação para os significados das palavras é a base da hipótese distributiva da área da semântica distributiva (SAHLGREN, 2008). Essa ideia tem suas raízes teóricas na linguística estruturalista e na filosofia da linguagem, aparecendo em obras importantes na década de 1950. A Figura 7 ilustra essa ideia. Nela, é possível observar algumas ocorrências da palavra "olhos" e o contexto em que elas ocorrem no livro Memórias Póstumas de Brás Cubas, de Machado de Assis.

Figura 7 – Contextos onde a palavra "olhos" aparece no livro Memórias Póstumas de Brás Cubas, de Machado de Assis (ASSIS, 1994).

```
à cabeceira da cama, com os olhos estúpidos, a boca entreaberta
Pela minha parte fechei os olhos e deixei-me ir à ventura.
Como ia de olhos fechados, não via o caminho;
Com efeito, abri os olhos e vi que o meu animal galopava
Então, encarei-a com olhos súplices, e pedi mais alguns
e, não obstante, porque os olhos do delírio são outros, eu via
de estômagos satisfeitos; os olhos moles e úmidos, ou vivos e cá
cado de orquestra, e todos os olhos se voltavam para o glosador.
ou nada comi, porque só tinha olhos para a dona da casa.
que junto à amurada, tinha os olhos fitos no horizonte.
me que não; ao mesmo tempo os olhos me contavam que, já outrora
não souberam ver-lha; eram olhos da primeira edição.
sorriso de outro tempo, e nos olhos uma concentração de luz, que
de um lado para outro, com os olhos no chão.
andando, concentrado, com os olhos para baixo ou para a frente
Nesse instante é que os olhos se fixam na ponta do nariz.
```

Fonte: Autoria Própria.

Uma maneira simples e ilustrativa de gerar vetores de palavras é contar o número de vezes que cada palavra aparece dentro de uma janela de um tamanho específico, em torno da palavra de interesse. Essa contagem é feita para todas as palavras presentes no corpus. Considere o exemplo a seguir, que utiliza um corpus com apenas três sentenças e tamanho da janela igual a 1.

- I like deep learning
- I like NLP
- I enjoy flying

Tabela 3 – Matriz de co-ocorrências.

	I	like	enjoy	deep	learning	NLP	flying
I	0	2	1	0	0	0	0
like	2	0	0	1	0	1	0
enjoy	1	0	0	0	0	0	1
deep	0	1	0	0	1	0	0
learning	0	0	0	1	0	0	0
NLP	0	1	0	0	0	0	0
flying	0	0	1	0	0	0	0

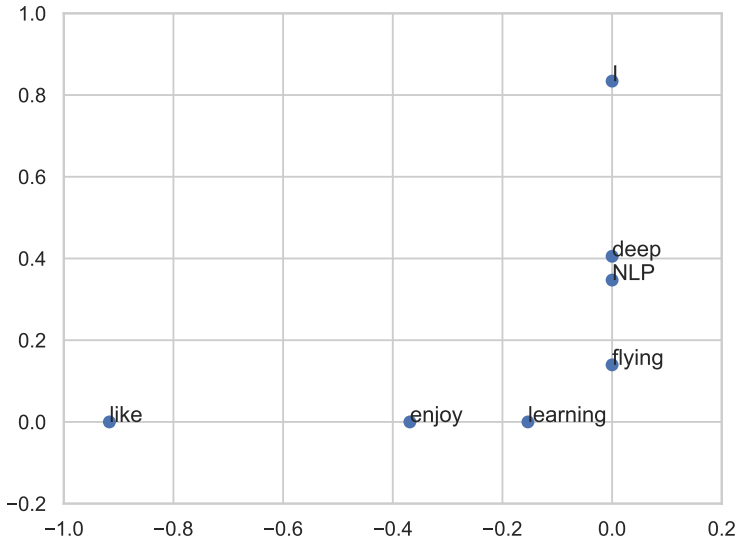
A Tabela 3 mostra a matriz de co-ocorrências das palavras. Nota-se que cada palavra é representada por um vetor, que pode ser uma linha ou uma coluna da matriz. Essa abordagem, embora seja simples, apresenta alguns problemas. As dimensões da matriz podem ser grandes, sendo que cada vetor de palavra é do tamanho do vocabulário, e a matriz é extremamente esparsa sendo que a maioria das palavras não co-ocorrem. Para minimizar esses problemas, geralmente utiliza-se métodos para reduzir a dimensionalidade, como o SVD (do inglês *singular value decomposition*), produzindo matrizes mais compactas. A Análise Semântica Latente (TURNEY; PANTEL, 2010), por exemplo, é um método que utiliza matrizes de co-ocorrências de palavras e SVD para gerar tais modelos.

A Figura 8 ilustra os vetores da Tabela 3 projetados em um espaço bi-dimensional utilizando SVD. Observa-se que palavras que ocorrem em contextos similares correspondem a pontos próximos no gráfico. A redução da dimensionalidade aplicando SVD entretanto, gera outros problemas. Por exemplo, o processo não é incremental. Assim, cada vez que o corpus muda, é necessário calcular o SVD da matriz toda novamente, que tem um custo computacional quadrático em relação a quantidade de termos.

Bengio et al. (2003) propuseram uma alternativa radicalmente diferente à construção tradicional dos modelos de linguagem. Ao invés de primeiro coletar a matriz de co-ocorrências e depois aplicar transformações com base em vários critérios, vetores de palavras passaram a corresponder aos pesos de uma rede neural artificial, que são otimizados para maximizar a probabilidade dos contextos em que a palavra é observada no corpus. Eles cunharam o termo ***word embeddings*** para se referir a esses vetores. Os primeiros a mostrar a utilidade dos vetores de palavras pré-treinados foram Collobert e Weston (2008), no artigo que se tornou referência na área. Os autores não só estabelecem as *word embeddings* como ferramenta útil para tarefas relacionadas, mas também introduzem uma arquitetura de rede neural que constitui o alicerce de muitas abordagens atuais.

Nas próximas seções, serão detalhados os principais métodos para gerar representações vetoriais distribuídas de textos que surgiram após o trabalho de Bengio et al. (2003).

Figura 8 – Projeção em 2D de um modelo vetorial de palavras.



Fonte: Autoria Própria.

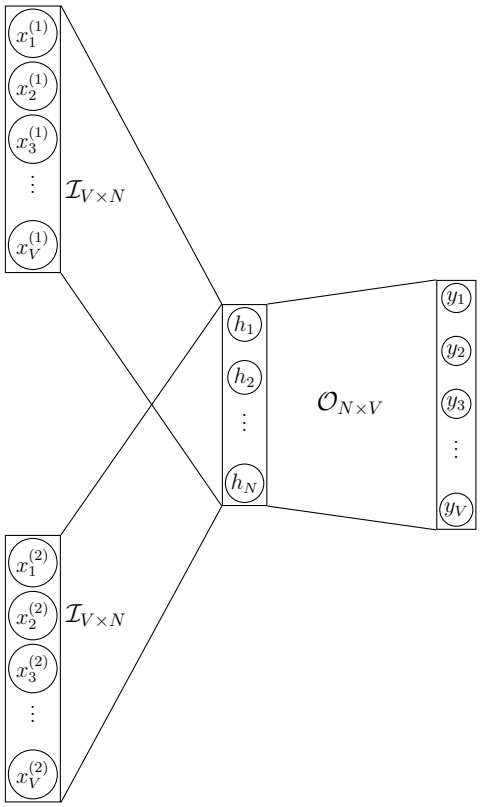
3.1 CBOW e Skip-gram

Um problema do uso de redes neurais para gerar modelos de linguagem distribuídos é que para capturar as muitas similaridades semânticas é preciso utilizar corpora muito grandes. Isso faz com que a rede neural precise ter muitos neurônios, tornando o tempo de treinamento proibitivamente longo. Mikolov et al. (2013b) propuseram dois novos métodos que possibilitaram gerar esses modelos com corpora bem maiores do que os usados anteriormente, fazendo ressurgir o interesse nesses modelos. Os métodos propostos foram: *continuous bag of words* (CBOW) e Skip-gram.

A arquitetura da rede neural do método CBOW é mostrada na Figura 9. A rede tem três camadas: entrada, projeção e saída. O objetivo do treinamento, nesta arquitetura, é maximizar a probabilidade condicional de inferir uma palavra alvo, dadas as palavras do contexto. Para treinar a rede, o corpus é percorrido da primeira até a última palavra. Em cada nova iteração, considera-se a próxima palavra como sendo a palavra alvo, e as palavras do contexto são as que ocorrem antes e depois da palavra alvo, levando em consideração uma distância que é chamada de janela. Assim, se o tamanho da janela é 1, por exemplo, a cada iteração haverá duas palavras de contexto, a que ocorre imediatamente antes e a que ocorre imediatamente depois da palavra alvo. Na Figura 10, pode-se ver as diferentes palavras alvo e as respectivas palavras de contexto que se obtêm ao varrer um corpus ilustrativo.

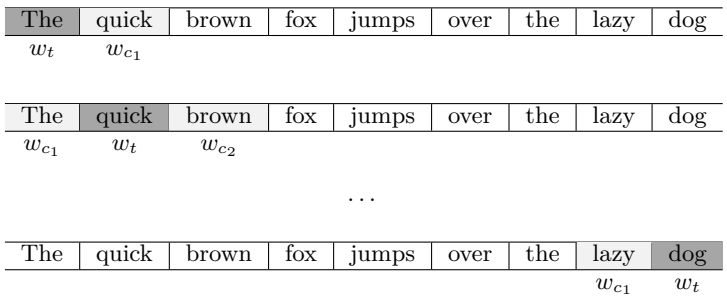
Na camada de entrada, existem C grupos de neurônios – um grupo de neurônios para cada palavra do contexto. O número de neurônios em cada um desses grupos vai ser igual ao tamanho do vocabulário V. Cada palavra do contexto é transformada em um

Figura 9 – Arquitetura da rede neural do método CBOW.



Fonte: Autoria Própria.

Figura 10 – Palavras alvo (w_t) e de contexto (w_{c1} e w_{c2}) utilizando janela de tamanho 1.



Fonte: Autoria Própria.

vetor codificado utilizando o padrão *one-hot*. Nesse padrão, um dos V valores do vetor $\{x_1, x_2, ..., x_V\}$ será 1 e os outros 0. Assim, ao fornecer a entrada na rede neural, um dos neurônios de cada grupo será ativado e os outros não.

A conexão entre a camada de entrada e a camada de projeção é completa, e os pesos utilizados para a ativação da camada de projeção residem na matriz $\mathcal{I}$, que é compartilhada por todos os grupos de neurônios da camada de entrada. Sua dimensão é $V \times N$, sendo V o tamanho do vocabulário e N é a dimensão dos vetores de palavras. Entre a camada de projeção e a camada de saída existe outra matriz de pesos $\mathcal{O}$ com

dimensão $N \times V$. Essas duas matrizes $\mathcal{I}$ e $\mathcal{O}$ armazenam os pesos da rede neural. Cada linha da matriz de entrada e cada coluna da matriz de saída correspondem a um vetor de uma palavra, assim, cada palavra tem duas representações no CBOW. A camada de saída é do tamanho do vocabulário, contendo V neurônios, e o valor esperado em cada neurônio é a probabilidade da palavra representada por aquele neurônio ser a palavra alvo. A soma de todas as probabilidades deve ser 1.

No início do treinamento, as matrizes são inicializadas com valores aleatórios. A partir disso, o corpus é percorrido e são determinadas a palavra alvo e as palavras de contexto, que são convertidas no padrão *one-hot*. As palavras do contexto alimentam os grupos de neurônio na camada de entrada. A ativação dos neurônios da camada oculta é dada pela média dos vetores correspondentes às palavras de entrada. Essa seleção dos vetores pode ser feita utilizando multiplicação. Como as palavras na camada de entrada são vetores *one-hot*, ao multiplicar esse vetor de entrada pela matriz $\mathcal{I}$ obtêm-se um resultado onde apenas uma linha (vetor da palavra) é diferente de zero. A média é obtida somando-se cada uma dessas linhas resultantes e dividindo a soma pelo número de linhas, que equivale ao número de palavras de contexto (C), conforme a Equação 3.1.

$$h = \frac{\mathcal{I}(x^{(1)} + x^{(2)} + \cdots + x^{(c)})}{C} \tag{3.1}$$

Após calculado h, é possível calcular a função de custo da rede neural. Utiliza-se aqui a entropia cruzada, dada na Equação 3.2.

$$H(\hat{y}, y) = -\sum_{j=1}^{V} y_j \log(\hat{y}_j) \tag{3.2}$$

Como y é um vetor *one-hot*, apenas um dos elementos do vetor tem valor 1, e todos os outros 0, então é possível simplificar a equação, chegando na Equação 3.3.

$$H(\hat{y}, y) = -y_t \log(\hat{y}_t) \tag{3.3}$$

A função *softmax* dada na Equação 3.4, retorna a probabilidade da palavra representada pela j-ésima coluna da matriz $\mathcal{O}$ ser a palavra alvo, dado que h já tenha sido calculado.

$$\hat{y}_j = \frac{\exp(h \times \mathcal{O}_{:,j})}{\sum_{i=1}^{V} \exp(h \times \mathcal{O}_{:,i})} \tag{3.4}$$

Como $y_t = 1$ (palavra alvo), obtêm-se a função de custo apresentada na Equação 3.5, onde w_t corresponde à palavra alvo e $w_{c_1}, w_{c_2}..., w_{c_n}$ são as palavras de contexto.

$$
\begin{aligned}
J &= -\log P(w_t|w_{c_1}, w_{c_2}..., w_{c_n}) \\
&= -\log P(\mathcal{O}_{:,t}|h) \\
&= -\log \frac{\exp(h \times \mathcal{O}_{:,t})}{\sum_{i=1}^{V} \exp(h \times \mathcal{O}_{:,i})} \\
&= \log \sum_{i=1}^{V} \exp(h \times \mathcal{O}_{:,i}) - \exp(h \times \mathcal{O}_{:,t})
\end{aligned} \tag{3.5}
$$

O objetivo se torna então minimizar a função de custo J. A cada iteração, após a palavra alvo e as palavras de contexto terem sido determinadas, e h ter sido calculado usando vetores da matriz $\mathcal{I}$, todas as linhas de $\mathcal{O}$ são utilizadas para calcular o custo J, levando em consideração o vetor da palavra alvo, $\mathcal{O}_{:,t}$. Após o custo ser calculado, os valores dos pesos da rede são ajustados pelo algoritmo de gradiente descendente utilizando *backpropagation*.

Assim, a cada iteração os pesos da rede que correspondem aos vetores de palavras são ajustados para maximizar a probabilidade de encontrar a palavra alvo, dadas as palavras do contexto. Dessa maneira, espera-se que os vetores de palavras que ocorrem em contexto similares, terão valores similares, ou seja, serão pontos próximos entre si no espaço N-dimensional de vetores de palavras.

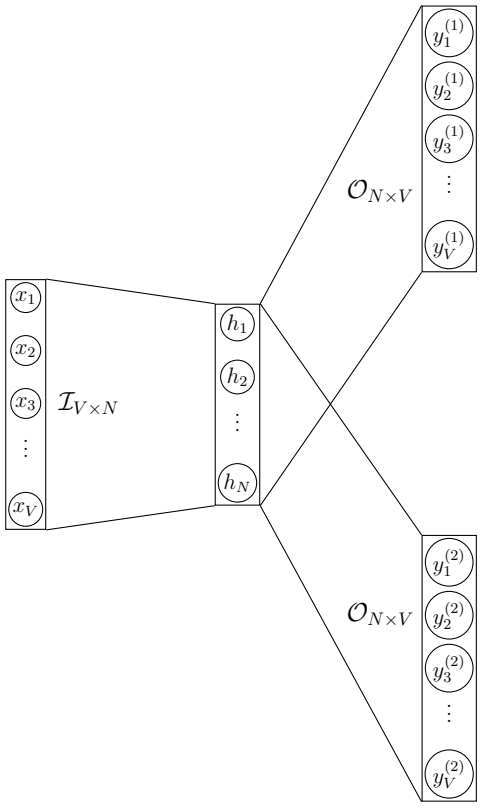
3.1.1 Skip-gram

A arquitetura do método Skip-gram é semelhante à do CBOW, também com três camadas: entrada, projeção e saída. A direção da predição, entretanto, é invertida (Figura 11). Neste caso, o objetivo é maximizar a probabilidade condicional de inferir as palavras do contexto dada uma palavra alvo. A camada de entrada tem tamanho V, pois apenas uma palavra é apresentada para a rede a cada janela analisada no corpo de texto. Essa palavra é codificada como um vetor no padrão *one-hot*. Assim como no CBOW, existem duas matrizes $\mathcal{I}_{V\times N}$ e $\mathcal{O}_{N\times V}$ que armazenam os pesos da rede neural. A camada de saída no método Skip-gram é composta por C grupos de neurônios, que compartilham os mesmos pesos ($\mathcal{O}$), de maneira análoga à camada de entrada do método CBOW.

Assim como no CBOW, no início do treinamento as matrizes são inicializadas com valores aleatórios. A partir disso, o corpo de texto é percorrido, são determinadas a palavra alvo e as palavras de contexto, que são convertidas no padrão *one-hot*. No caso do Skip-gram, a camada de projeção é ativada como uma cópia da linha da matriz $\mathcal{I}$ correspondente a palavra de entrada (Equação 3.6).

$$
h = x^T \times \mathcal{I} \tag{3.6}
$$

Figura 11 – Arquitetura da rede neural do método Skip-gram.



Fonte: Autoria Própria.

Após calculado h, é possível calcular o custo. No caso do Skip-gram, também se usa entropia cruzada e *softmax* para calcular a probabilidade do contexto dada a palavra alvo. Uma diferença fundamental é que aqui se assume uma suposição de independência ou seja, dada a palavra alvo, todas as palavras do contexto são consideradas completamente independentes umas das outras, e por esse motivo as probabilidades são multiplicadas. A Equação 3.7 apresenta a função de custo, onde $\mathcal{O}_{:,c_m}$ equivale a coluna na matriz $\mathcal{O}$ que é determinada pela m-ésima palavra de contexto.

$$\begin{aligned} J &= -\log P(w_{c_1}, w_{c_2}..., w_{c_n}|w_t) \\ &= -\log \prod_{m=1}^{C} P(\mathcal{O}_{:,c_m}|h) \\ &= -\log \prod_{m=1}^{C} \frac{\exp(h \times \mathcal{O}_{:,c_m})}{\sum_{i=1}^{V} \exp(h \times \mathcal{O}_{:,i})} \\ &= C \cdot \log \sum_{i=1}^{V} \exp(h \times \mathcal{O}_{:,i}) - \sum_{m=1}^{C} h \times \mathcal{O}_{:,c_m} \end{aligned} \tag{3.7}$$

O treinamento ocorre de maneira similar ao método CBOW. Após o custo ser calculado, os valores dos pesos da rede são ajustados pelo algoritmo de gradiente descendente

utilizando *backpropagation*. O resultado esperado é similar, mas ao se comparar as duas arquiteturas verifica-se que o CBOW suaviza a distribuição estatística ao calcular a média dos termos do contexto. Com poucos dados, esse efeito de regularização do CBOW acaba tendo utilidade, mas, por outro lado, o Skip-gram é capaz de extrair mais informações quando há mais dados disponíveis (MIKOLOV et al., 2013a).

Um problema que ocorre em ambos os modelos é o custo computacional do treinamento da rede neural. Isso acontece devido ao fato de que, tanto no CBOW quanto no Skip-gram, é necessário calcular a soma da Equação 3.8 para cada amostra (palavra alvo + palavras de contexto).

$$\sum_{i=1}^{V} \exp(h \times \mathcal{O}_{:,i}) \tag{3.8}$$

Para ilustrar esse problema, é possível considerar um exemplo onde a rede é treinada com vetores de palavras de 300 componentes e vocabulário de 10.000 palavras. Nesse caso, seriam necessárias 10.000 multiplicações de matrizes e a soma dos 10.000 valores resultantes das multiplicações realizadas tantas vezes quantas forem as palavras do corpo de texto. Além disso, teriam duas matrizes de peso com $300 \times 10.000 = 3$ milhões de pesos cada uma. Assim, para cada janela também seria necessário atualizar pelo menos 3 milhões de pesos (da matriz de entrada $\mathcal{I}$, somente os pesos correspondes às palavras de entrada precisam ser atualizados). Com uma quantidade tão grande de neurônios seria necessária uma quantidade enorme de dados de treinamento para ajustar todos os pesos evitando *overfitting*. Milhões de pesos multiplicados por possíveis bilhões de amostras de treinamento significa que este modelo seria inviável de ser treinado.

Mikolov et al. (2013a) apresentaram algumas estratégias para abordar esse problema. Eles criaram mecanismos para identificar pares ou grupos de palavras que ocorrem juntas com frequência (por exemplo "*New York*"), e tratam esses grupos como se fossem uma única palavra, reduzindo assim o tamanho do vocabulário. Outra melhoria, foi a implementação de sub-amostragem de palavras frequentes. Essa funcionalidade recebe dois parâmetros: t e p. Quando o número de ocorrências de uma palavra ultrapassa o limite t, novas ocorrências são removidas aleatoriamente com uma probabilidade p. Essa ação tem efeito semelhante a remoção de *stop-words*, e parte do princípio que palavras muito frequentes contribuem pouco para o objetivo de treinamento, levando em consideração o custo computacional que elas incorrem. Os autores também implementaram duas alternativas ao *softmax*: com a **amostragem negativa**, onde cada amostra de treinamento atualiza apenas uma pequena porcentagem dos pesos do modelo; e o ***softmax* hierárquico**, que permite calcular a probabilidade de cada palavra do vocabulário de maneira bem mais rápida. Ambas alternativas serão discutidas a seguir.

3.1.2 Amostragem negativa

A ideia da amostragem negativa é simples. Ao invés de serem atualizados todos os pesos da matriz de saída, são atualizados apenas uma amostra deles. Em ambos os modelos, a camada de saída da rede neural corresponde a um ou mais vetores codificados no padrão *one-hot*. No CBOW tem-se um vetor correspondente à palavra alvo, no Skip-gram tem-se vários vetores que correspondem às palavras do contexto. Assim, na camada de saída de ambos os modelos, sabe-se quais as palavras são esperadas, ou seja, quais neurônios deveriam ser ativados e quais não deveriam. As palavras correspondentes aos neurônios que deveriam ser ativados podem ser chamadas de "positivas", e as outras "negativas". A técnica de amostragem negativa consiste em selecionar aleatoriamente algumas palavras negativas, além da palavra positiva, para que somente os pesos correspondentes a essas palavras sejam atualizados.

Embora a amostragem negativa seja baseada no Skip-gram, ela utiliza uma função de custo diferente. Ao invés de usar uma forma de amostragem negativa que produz uma distribuição multinomial posterior bem definida, os autores (MIKOLOV et al., 2013a) argumentam que a função de custo da Equação 3.10 é capaz de produzir vetores de palavras de alta qualidade.

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \tag{3.9}$$

$$J = -\log \sigma(h \times \mathcal{O}_{:,j}) - \sum_{i \in neg} \sigma(-h \times \mathcal{O}_{:,i}) \tag{3.10}$$

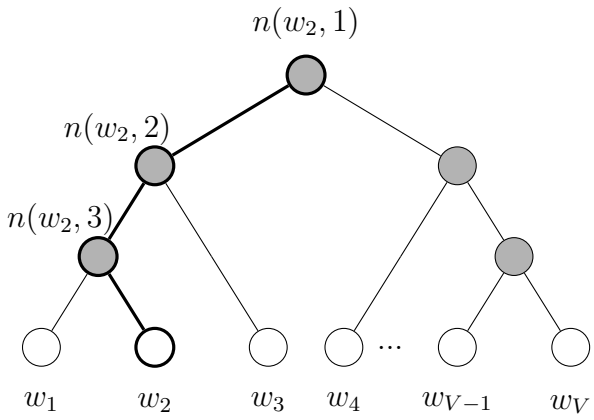
A definição de $\sigma(x)$ é dada na Equação 3.9. Na Equação 3.10, $O_{:,i}$ corresponde à coluna i da matriz de saída. Os valores de i são determinados por uma distribuição probabilística que seleciona com mais frequência palavras que ocorrem mais frequentemente no corpo de texto. A equação pode ser usada tanto para o CBOW como para o modelo Skip-gram. Para o Skip-gram, aplica-se esta equação para uma palavra de contexto de cada vez.

3.1.3 *Softmax* hierárquico

O *softmax* hierárquico é uma técnica que consiste em organizar as palavras do vocabulário em uma árvore binária balanceada. As palavras do vocabulário correspondem às folhas da árvore. Para cada folha, existe um caminho único a partir da raiz, e esse caminho é usado para calcular a probabilidade da palavra.

Seja $L(w)$ o número de nós no caminho da raiz até a folha w. Por exemplo, $L(w_2)$ na Figura 12 é 4. $n(w, i)$ será o i-ésimo nó neste caminho com vetor $\mathcal{O}_{:,n(w,i)}$ associado. Então $n(w, 1)$ é a raiz, enquanto $n(w, L(w))$ corresponde ao nó pai da palavra w. Para

Figura 12 – Árvore binária utilizada pelo *softmax* hierárquico.



Fonte: Autoria Própria.

cada nó interno n, é escolhido arbitrariamente um de seus filhos, chamado de $ch(n)$. Então, a probabilidade é calculada de acordo com a Equação 3.11.

$$p(w|w_i) = \prod_{j=1}^{L(w)-1} \sigma([n(w, j+1) = ch(n(w, j))] \cdot h \times \mathcal{O}_{:,n(w,j)}) \tag{3.11}$$

$$[x] = \begin{cases} 1, & \text{se } x \text{ é verdadeiro} \\ -1, & \text{caso contrário} \end{cases}$$

$$p(n, esquerda) = \sigma(h \times \mathcal{O}_{:,n(w,j)}) \tag{3.12}$$

$$\begin{aligned} p(n, direita) &= 1 - \sigma(h \times \mathcal{O}_{:,n(w,j)}) \\ &= \sigma(-h \times \mathcal{O}_{:,n(w,j)}) \end{aligned} \tag{3.13}$$

A definição de $\sigma(x)$ é mesma da Equação 3.9. A Equação 3.11 calcula o produto dos termos a partir da raiz $n(w, 1)$ até a folha w. Se for estabelecido que $ch(n)$ é sempre o nó esquerdo de n, então o termo $[n(w, j+1) = ch(n(w, j))]$ retorna 1 quando o caminho vai para a esquerda e -1 se direita. Além disso, o termo $[n(w, j+1) = ch(n(w, j))]$ fornece a normalização. Em um nó n, se forem somadas as probabilidades de percurso para o nó esquerdo e direito, é possível verificar que $\sigma(h \times \mathcal{O}_{:,n(w,j)}) + \sigma(-h \times \mathcal{O}_{:,n(w,j)}) = 1$. Essa normalização garante que a soma das probabilidades de todas as palavras é 1. A Equação 3.14 apresenta o cálculo de probabilidade da palavra w_2, dada uma palavra de entrada w_i.

$$p(w_2|w_i) = p(n(w_2, 1), left) \cdot p(n(w_2, 2), left) \cdot p(n(w_2, 3), right) \tag{3.14}$$

Ao se construir a árvore, é preciso armazenar qual é o caminho para cada palavra a partir da raiz, pois esse caminho será utilizado para o cálculo da probabilidade. Como a probabilidade de escolher o nó da esquerda ou da direita é calculada na medida que o caminho é traçado, se atribui um vetor para cada nó interno. Assim, diferentemente da função *softmax*, não há uma matriz com pesos que representam as palavras, mas sim uma matriz com dimensão $N \times (V-1)$ que representa todos $(V-1)$ nós da árvore balanceada, com exceção das folhas. O objetivo do treinamento é minimizar $-logP(w|w_i)$. Contudo, em vez de serem atualizados vetores de saída por palavra, são atualizados os vetores dos nós na árvore binária que estão no caminho do nó da raiz para a folha. A utilização de *softmax* hierárquico reduz a complexidade computacional por amostra apresentada à rede de $O(V)$ para $O(\log(V))$. Em termos de memória, ainda se tem aproximadamente o mesmo número de parâmetros ($V-1$ vetores, correspondendo aos nós internos da árvore). Na prática, o *softmax* hierárquico tende a ser mais eficiente para palavras menos frequentes, enquanto a amostragem negativa funciona melhor para palavras frequentes e vetores com dimensões menores.

3.2 Vetores de parágrafos

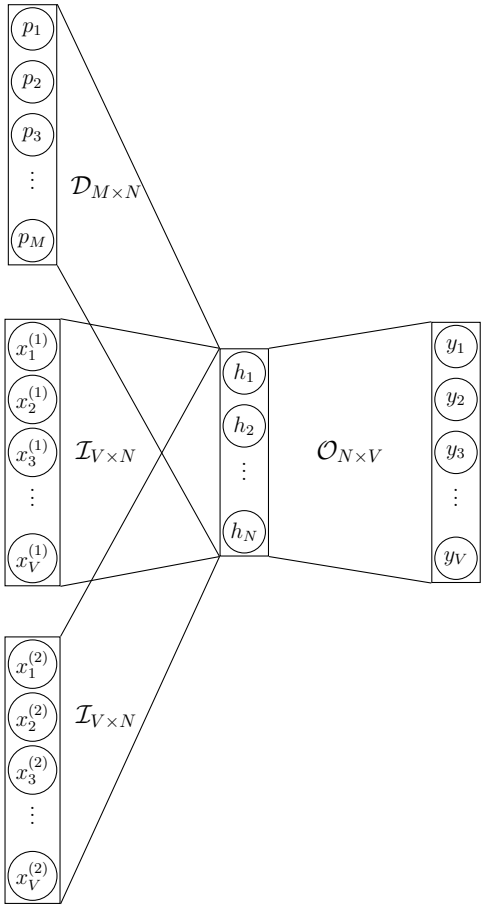
Algoritmos de aprendizado de máquina utilizados para classificação de texto, geralmente necessitam que a entrada seja um vetor de tamanho fixo. Para que seja possível utilizar os vetores de palavras gerados por CBOW ou Skip-gram, normalmente utiliza-se a média dos vetores das palavras contidas no texto. Essa abordagem tem o problema de não levar em consideração a ordem de ocorrência das palavras no texto. Uma abordagem mais sofisticada consiste em combinar os vetores de palavras em uma ordem dada por uma árvore sintática da sentença, usando operações vetoriais (SOCHER; LIN, 2011). Essa abordagem, entretanto, funciona apenas para frases sintaticamente corretas. O modelo de vetores de parágrafos (do inglês *Paragraph Vector* – PV) apresentado por Le e Mikolov (2014), é capaz de construir uma representação de tamanho fixo a partir de uma entrada com tamanho variável, levando em consideração a ordem das palavras sem depender de árvores sintáticas. Os autores apresentaram duas abordagens principais para treinar o modelo: modelo de memória distribuída e *bag of words* distribuído, conforme explicado a seguir.

3.2.1 Modelo de memória distribuída

O modelo de memória distribuída, abreviado como PV-DM (do inglês *Paragraph Vector – Distributed Memory*), é similar ao modelo CBOW, com três camadas: entrada, projeção e saída. A diferença é que nesse modelo, o parágrafo vai contribuir para a entrada de maneira similar como as palavras contribuem. Assim, além da matriz $\mathcal{I}$ que contém uma versão dos vetores de palavras, o modelo usa uma matriz $\mathcal{D}$ que contém um vetor

para cada parágrafo. A camada intermediária pode conter a média do vetor de parágrafo e os vetores de palavras, ou a concatenação desses vetores. A Figura 13 ilustra o caso em que a camada intermediária armazena a média dos vetores de entrada. Nos experimentos reportados por Le e Mikolov (2014), os autores usaram concatenação como método para combinar os vetores.

Figura 13 – Arquitetura da rede neural do método PV-DM.



Fonte: Autoria Própria.

A ativação da camada intermediária tem uma pequena diferença. No CBOW, a ativação é dada na Equação 3.1, enquanto que no PV-DM, o h é construído a partir de $\mathcal{I}$ e $\mathcal{D}$. No caso específico de se usar a média dos vetores, o h é calculado pela Equação 3.15. Na entrada, cada parágrafo é representado por um número (um *id*) que é transformado em um vetor *one-hot*. Assim, o *id* do parágrafo pode ser visto como sendo uma outra palavra. Ele atua como uma "memória" para lembrar o que falta no contexto atual, ou o tópico do parágrafo.

$$h = \frac{\mathcal{I}(x^{(1)} + x^{(2)} + \cdots + x^{(c)}) + (\mathcal{D} \times p)}{C + 1} \tag{3.15}$$

A quantidade de palavras no contexto, assim como no CBOW, é determinada pelo

tamanho de uma janela deslizante sobre o parágrafo. O vetor do parágrafo é compartilhado em todos os contextos gerados a partir do mesmo parágrafo. Para todos os parágrafos, a mesma matriz de palavras $\mathcal{I}$ é utilizada, ou seja, o vetor para uma determinada palavra é o mesmo para todos os parágrafos.

Ambos os vetores de palavras e os vetores de parágrafos são treinados com gradiente descendente estocástico e *backpropagation*. Posteriormente, os vetores de parágrafos podem ser usados como atributos que representam um parágrafo para entrada de algoritmos de aprendizado de máquinas. Se o modelo gerado for usado com um conjunto de dados diferente daquele que será usado em tarefas de classificação, é necessário inferir novos vetores para os parágrafos ainda não vistos. Nesta etapa, todo os parâmetros do modelo são fixados, e apenas os vetores de parágrafos são corrigidos utilizando gradiente descendente.

3.2.2 *Bag of words* distribuído

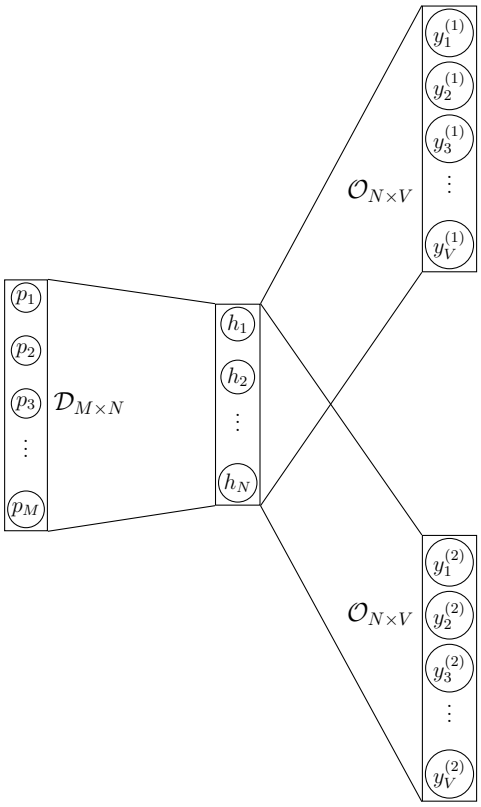
Diferentemente do método PV-DM, que considera o parágrafo e as palavras do contexto para prever a próxima palavra, o método *bag of words* distribuído, ou PV-DBOW, utiliza apenas o parágrafo como entrada. Como pode ser visto na Figura 14, esse modelo é similar ao Skip-gram. Ele também tem 3 camadas: entrada, projeção e saída. A entrada recebe o parágrafo, que é representado no padrão *one-hot*, assim como a palavra alvo no Skip-gram.

Existem duas possibilidades de implementação. Na primeira, são consideradas todas as palavras do parágrafo na saída, e o vetor de parágrafo é treinado para prever cada uma daquelas palavras. Na segunda possível implementação, que é descrita em Le e Mikolov (2014), uma janela do parágrafo é escolhida aleatoriamente, e palavras são amostradas aleatoriamente desta janela, para realizar o treinamento.

3.3 Glove

O uso de redes neurais em janelas de contexto local é uma alternativa à criação de matrizes globais de co-ocorrência e subsequente aplicação de operações vetoriais. Ambas as abordagens, entretanto, apresentam desvantagens. Mikolov, Yih e Zweig (2013) introduziram um esquema de avaliação de vetores de palavras baseado em analogias. Esse esquema examina a estrutura mais fina das palavras no espaço vetorial ao invés de avaliar a distância entre os vetores da palavras. Por exemplo, a analogia “o rei está para a rainha assim como o homem está para a mulher” deve ser codificada no espaço vetorial pela equação vetorial $rei - rainha = homem - mulher$. Este esquema de avaliação favorece os modelos que produzem mais dimensões de significado, capturando a idéia de multi-agrupamento das representações distribuídas (BENGIO, 2009).

Figura 14 – Arquitetura da rede neural do método PV-DBOW.



Fonte: Autoria Própria.

Os métodos que geram matrizes globais conseguem utilizar eficientemente informações estatísticas do corpo de texto como um todo, porém apresentam um desempenho relativamente ruim na tarefa de analogia descrita acima, o que indica que a estrutura do espaço vetorial gerado não é otimizada. Os métodos como o Skip-gram conseguem se sair melhor na tarefa de analogia, entretanto, utilizam mal as estatísticas do corpo de texto, porque são treinados utilizando janelas de contexto local separadas, ao invés das contagens de co-ocorrência globais, e não conseguem tirar proveito da grande quantidade de repetição nos dados.

O modelo GloVe proposto por Pennington, Socher e Manning (2014), tem como objetivo gerar vetores de palavras otimizados que tiram proveito das informações estatísticas do corpo de texto como um todo, sem ter uma etapa adicional de redução de dimensionalidade. O modelo assume que o ponto de partida apropriado para a aprendizagem de vetores de palavras deve ser as proporções entre as probabilidades de co-ocorrência ao invés das próprias probabilidades.

O modelo funciona da seguinte maneira. Antes da fase de treinamento, é construída uma matriz de co-ocorrência $\mathcal{X}$, sendo que $\mathcal{X}_{ij}$ representa a frequência com que a palavra i aparece no contexto da palavra j. O corpo de texto é percorrido apenas uma vez para construir essa matriz. A partir desse momento, essa matriz é utilizada para gerar os

vetores de palavras que são otimizados de maneira que cada par i e j atenda a restrição da Equação 3.16:

$$w_i^T \tilde{w}_j + b_i + \tilde{b}_j = \log(\mathcal{X}_{ij}), \tag{3.16}$$

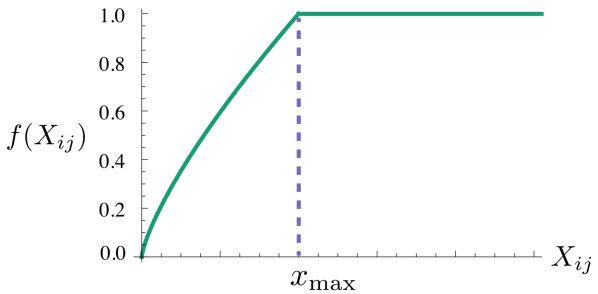
sendo que, w_i e $\tilde{w}_j$ são os vetores das palavras i e j, e b_i e $\tilde{b}_j$ são viézes associados às palavras i e j. Dessa maneira, o objetivo é construir vetores de palavras que conservem algumas informações úteis sobre como cada par de palavras i e j co-ocorrem, minimizando uma função de treinamento J, dada pela Equação 3.17.

$$J = \sum_{i=1}^{V} \sum_{j=1}^{V} f(\mathcal{X}_{ij})(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log(\mathcal{X}_{ij}))^2 \tag{3.17}$$

A função f é uma função de peso. Pennington, Socher e Manning (2014) escolheram para os seus experimentos a função apresentada na Equação 3.18, usando $xmax = 100$ e $\alpha = 3/4$, cujo gráfico pode ser observado na Figura 15.

$$f(x) = \begin{cases} (\frac{x}{x_{max}})^{\alpha} & \text{se } x < x_{max} \\ 1 & \text{caso contrário} \end{cases} \tag{3.18}$$

Figura 15 – Função de peso com $\alpha = 3/4$.



Fonte: Pennington, Socher e Manning (2014).

Segundo os autores, o modelo GloVe supera outros modelos na tarefas de analogia de palavras, semelhança de palavras e reconhecimento de entidades nomeadas.

3.4 Representação distribuída aplicada em análise de sentimento

Em muitos trabalhos da literatura, é comum encontrar o termo **Word2Vec** referindo-se de maneira coletiva aos algoritmos CBOW ou Skip-gram. O motivo é porque os autores desses métodos disponibilizaram publicamente uma biblioteca com esse mesmo nome,

contendo a implementação desses algoritmos[1](MIKOLOV et al., 2013b; MIKOLOV et al., 2013a). De maneira análoga, muitos artigos também se referem aos algoritmos para gerar vetores de parágrafos pelo nome **Doc2Vec**. Esses termos são utilizados na discussão que se segue, que apresenta alguns trabalhos que empregaram representação distribuída de texto em problemas de análise de sentimento.

Wang et al. (2014) combinaram o modelo de linguagem proposto por Bengio et al. (2003) com um método supervisionado para avaliar o sentimento de cada palavra. Dessa forma, o método proposto gera vetores que contém simultaneamente informação semântica e de polaridade das palavras. Eles usaram esses vetores em problemas de análise de sentimento em avaliações de produtos disponíveis *online*. Segundo os autores, os experimentos demonstraram que o método proposto supera os métodos tradicionais.

Rexha et al. (2016) apresentaram um algoritmo que prevê automaticamente a polaridade de frases-alvo em mensagens do Twitter. Os autores fornecem o seguinte exemplo com duas frases-alvo: *"New features @Microsoft suck. Check them back! #Linux solutions are awesome"*, sendo que o algoritmo retorna classificações distintas para **Microsoft** e **Linux**. Ele explora o uso de informações semânticas fornecidas por um modelo Word2Vec treinado em mensagens do Twitter, sem usar informações de polaridade sobre palavras isoladas. Para avaliar o algoritmo, foram usados os conjuntos do desafio SemEval 2016 Tarefa Nº4. Em seus experimentos, foram obtidas F-medida de 90% nas classes positivas e 54% nas negativas.

Giatsoglou et al. (2017) utilizaram algoritmos de aprendizado de máquina para treinar modelos de classificação de polaridade usando três abordagens de representação de vetores de textos: baseadas em dicionários de sentimento; baseadas no contexto de ocorrência das palavras; híbridas. O desempenho dessas representações na tarefa de classificação de sentimento foi avaliado por meio de experimentos em quatro conjuntos de dados contendo avaliações *online* escritos nos idiomas grego e inglês, que representam grupos de linguagem de inflexão alta e fraca, respectivamente.

A abordagem baseada em dicionários gera um vetor para cada palavra da amostra conforme descrito a seguir. Dado que o dicionário contém n palavras, cada uma pontuada para m emoções em alguma escala de classificação: utilizando-se *bag of words*, é gerado um vetor de dimensão n, indicando a ocorrência ou não de cada uma das n palavras na amostra; é gerado um vetor de tamanho m que representa a emoção média da amostra; os dois vetores são combinados em um único vector cujo tamanho é $n+m$. Na abordagem baseada no contexto de ocorrência das palavras, o Word2Vec é utilizado para gerar um vetor para cada palavra de cada amostra. A representação de uma dada sentença corresponde ao vetor da média dos vetores de todas as suas palavras, e a representação da amostra

[1] Word2Vec. Disponível em <https://code.google.com/archive/p/word2vec/>. Acessado em 28/10/2018

corresponde ao vetor da média dos vetores das sentenças. Essa abordagem mantém um tamanho fixo e relativamente pequeno para a representação das amostras. Na abordagem híbrida, um vetor é obtido através da concatenação dos vetores baseados em dicionários e os vetores de palavras.

Comparando os resultados das diferentes abordagens, foi observado que os vetores híbridos sempre resultam em uma melhoria significativa no desempenho em comparação com seus equivalentes baseados em dicionário ou aqueles gerados com o Word2Vec. Além disso, o uso de vetores híbridos não parece aumentar significativamente o tempo de execução necessário para o algoritmo de classificação.

Arslan, Küçük e Birturk (2018) conduziram uma série de experimentos para avaliar o desempenho do uso de vetores de palavras e de parágrafos gerados com Word2Vec e Doc2Vec, em problemas de análise de sentimento em mensagens do Twitter. Os resultados dos experimentos indicam que a filtragem de *emoticons* parece ser um fator de melhoria em conjuntos de dados limpos e anotados, e o uso de vetores de palavras gerados a partir de corpora maiores levam a precisões de análise de sentimento favoráveis.

Apesar de representações distribuídas de texto terem sido utilizadas com sucesso em problemas de análise de sentimento, pelo menos três questões importantes necessitam de mais investigação. A **primeira** é sobre qual o impacto no desempenho da classificação de textos curtos e ruidosos, causado pela escolha do corpus utilizado na geração do modelo. A **segunda** é se o uso desse tipo de representação pode proporcionar um desempenho equivalente ou superior ao uso de *bag of words* aliado a técnicas de normalização léxica e indexação semântica. Caso seja estabelecido que sim, então a **terceira** questão consiste em determinar qual dos diferentes modelos proporciona melhor desempenho. O capítulo seguinte descreve uma série de experimentos que foram realizados com intuito de responder a primeira questão. As outras duas questões foram investigadas por experimentos cujos resultados são apresentados no Capítulo 5.

4 Impacto do corpus na geração dos modelos de linguagem

Apesar dos avanços recentes e de vários estudos disponíveis na literatura, ainda não há evidências de que o corpus utilizado para gerar o modelo de representação possa proporcionar um impacto significativo no desempenho da aplicação. Embora muitos trabalhos na literatura empreguem corpus formal como a Wikipédia, sua estrutura textual pode não ser adequada o suficiente para tarefas que envolvem a classificação de textos curtos e ruidosos.

Este capítulo descreve uma série de experimentos que foram realizados para preencher essas lacunas. Para cada um dos três algoritmos escolhidos para geração de modelos de representação distribuída, foram criadas duas versões de modelo: uma a partir de um corpus de texto formal e a outra de um corpus de mensagens curtas e ruidosas. A hipótese testada é que usar modelos gerados com corpus com as mesmas características do domínio – mensagens curtas e ruidosas – é mais recomendado do que usar um corpus formal, como é tradicionalmente realizado na literatura. Para isso, foi usada como tarefa de *benchmark* a detecção de polaridade em mensagens curtas e ruidosas, cujos rótulos indicam se a mensagem expressa uma opinião positiva ou negativa. Cada versão dos modelos de linguagem foi avaliada na mesma tarefa, e os resultados comparados.

4.1 Protocolo experimental

Com intuito de tornar os experimentos reproduzíveis, são apresentados a seguir os detalhes sobre como os modelos de linguagem foram gerados, e quais conjuntos de dados e métodos de classificação foram utilizados.

4.1.1 Métodos de representação distribuída de texto

Os métodos utilizados para gerar representações distribuídas de texto avaliados nestes experimentos são os três mais populares em termos de citações na literatura. Eles são CBOW e Skip-gram (MIKOLOV et al., 2013a; MIKOLOV et al., 2013b), e GloVe (PENNINGTON; SOCHER; MANNING, 2014). Diferentes modelos foram gerados a partir do mesmo corpus, variando os valores dos parâmetros relevantes. Nos casos do CBOW e Skip-gram, foi utilizada a implementação Word2Vec da biblioteca Gensim[1] para linguagem

[1] Gensim. Disponível em <http://github.com/piskvorky/gensim>. Acessado em 26/01/2018

Python. No caso do GloVe, foi utilizada uma implementação do método disponibilizada pela Universidade de Stanford[2].

Como recomendado por Mikolov et al. (2013a), para CBOW e Skip-gram, o tamanho da janela foi avaliado com os valores 2 e 3, e o último obteve melhores resultados. O tamanho do vetor da palavra foi fixado em 300, e os parâmetros restantes foram definidos para os valores padrão da ferramenta Word2Vec. Como o GloVe e os métodos fornecidos pelo Word2Vec possuem parâmetros semelhantes, eles foram avaliados com os mesmos valores para oferecer uma comparação justa. Além disso, o número de iterações do GloVe foi definido empiricamente como 5, enquanto os demais parâmetros foram definidos para seus valores padrão. Como a tarefa de *benchmark* envolve a classificação de mensagens, é necessário decidir como representá-las a partir dos vetores de palavras individuais. Nos experimentos, cada mensagem foi representada pela média dos vetores das suas palavras, como é usualmente realizado na literatura.

4.1.2 Corpora

Os modelos de linguagem foram gerados a partir de dois grandes corpora com características distintas, ambos com aproximadamente 2 bilhões de palavras, descritos a seguir.

Wikipédia (Wiki). Composto de artigos bem escritos em inglês sobre vários assuntos extraídos de um *dump*[3] da enciclopédia virtual Wikipédia, obtida em 21 de abril de 2017. A classe WikiCorpus da biblioteca NLTK[4] foi usada para extrair sentenças dos artigos.

Twitter7 (T7). Yang e Leskovec (2011) coletaram 580 milhões de *tweets* que cobrem o período de junho de 2009 até janeiro de 2010. Parte desses *tweets*, contemplando um período de junho a dezembro de 2009, com 467 milhões de mensagens, foi disponibilizado *online*, em um corpus chamado Twitter7[5]. Deste corpus, para oferecer uma comparação justa, foram usados apenas os dados do mês de agosto para ter uma quantidade similar de palavras ao corpus da Wikipédia (cerca de 2 bilhões). Pontuações, nomes e menções de usuários foram removidos.

4.1.3 Conjuntos de dados

Os modelos de representação distribuída de texto gerados foram avaliados em tarefa de detecção de polaridade com dez conjuntos de dados reais, públicos e não-codificados. A Tabela 4 apresenta informações básicas sobre cada conjunto de dados.

[2] GloVe. Disponível em <http://nlp.stanford.edu/projects/glove>. Acessado em 26/01/2018
[3] Wikipedia *Dump*. Disponível em <https://dumps.wikimedia.org/>. Acessado em 26/01/2018.
[4] *Natural Language Toolkit*. Disponível em <http://www.nltk.org/>. Acessado em 26/01/2018.
[5] *Twitter7*. Disponível em <https://snap.stanford.edu/data/twitter7.html>. Acessado em 30/08/2018

Cada amostra corresponde a um *tweet* que expressa uma opinião, que é rotulada como positiva ou negativa.

Os conjuntos de dados STS-Test, HCR, OMD e SS-Tweet foram usados em trabalhos anteriores (AGARWAL et al., 2011; SPERIOSU et al., 2011; SHAMMA; KENNEDY; CHURCHILL, 2009; THELWALL; BUCKLEY; PALTOGLOU, 2012). Sanders[6] e UMICH[7] estão disponíveis publicamente, e os conjuntos de dados iPhone6, Archeage e Hobbit3 são compostos por mensagens coletadas do Twitter e rotuladas pelo grupo de pesquisa do Laboratório de Sistemas Inteligentes e Distribuídos (LaSID) da Universidade Federal de São Carlos, campus Sorocaba[8].

Tabela 4 – Conjuntos de dados utilizados para avaliar modelos de representação distribuída de texto na tarefa de detecção de polaridade.

Conjunto de dados	# Positivos	# Negativos	Tema
STS-Test	182	177	Geral
HCR	537	886	Medicina
OMD	710	1.196	Política
SS-Tweet	1.252	1.037	Geral
Sanders	519	572	Geral
UMICH	3.943	2.975	Geral
iPhone6	371	161	*Smartphones*
Archeage	724	994	Jogos
Hobbit3	354	168	Filmes
ALL	8.592	8.166	Geral

Os cinco primeiros conjuntos foram pré-processados da mesma maneira descrita por Saif et al. (2013). UMICH, iPhone6, Archeage e Hobbit3 foram usados como disponíveis no repositório original. O conjunto de dados ALL é composto pelas amostras de todos os conjuntos de dados mencionados.

Também foram coletadas informações sobre a distribuição das palavras por corpus e por conjunto de dados. Esses detalhes, assim como a quantidade de documentos ($\mathcal{D}$), a média de *tokens* por documento ($\mathcal{T}$) e a média de palavras em inglês por documento ($\mathcal{E}$), são apresentados na Tabela 5. Essas estatísticas destacam semelhanças entre as bases de dados do domínio e o corpus T7.

4.1.4 Métodos de classificação

Os seguintes métodos de classificação foram empregados: Bernoulli Naive Bayes (NB), máquinas de vetores de suporte (do inglês *support vector machines* ou SVM) e

[6] Sanders Analytics. Disponível em <http://www.sananalytics.com/lab/twitter-sentiment>. Acessado em 12/05/2016

[7] UMICH. Disponível em <https://inclass.kaggle.com/c/si650winter11>. Acessado em 12/05/2016

[8] SentCollection. Disponível em <http://www.dt.fee.unicamp.br/~tiago/sentcollection>. Acessado em 26/01/2018

Tabela 5 – Quantidade de amostras e distribuição de palavras em cada corpora e conjunto de dados.

Corpus	$\mathcal{D}$	$\mathcal{T}$	$\mathcal{E}$
Wiki	4.25 M	548	502
T7	131.53 M	14	12
Conjunto de Dados	$\mathcal{D}$	$\mathcal{T}$	$\mathcal{E}$
Archeage	1.718	16	11
HCR	1.423	18	8
Hobbit3	522	15	12
iPhone6	532	13	9
OMD	1.906	14	8
Sanders	1.091	16	8
SS-Tweet	2.289	16	9
STS-Test	359	15	8
UMICH	6.918	11	6
ALL	16.758	14	8

floresta aleatória (B.C4.5). Foram selecionadas abordagens com diferentes técnicas de representação e seleção de hipóteses: probabilidade, otimização e árvore. As implementações para esses métodos estão disponíveis na biblioteca Scikit-learn[9].

4.2 Resultados

Uma maneira de identificar se um modelo de representação é capaz de capturar características semânticas e sintáticas é analisar as palavras próximas entre si (MIKOLOV et al., 2013a). Isso é feito selecionando uma palavra arbitrariamente e calculando sua distância para todas as outras. Em seguida, as palavras mais próximas são selecionadas, e sua relação com a palavra original é analisada.

A Tabela 6 apresenta uma comparação de seis palavras escolhidas aleatoriamente, e as cinco palavras mais próximas a elas, nos diferentes modelos. Em geral, todos os modelos capturaram significado semântico similar para as palavras selecionadas. Por exemplo, nomes de países e cidades foram encontrados próximos à palavra *"england"*, e palavras relacionadas ao espaço foram encontradas próximas à palavra *"universe"*. Nesse caso, o GloVe também encontrou *"competition"* provavelmente devido à existência de textos sobre a competição de *Miss* Universo.

Ao comparar palavras usando modelos treinados com corpora de domínio distinto, grandes diferenças aparecem. Por exemplo, com modelos gerados a partir do Twitter, as palavras *"dude"*, *"guy"* e até mesmo *"shit"* estão próximas à palavra *"man"*, além de palavras com erros ortográficos como *"mannnn"* e *"maaan"*. Próximas à *"graffitti"* estão

[9] Scikit-learn. Disponível em <http://scikit-learn.org/>. Acessado em 28/01/2018

Tabela 6 – Comparação de palavras similares em diferentes modelos de representação distribuída de textos treinados com corpora distintos.

	CBOW		**SG**		**GloVe**	
Palavra-alvo	**T7**	**Wiki**	**T7**	**Wiki**	**T7**	**Wiki**
school	college	college	skewl	college	student	academy
	skewl	seminary	highscool	elementary	orientation	education
	highschool	academy	preschool	preparatory	class	grades
	uni	kindergarten	orientation	kindergarten	college	attended
	class	university	kindergarten	graduating	elementary	elementary
man	woman	woman	woman	woman	shit	woman
	dude	girl	dude	boy	boy	dead
	guy	boy	mannnn	girl	woman	my
	girl	villager	maaan	person	dude	appears
	boy	stranger	guy	stranger	was	hero
england	wales	scotland	wales	scotland	ireland	manchester
	scotland	wales	scotland	wales	united	britain
	zealand	hampshire	australia	britain	ashes	yorkshire
	australia	ireland	zealandy	ireland	aussies	wales
	ireland	britain	newcastle	lancashire	australia	sheffield
universe	world	multiverse	pageant	multiverse	crowned	marvel
	cosmos	worlds	contestants	earth	universla	earth
	existence	marvel	cosmos	pageant	nature	cosmic
	realm	earth	heidi	cosmic	heide	planet
	darkness	cosmos	universo	miss	competition	comics
update	upgrade	updated	updated	updated	updated	android
	updating	upgrade	updating	fixes	latest	user
	updated	app	upgrade	updating	post	fixes
	post	refresh	status	ios	news	updated
	install	firmware	report	firmware	page	firmware
graffiti	#graffiti	stencil	#streetart	collage	mural	slogans
	#streetart	posters	mural	stencil	watercolor	res
	typography	collage	stencil	posters	illustrations	collage
	mural	billboards	#graffiti	billboards	typography	artists
	#art	murals	art	vandalism	mosaic	subculture

as palavras *"#streetart"*, *"#art"* e *"typography"*. Esta é uma indicação clara de que treinar o modelo de representação usando corpus do mesmo domínio da aplicação pode promover um desempenho melhor na tarefa final (*e.g.*, classificação, agrupamento etc).

4.2.1 Resultados na tarefa de análise de sentimento

Os resultados dos experimentos na aplicação são apresentados na Tabela 7. A medida de desempenho é a tradicional F-medida obtida em uma validação cruzada de 5 partições. Os valores em negrito indicam os melhores resultados encontrados em cada conjunto de dados para cada classificador, enquanto os melhores resultados entre todos os experimentos para um conjunto de dados específico são indicados por $(*)$.

Considerando apenas os melhores resultados obtidos em cada conjunto de dados (valores destacados com $*$), é possível observar que em todos os conjuntos de dados o melhor desempenho geral foi atingido quando o modelo de representação distribuída foi treinado com o corpus T7. Em apenas um conjunto de dados (Hobbit), a melhor F-medida foi obtida usando um modelo de representação distribuída gerada com Wiki.

Em alguns casos, a melhoria na F-medida é significativamente maior quando o modelo de representação foi gerado com T7. Por exemplo, no conjunto de dados STS-

Tabela 7 – F-medida obtida por cada método na tarefa de análise de sentimento.

		NB		**SVM**		**B.C4.5**	
Dataset	**Modelo**	**T7**	**Wiki**	**T7**	**Wiki**	**T7**	**Wiki**
ALL	CBOW	0,74	0,68	**0,85***	0,82	0,78	0,76
	SG	**0,76**	0,66	**0,85***	0,83	0,79	0,76
	GloVe	0,75	0,68	0,84	0,82	**0,80**	0,76
Archeage	CBOW	**0,75**	0,72	**0,80***	0,78	0,75	0,74
	SG	**0,75**	0,72	**0,80***	0,78	0,75	0,72
	GloVe	0,73	0,67	**0,80***	0,78	**0,78**	0,73
HCR	CBOW	0,65	0,62	0,69	0,59	0,58	0,54
	SG	0,65	0,61	**0,71***	0,63	0,59	0,52
	GloVe	**0,66**	0,62	0,70	0,63	**0,60**	0,58
Hobbit	CBOW	0,73	0,75	0,65	0,65	0,71	0,71
	SG	**0,80***	0,79	0,67	0,67	**0,77**	0,76
	GloVe	0,76	**0,80***	0,70	**0,75**	0,76	0,71
iPhone6	CBOW	**0,76***	0,72	**0,66**	0,65	**0,74**	0,69
	SG	0,72	0,74	0,61	0,65	0,70	0,65
	GloVe	0,67	0,67	0,64	0,65	0,66	0,69
OMD	CBOW	**0,65**	0,63	0,69	0,67	0,65	0,64
	SG	0,64	0,64	0,70	0,67	0,64	0,62
	GloVe	0,63	0,62	**0,71***	0,67	**0,67**	0,62
Sanders	CBOW	**0,68**	0,65	0,74	0,69	0,68	0,63
	SG	0,67	0,63	0,74	0,69	0,69	0,64
	GloVe	0,65	0,62	**0,75***	0,70	**0,71**	0,65
SS-Tweet	CBOW	0,65	0,63	0,74	0,70	0,61	0,56
	SG	**0,67**	0,61	0,75	0,71	0,63	0,56
	GloVe	**0,67**	0,63	**0,76***	0,70	**0,65**	0,58
STS-Test	CBOW	**0,79**	0,69	0,80	0,70	0,74	0,61
	SG	**0,79**	0,71	**0,84***	0,67	0,70	0,62
	GloVe	0,77	0,69	0,82	0,69	**0,78**	0,61
UMICH	CBOW	**0,94**	0,90	0,98	0,98	0,96	0,96
	SG	0,93	**0,94**	**0,99***	0,98	**0,97**	0,96
	GloVe	0,92	0,88	0,98	0,98	**0,97**	0,96

Test, o SVM + SG atingiu F-medida de 0,84, resultado cerca de 25% maior do que os 0,67 obtidos pela mesma combinação quando o modelo foi gerado com Wiki. As mesmas diferenças altas podem ser vistas no HCR (SVM + SG + T7 = 0,71 *vs* SVM + SG + Wiki = 0,63) e SS-Tweet (SVM + GloVe + T7 = 0,76 *vs* SVM + GloVe + Wiki = 0,70).

Comparando todos os noventa pares de resultados obtidos por cada método de classificação e modelo de representação gerados com os corpora T7 e Wiki, os modelos treinados com T7 superaram significativamente os modelos baseados em Wiki (74 testes - 82,2%). Os modelos baseados em Wiki só tiveram desempenho melhor do que os baseados em T7 em 8 testes (8,9%), e obtiveram os mesmos resultados para outros 8 testes (8,9 %).

Os resultados obtidos confirmam a hipótese de que treinar modelos de representação distribuída com um corpus composto por texto com as mesmas características do domínio de aplicação proporciona um desempenho melhor. Analisando os melhores resultados obtidos em todos os experimentos, o classificador NB encontrou o melhor de-

sempenho apenas em 2 conjuntos de dados. O SVM ofereceu a hipótese mais geral para diferentes contextos, apresentando o melhor desempenho na maioria dos conjuntos de dados utilizados (8 de 10 conjuntos de dados). No entanto, para todos os classificadores avaliados, a melhoria fornecida pelos modelos treinados com o T7 foi consistente em todos os conjuntos de dados. Nenhum deles foi mais (ou menos) impactado pela escolha do modelo de representação.

4.3 Considerações finais

Os resultados encontrados nos experimentos fornecem fundamentação empírica de que, para detectar polaridade em mensagens curtas e ruidosas, deve-se usar modelos de linguagem gerados a partir de corpus contendo mensagens com essas mesmas características (sintáticas e semânticas) do domínio. Esses experimentos constituem uma parte importante deste trabalho e mais detalhes podem ser obtidos em Lochter et al. (2018).

5 Avaliação da representação distribuída de texto em análise de sentimento

O capítulo anterior evidenciou as vantagens de usar modelos distribuídos de linguagem treinados a partir de mensagens com características do domínio da aplicação. Neste capítulo, são apresentados os resultados dos experimentos que avaliam se o uso da representação distribuída pode proporcionar um desempenho equivalente ou superior ao da *bag of words* aliada à técnicas de normalização léxica e indexação semântica, aplicadas na tarefa de análise de sentimento de mensagens curtas e ruidosas.

5.1 Geração dos modelos de linguagem

Para gerar os modelos de linguagem, foi utilizado o Twitter7, o mesmo corpus de mensagens do Twitter utilizado nos experimentos descritos no Capítulo 4. A partir deste corpus, várias etapas foram realizadas para chegar em um arquivo final, utilizado para gerar os vetores de palavras e parágrafos. Inicialmente, foram excluídos os *tweets* vazios e os *re-tweets*. Como o corpus original contém *tweets* de várias línguas, e os conjuntos de dados que foram utilizados para análise de sentimento são todos em inglês, foi realizado um processo de seleção para extrair do corpo de texto apenas os *tweets* em inglês. Para isso foi utilizado FastText (JOULIN et al., 2016a; JOULIN et al., 2016b) com um modelo de detecção de idiomas pré-treinado. FastText é uma biblioteca de código aberto que permite aos usuários aprender representações e classificadores de texto. O modelo[1] utilizado pode reconhecer 176 idiomas e é treinado com Wikipédia, Tatoeba e SETimes. Foi utilizada a versão não comprimida que, segundo o *website* da ferramenta, é mais rápida e mais precisa. Após terem sido extraídos apenas os *tweets* em inglês, cada mensagem foi pré-processada de acordo com as seguintes regras:

- todas as palavras foram convertidas para letras minúsculas;
- e-mails foram substituídos pelo *token* `<email>`;
- endereços de internet foram substituídos por `<url>`;
- os usuários do Twitter, cujos nomes começam com "@", foram substituídos por `<user>`;

[1] FastText. Disponível em <https://fasttext.cc/docs/en/language-identification.html>. Acessado em 30/09/2018

- *smileys* foram substituídos pelos tokens `<smile>`, `<lolface>`, `<sadface>`, `<neutralface>` de maneira correspondente;
- "<3" foi substituído por `<heart>`;
- números foram substituídos por `<number>`;
- o símbolo "#" de uma *hashtag* foi substituído por `<hashtag>`, mantendo a palavra da *hashtag*. Por exemplo "#iphone" foi convertido em "<hashtag> iphone";
- palavras com letras repetidas no final foram normalizadas. Por exemplo, "noooooo" é convertido em "no";
- múltiplos espaços foram convertidos para um simples espaço; e
- caracteres especiais foram excluídos.

Como resultado deste processo, foi gerado um corpus de **28 Gigabytes**, contendo **364.284.225 *tweets***. Treinar as redes neurais artificiais para gerar modelos de linguagem distribuídos usando todas as amostras do corpus se mostrou inviável tanto devido ao tempo de processamento necessário quanto ao tamanho dos arquivos que seriam gerados. Consequentemente, dois subconjuntos de mensagens desse corpus, escolhidas aleatoriamente, foram utilizados: um com 25 milhões de *tweets* e outro com 100 milhões.

Para cada subconjunto utilizado, foram gerados os modelos de linguagem com os métodos CBOW, Skip-gram, Paragraph Vector (PV) e GloVe. Para gerar os modelos com CBOW e Skip-gram, foi utilizada a implementação do FastText[2] disponibilizada pelo Facebook. Para o Paragraph Vector, foi utilizada a implementação Doc2Vec da biblioteca Gensim[3] para linguagem Python. Para cada subconjunto, foi gerado um modelo para PV-DBOW e no caso do PV-DM, foi gerado um modelo com cada uma da opções de função da camada oculta – concatenação, soma e média – totalizando 4 modelos diferentes. No caso do GloVe[4], foi utilizada uma implementação do método disponibilizada pela Universidade de Stanford.

Como é importante que os diferentes modelos tenham os mesmos parâmetros para que os resultados possam ser comparados entre si, todos os modelos foram gerados com vetores de tamanho 200, janela de tamanho 4, número mínimo de ocorrência para uma palavra entrar no vocabulário igual a 5 e amostragem igual a 0,001. O tamanho dos vetores foi determinado de maneira empírica após experimentações com diferentes valores. Valores acima de 200 resultariam em modelos muito grandes no caso do Paragraph Vector. No caso do tamanho da janela, Pennington, Socher e Manning (2014) realizaram experimentos

[2] FastText. Disponível em<https://github.com/facebookresearch/fastText/>. Acessado 30/09/2018
[3] Gensim. Disponível em <http://github.com/piskvorky/gensim>. Acessado 4/5/2016
[4] GloVe. Disponível em <http://nlp.stanford.edu/projects/glove>. Acessado 4/5/2016

com valores de 2 até 10. Por ser inviável testar todos os valores nesta faixa, foi decidido utilizar 4 como um palpite guiado pelo fato de *tweets* serem menores do que as amostras utilizadas pelos autores. Para os outros parâmetros foram usados os valores padrões dos métodos. Os modelos gerados a partir de 25 milhões de *tweets* foram treinados com 20 épocas, enquanto que os gerados a partir de 100 milhões foram treinados usando 40 épocas. O número de épocas foi determinado levando em consideração o tempo necessário para gerar modelos usando mais épocas, e o consequente ganho em desempenho em utilizar esses modelos. Usar mais de 40 épocas se mostrou inviável devido ao tempo e desnecessário levando em consideração o aumento relativamente pequeno no desempenho.

5.2 Experimentos

Os experimentos foram planejados para que os resultados pudessem ser comparados com os resultados obtidos por Lochter (2015) e Lochter et al. (2016). Por esse motivo, foram utilizados os mesmos conjuntos de dados, listados na Tabela 8.

Tabela 8 – Conjuntos de dados utilizados.

Conjunto	**# Positivas**	**# Negativas**	**Tema**
Archeage	724	994	Jogo
HCR	537	886	Medicina
Hobbit	354	168	Filme
iPhone6	371	161	Smartphone
OMD	710	1196	Política
Sanders	519	572	Geral
SS-Tweet	1252	1037	Geral
STS-Test	182	177	Geral
UMICH	3943	2975	Filme

Os conjuntos de dados STS-Test, HCR, OMD e SS-Tweet foram usados em diversos trabalhos anteriores (AGARWAL et al., 2011; SPERIOSU et al., 2011; SHAMMA; KENNEDY; CHURCHILL, 2009; THELWALL; BUCKLEY; PALTOGLOU, 2012). Sanders[5] e UMICH[6] estão disponíveis publicamente e os conjuntos de dados iPhone6, Archeage e Hobbit3 são compostos por mensagens coletadas do Twitter e rotuladas pelo grupo de pesquisa do Laboratório de Sistemas Inteligentes e Distribuídos (LaSID) da Universidade Federal de São Carlos, campus Sorocaba[7]. Cada conjunto de dados foi pré-processado com as mesmas regras descritas na seção anterior.

[5] Sanders Analytics. Disponível em <http://www.sananalytics.com/lab/twitter-sentiment>. Acessado em 12/05/2016

[6] UMICH. Disponível em <https://inclass.kaggle.com/c/si650winter11>. Acessado em 12/05/2016

[7] *SentCollection*. Disponível em <http://www.dt.fee.unicamp.br/~tiago//sentcollection>. Acessado em 26/01/2018

Para cada um dos dois conjuntos de modelos de linguagem – gerados a partir de 25 e 100 milhões de amostras – foram realizadas duas etapas: seleção de modelo e classificação. Em ambas, foram utilizadas as bibliotecas Scikit-learn (PEDREGOSA et al., 2012) e Pandas (MCKINNEY, 2010) para linguagem de programação Python.

Os métodos de classificação utilizados são:

- Naive Bayes Bernoulli (NB-B) (MCCALLUM; NIGAM, 1998);
- Naive Bayes Gaussiano (NB-G) (MCCALLUM; NIGAM, 1998);
- Boosted C4.5 (B.C4.5) (SCHAPIRE, 1999);
- Árvores de decisão (C4.5) (BREIMAN et al., 1984);
- k-vizinhos mais próximos (k-NN) (SALTON; MCGILL, 1983);
- Regressão logística (RL) (WALKER; DUNCAN, 1967);
- Máquinas de vetores de suporte com kernel linear (SVM-L) (CORTES; VAPNIK, 1995);
- Máquinas de vetores de suporte com kernel de função de base radial (SVM-R) (BOSER; GUYON; VAPNIK, 1992); e
- Comitê de classificadores (XIA; ZONG; LI, 2011).

Os métodos de classificação avaliados são exatamente os mesmos utilizados por Lochter et al. (2016), com exceção do Naive Bayes Multinomial, que não aceita valores negativos e do Comitê, que foi utilizada a classe *VotingClassifier* do Scikit-learn com estratégia de voto *soft*, que é muito similar a técnica usado pelos autores.

Na etapa de seleção do modelo, foi utilizada busca em grade para encontrar os melhores parâmetros para cada combinação método de classificação × modelo de linguagem. Para isso, foi utilizada validação cruzada com 5 partições. A Tabela 9 apresenta a lista de parâmetros e os intervalos de valores que foram analisados. Após determinados os melhores valores, foi realizada a etapa de classificação. Os demais parâmetros de cada método foram mantidos com seus valores padrões.

Para cada combinação conjunto de dados × método de classificação × modelo de linguagem, a classificação foi feita utilizando validação cruzada com 5 partições e computados a acurácia, precisão, revocação e F-medida. As Tabelas 10 e 11 apresentam os parâmetros encontrados pelo método de busca em grade aplicada para cada modelo de linguagem. Esses foram os parâmetros utilizados em cada experimento.

Ao todo, foram testadas **1.134 combinações**, sendo que: 7 modelos de linguagem (CBOW, Skip-Gram, GloVe, PVDBOW, PVDM-Concat, PVDM-Mean, PVDM-Sum) que foram gerados a partir de 2 subconjuntos da Twitter7 (25 milhões e 100 milhões de amostras). Cada um desses 14 possíveis modelos foi usado para representar computacionalmente cada um dos 9 conjuntos dados na tarefa de análise de sentimento (Tabela 8). Cada base de dados foi usada para treinar e testar cada um dos 9 métodos de classificação empregados.

Tabela 9 – Parâmetros dos métodos otimizados por *grid search*.

Métodos	**Parâmetro**	**Valores**
B.C4.5	*criterion*	gini, entropy
B.C4.5	*max_features*	*sqrt*, log_2
B.C4.5	*n_estimators*	$50, 100, 250, 500$
C4.5	*criterion*	gini, entropy
C4.5	*max_features*	sqrt, log
NB-B	*alpha*	$10^{-7}...10^{7}$, passo 1 na potência
NB-B	*binarize*	$10^{-7}...10^{7}$, passo 1 na potência
RL	*C*	$10^{-3}...10^{3}$, passo 1 na potência
SVM-L	*C*	$10^{-3}...10^{3}$, passo 1 na potência
SVM-R	*C*	$10^{-3}...10^{3}$, passo 1 na potência
SVM-R	*gama*	$10^{-3}...10^{3}$, passo 1 na potência
k-NN	*n_neighbors*	1...20, passo 1
k-NN	*weights*	uniform, distance

Tabela 10 – Parâmetros encontrados na busca em grade usando modelos de linguagem gerados com 25 milhões de amostras.

Método	Parâmetro	CBOW	GloVe	PVDBOW	PVDM Concat	PVDM Mean	PVDM Sum	Skip-Gram
B.C4.5	*criterion*	entropy	gini	gini	entropy	gini	entropy	gini
B.C4.5	*max_features*	sqrt	sqrt	sqrt	sqrt	sqrt	sqrt	sqrt
B.C4.5	*n_estimators*	50	100	500	50	50	50	50
C4.5	*criterion*	gini	entropy	gini	entropy	entropy	gini	entropy
C4.5	*max_features*	sqrt	sqrt	sqrt	log2	log2	log2	sqrt
NB-B	*alpha*	0	10000	100000	0	100000	100000	10000
NB-B	*binarize*	10	0,001	0,001	10	0,01	0,001	0,01
RL	*C*	0,1	0,01	0,001	0,001	100	0,001	0,1
SVM-L	*C*	0,1	1	100	100	1	0,01	0,01
SVM-R	*C*	100	100	0,001	0,1	10	100	10
SVM-R	*gamma*	0,001	0,01	0,001	0,001	0,01	0,01	0,1
k-NN	*n_neighbors*	19	13	18	15	15	19	19
k-NN	*weights*	distance	distance	distance	distance	distance	uniform	distance

5.3 Resultados

As Tabelas 12 e 13 sumarizam qual modelo de linguagem, treinado com 25 milhões e 100 milhões de amostras respectivamente, obteve a maior F-medida para cada combina-

Tabela 11 – Parâmetros encontrados na busca em grade usando modelos de linguagem gerados com 100 milhões de amostras.

Método	Parâmetro	CBOW	GloVe	PVDBOW	PVDM Concat	PVDM Mean	PVDM Sum	Skip-Gram
B.C4.5	*criterion*	gini	entropy	entropy	entropy	entropy	entropy	entropy
B.C4.5	*max_features*	sqrt	sqrt	sqrt	sqrt	sqrt	sqrt	sqrt
B.C4.5	*n_estimators*	500	50	50	50	50	50	50
C4.5	*criterion*	entropy	entropy	gini	entropy	gini	entropy	entropy
C4.5	*max_features*	sqrt	sqrt	sqrt	sqrt	log2	sqrt	sqrt
NB-B	*alpha*	100000	10000	10000	100	100000	100000	100000
NB-B	*binarize*	1	0,1	0,01	1	0,001	0,01	0,01
RL	*C*	1	10	0,001	0,01	100	0,001	10
SVM-L	*C*	0,1	1	0,001	0,001	1	0,01	10
SVM-R	*C*	100	10	0,1	0,001	10	0,001	10
SVM-R	*gamma*	0,001	0,01	0,001	0,001	0,01	0,001	0,1
k-NN	*n_neighbors*	7	19	19	10	6	18	19
k-NN	*weights*	distance	distance	uniform	distance	distance	distance	distance

Tabela 12 – Modelo de linguagem, treinado com 25 milhões de amostras, que obteve a maior F-medida para cada combinação conjunto de dados × método de classificação.

Método	**Archeage**	**HCR**	**Hobbit**	**iPhone6**	**OMD**	**Sanders**	**SS-Tweet**	**STS-Test**	**UMICH**
NB-B	CBOW 0,579	CBOW 0,623	CBOW 0,678	CBOW 0,697	CBOW 0,627	Skip-Gram 0,529	CBOW 0,547	GloVe 0,599	PVDM-Mean 0,570
B.C4.5	Skip-Gram 0,732	Skip-Gram 0,628	Skip-Gram 0,720	Skip-Gram 0,709	Skip-Gram 0,634	Skip-Gram 0,645	GloVe 0,615	Skip-Gram 0,696	CBOW 0,952
C4.5	Skip-Gram 0,747	GloVe 0,619	GloVe 0,709	Skip-Gram 0,714	Skip-Gram 0,652	Skip-Gram 0,668	GloVe 0,616	CBOW 0,641	GloVe 0,959
Comitê	CBOW 0,831	Skip-Gram 0,731	GloVe 0,864	Skip-Gram 0,803	CBOW 0,761	CBOW 0,779	Skip-Gram 0,751	Skip-Gram 0,855	CBOW 0,982
NB-G	Skip-Gram 0,746	Skip-Gram 0,630	Skip-Gram 0,787	Skip-Gram 0,712	GloVe 0,685	Skip-Gram 0,685	Skip-Gram 0,704	Skip-Gram 0,875	GloVe 0,856
k-NN	Skip-Gram 0,836	Skip-Gram 0,732	GloVe 0,780	Skip-Gram 0,784	Skip-Gram 0,748	Skip-Gram 0,763	Skip-Gram 0,734	Skip-Gram 0,855	Skip-Gram 0,966
RL	CBOW 0,853	CBOW 0,741	CBOW 0,868	**CBOW 0,808**	CBOW 0,772	CBOW 0,809	CBOW 0,783	**Skip-Gram 0,894**	CBOW 0,982
SVM-L	CBOW 0,856	CBOW 0,760	GloVe 0,908	CBOW 0,801	CBOW 0,779	GloVe 0,818	CBOW 0,782	CBOW 0,861	GloVe 0,991
SVM-R	**Skip-Gram 0,879**	**Skip-Gram 0,772**	**GloVe 0,914**	CBOW 0,803	**Skip-Gram 0,802**	**Skip-Gram 0,839**	**Skip-Gram 0,795**	Skip-Gram 0,886	**Skip-Gram 0,991**

ção conjunto de dados × método de classificação. Todos os resultados dos experimentos estão disponíveis no Apêndice A. Além disso, a Tabela 14 inclui os resultados obtidos por Lochter (2015) para facilitar a comparação. Valores em negrito indicam o melhor resultado obtido em cada base de dados. As colunas das tabelas (cada conjunto de dados) são coloridas como mapas de calor, que vão do azul (menor F-medida) ao vermelho (maior F-medida).

Comparando somente os experimentos com representação distribuída (Tabelas 12 e 13) verifica-se que o método SVM-R obteve melhor F-medida na maioria dos experimentos, tanto usando modelos gerados com 25 milhões quanto com 100 milhões de amostras, e que o Skip-gram obteve melhor F-medida na maioria dos experimentos.

Um fato que chama a atenção é que os experimentos realizados com o Comitê de classificadores não obtiveram a melhor F-medida em nenhum dos conjuntos de dados,

Tabela 13 – Modelo de linguagem, treinado com 100 milhões de amostras, que obteve a maior F-medida para cada combinação conjunto de dados × método de classificação.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	CBOW 0,579	CBOW 0,623	CBOW 0,678	CBOW 0,697	CBOW 0,627	GloVe 0,541	CBOW 0,547	GloVe 0,577	PVDBOW 0,571
B.C4.5	GloVe 0,740	Skip-Gram 0,668	Skip-Gram 0,709	CBOW 0,699	Skip-Gram 0,635	Skip-Gram 0,664	GloVe 0,622	Skip-Gram 0,705	Skip-Gram 0,953
C4.5	Skip-Gram 0,746	GloVe 0,617	GloVe 0,724	CBOW 0,680	Skip-Gram 0,630	Skip-Gram 0,659	GloVe 0,616	Skip-Gram 0,677	Skip-Gram 0,957
Comitê	Skip-Gram 0,855	Skip-Gram 0,744	GloVe 0,875	Skip-Gram 0,821	Skip-Gram 0,765	Skip-Gram 0,808	Skip-Gram 0,779	Skip-Gram 0,894	Skip-Gram 0,988
NB-G	Skip-Gram 0,759	Skip-Gram 0,630	Skip-Gram 0,785	Skip-Gram 0,703	Skip-Gram 0,676	Skip-Gram 0,677	Skip-Gram 0,697	Skip-Gram 0,864	Skip-Gram 0,869
k-NN	Skip-Gram 0,827	Skip-Gram 0,729	Skip-Gram 0,759	Skip-Gram 0,789	Skip-Gram 0,748	Skip-Gram 0,774	Skip-Gram 0,738	Skip-Gram 0,844	Skip-Gram 0,966
RL	Skip-Gram 0,870	Skip-Gram 0,753	**GloVe 0,908**	Skip-Gram 0,818	Skip-Gram 0,794	**Skip-Gram 0,836**	Skip-Gram 0,798	Skip-Gram 0,894	Skip-Gram 0,991
SVM-L	GloVe 0,864	CBOW 0,753	Skip-Gram 0,906	Skip-Gram 0,806	Skip-Gram 0,788	Skip-Gram 0,822	Skip-Gram 0,789	Skip-Gram 0,883	CBOW 0,990
SVM-R	**Skip-Gram 0,873**	**Skip-Gram 0,773**	Skip-Gram 0,904	**Skip-Gram 0,823**	**CBOW 0,797**	Skip-Gram 0,836	**Skip-Gram 0,800**	**Skip-Gram 0,900**	**Skip-Gram 0,992**

Tabela 14 – Comparação da F-medida dos experimentos realizados por Lochter (2015) usando amostras expandidas sem aplicar seleção de atributos, com os experimentos realizados na pesquisa utilizando modelos de linguagem gerados com 100 milhões de *tweets* e 40 épocas de treinamento.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	SentMiner 0,860	SentMiner 0,716	SentMiner 0,862	CBOW 0,697	SentMiner 0,787	SentMiner 0,688	CBOW 0,547	SentMiner 0,821	SentMiner 0,951
B.C4.5	SentMiner 0,758	Skip-Gram 0,668	SentMiner 0,825	CBOW 0,699	SentMiner 0,703	Skip-Gram 0,664	GloVe 0,622	Skip-Gram 0,705	Skip-Gram 0,953
C4.5	SentMiner 0,746	SentMiner 0,627	SentMiner 0,838	SentMiner 0,687	SentMiner 0,707	Skip-Gram 0,659	GloVe 0,616	SentMiner 0,736	Skip-Gram 0,957
Comitê	SentMiner 0,871	SentMiner 0,746	**SentMiner 0,924**	Skip-Gram 0,821	**SentMiner 0,826**	Skip-Gram 0,808	Skip-Gram 0,779	Skip-Gram 0,894	Skip-Gram 0,988
NB-G	SentMiner 0,798	Skip-Gram 0,630	Skip-Gram 0,785	Skip-Gram 0,703	SentMiner 0,684	Skip-Gram 0,677	Skip-Gram 0,697	Skip-Gram 0,864	Skip-Gram 0,869
k-NN	Skip-Gram 0,827	Skip-Gram 0,729	SentMiner 0,828	Skip-Gram 0,789	Skip-Gram 0,748	Skip-Gram 0,774	Skip-Gram 0,738	Skip-Gram 0,844	Skip-Gram 0,966
RL	Skip-Gram 0,870	Skip-Gram 0,753	SentMiner 0,914	Skip-Gram 0,818	SentMiner 0,806	**Skip-Gram 0,836**	Skip-Gram 0,798	Skip-Gram 0,894	Skip-Gram 0,991
NB-M	SentMiner 0,848	SentMiner 0,705	SentMiner 0,862	SentMiner 0,696	SentMiner 0,790	SentMiner 0,710	SentMiner 0,545	SentMiner 0,833	SentMiner 0,941
SVM-L	GloVe 0,864	CBOW 0,753	SentMiner 0,913	Skip-Gram 0,806	SentMiner 0,802	Skip-Gram 0,822	Skip-Gram 0,789	Skip-Gram 0,883	CBOW 0,990
SVM-R	**Skip-Gram 0,873**	**Skip-Gram 0,773**	SentMiner 0,909	**Skip-Gram 0,823**	CBOW 0,797	Skip-Gram 0,836	**Skip-Gram 0,800**	**Skip-Gram 0,900**	**Skip-Gram 0,992**

com nenhum dos modelos de linguagem. A estratégia *soft* utilizada prevê o rótulo da amostra com base no *argmax* das somas das probabilidades das classes previstas. Essa estratégia foi escolhida, pois é a que mais se assemelha à utilizada por Lochter (2015). Entretanto, ela é recomendada quando os classificadores são bem calibrados. No caso dos experimentos deste trabalho, verifica-se que os piores valores de F-medida estão muito abaixo dos melhores valores, se compararmos com os resultados obtidos por Lochter (2015) (Tabela 16 do Apêndice A). Isso faz com que o resultado do Comitê seja penalizado pelos resultados ruins de classificadores individuais. Se fossem excluídos os classificadores com piores desempenhos, o resultado poderia ter sido diferente, mas a opção por mantê-los foi preservada, pois o intuito dos experimentos era poder comparar os resultados com os obtidos por Lochter (2015).

Embora as Tabelas 12 e 13 apresentem somente os melhores resultados, observando as tabelas presentes no Apêndice A, é possível verificar que os modelos de vetores de parágrafo obtiveram a pior F-medida na maioria dos experimentos. O custo computacional para gerar esses modelos limitou uma investigação mais aprofundada para identificar o motivo do resultado inesperadamente ruim. Usando um servidor com 40 núcleos e 364 Gb de memória RAM, o tempo necessário para gerar cada modelo de vetor de parágrafos a partir de um corpus com 100 milhões de amostras, foi de aproximadamente uma semana. Além disso, cada modelo resultante ocupa aproximadamente 78 Gb de espaço em disco. Embora os outros modelos demandem menos poder computacional e espaço, ainda são necessários dias e alguns gigabytes de memória para gerá-los. Levando em consideração que a média da diferença das melhores F-medidas com modelos gerados com 100 milhões e 25 milhões de amostras é 0.012, conclui-se que para obter uma melhora pequena no desempenho é necessário um aumento muito grande no número de amostras e épocas de treinamento, o que pode não ser justificável dado o custo computacional altíssimo necessário para gerar esses modelos.

Observando a Tabela 14, nota-se que para os métodos NB-B e C4.5, experimentos usando representação distribuída obtiveram resultados piores se comparados com o SentMiner. Como o Naive Bayes é um método probabilístico, ele funciona bem com *bag of words*, onde cada atributo representa a ocorrência ou não de uma palavra. O problema com o uso de representações distribuídas é que quando as amostras são convertidas em médias de vetores de palavras, como é o caso de Skip-gram, CBOW e GloVe, ou quando é inferido um vetor que representa a amostra como um todo, no caso do vetor de parágrafo, tratar cada dimensão especificamente como sendo um atributo não faz muito sentido. Nesses casos, raramente vão existir duas amostras, ambas contendo a mesma palavra, que têm o mesmo valor na mesma dimensão, o que explica o desempenho ruim nesses casos. Uma explicação similar pode ser dada para o desempenho ruim do C4.5, cuja criação da árvore de decisão depende de valores individuais de atributos.

Ainda observando a Tabela 14, verifica-se que o SentMiner obteve desempenho superior em muitas combinações de método de classificação $\times$ conjunto de dados, embora não tenha superado representações distribuídas na maioria dos casos. A tabela indica que SVM-R+Skip-gram obteve desempenho melhor na maioria dos casos. Entretanto, com intuito de comprovar a superioridade de pelo menos uma das representações distribuídas de maneira mais criteriosa, foi realizada a análise estatística dos resultados, que é apresentada a seguir.

5.4 Análise estatística

Para avaliar a hipótese de que o emprego de algum dos modelos distribuídos de linguagem oferece desempenho superior em relação aos resultados obtidos por Lochter (2015), foi realizada uma análise estatística seguindo as recomendações de Demšar (2006). Em seu trabalho, ele propõe um conjunto de testes não-paramétricos para comparações estatísticas de resultados de problemas de classificação.

Neste trabalho, foram gerados modelos de linguagem utilizando diferentes técnicas, com diferentes parâmetros para geração, e utilizando corpora de diferentes tamanhos. Após gerados esses modelos, eles foram avaliados usando diferentes algoritmos de classificação em diferentes conjuntos de dados. Assim, o objetivo é determinar se uma combinação **modelo de linguagem** × **parâmetros** × **tamanho do corpus** × **método de classificação** obteve resultado estatisticamente superior ao obtido por Lochter (2015). Como a quantidade de combinações é elevada, a abordagem adotada foi primeiro determinar se algum modelo de linguagem treinado com determinados parâmetros e tamanho de corpus obteve desempenho estatisticamente superior aos demais.

O teste de Friedman é um método não paramétrico usado para determinar se diferentes resultados são estatisticamente equivalentes. Seja r_j^i, o *rank* do resultado do j-ésimo de k algoritmos, obtido no i-ésimo de N conjuntos de dados, o teste compara os *rankings* médios dos algoritmos ($R_j = \frac{1}{N}\sum_i r_j^i$) para determinar se eles são equivalentes. A Equação 5.1 apresenta o teste de Friedman.

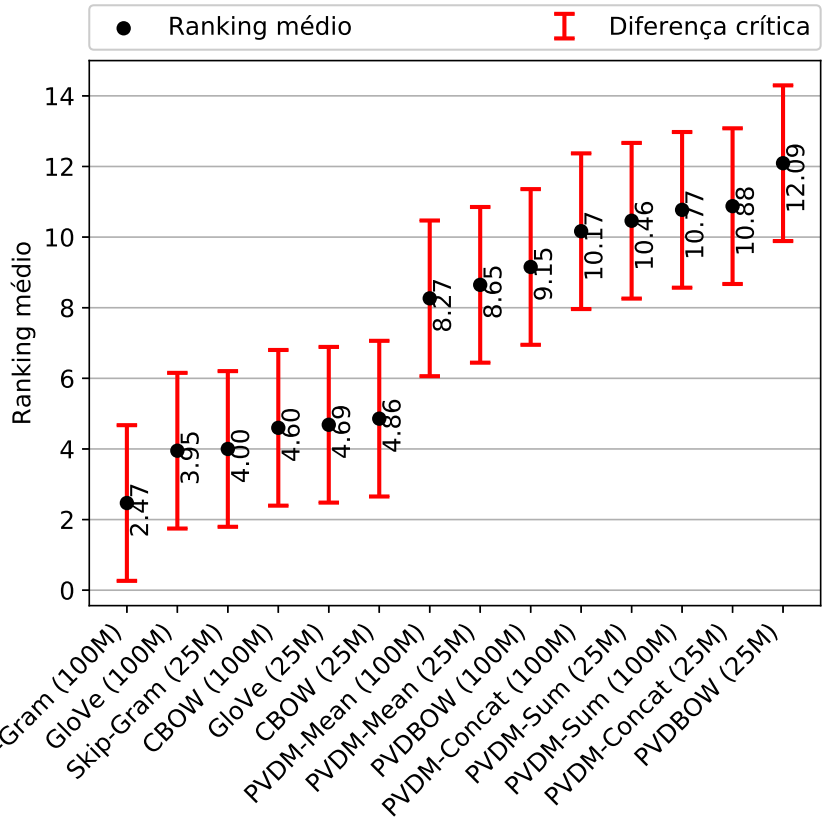
$$\mathcal{X}_F^2 = \frac{12N}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right] \tag{5.1}$$

Davenport (1980) mostrou que $\mathcal{X}_F^2$ é muito conservadora e derivou um teste mais robusto, apresentado na Equação 5.2.

$$F_F = \frac{(N-1)\mathcal{X}_F^2}{N(k-1) - \mathcal{X}_F^2} \tag{5.2}$$

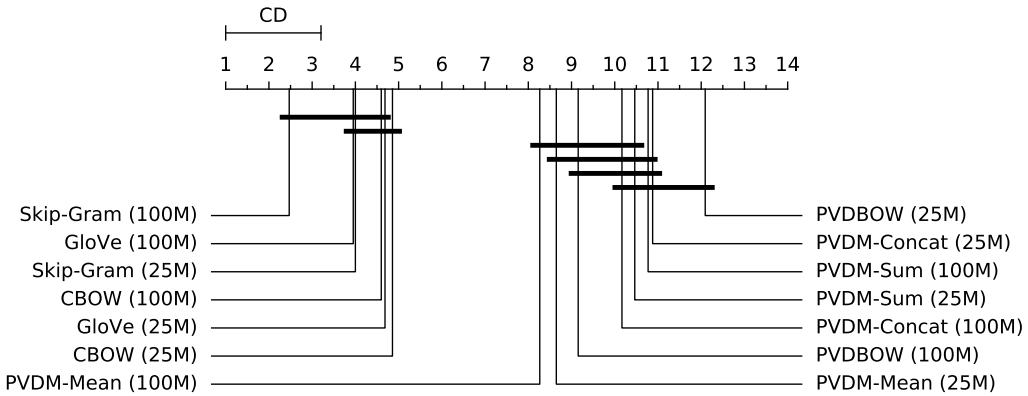
Todos os resultados obtidos pelos modelos de linguagem (e parâmetros + tamanho do corpus) foram estatisticamente comparados usando o teste não-paramétrico de Friedman. O teste F_F mostrou que os diferentes modelos de linguagens não são estatisticamente equivalentes, com grau de confiança de 95%. Com o mesmo grau de confiança, utilizando o método *post-hoc* Nemenyi (DEMŠAR, 2006), foi possível verificar que Skip-gram, CBOW e Glove gerados com 25 e 100 milhões de *tweets* são estatisticamente equivalentes e superiores aos modelos gerados pelo *Paragraph Vector*. As Figuras 16 e 17 permitem visualizar com mais clareza os *rankings* médios e diferenças críticas entre os diferentes modelos.

Figura 16 – *Ranking* médio e diferença crítica dos diferentes modelos de linguagem, obtida no teste *post-hoc* Nemenyi.



Fonte: Autoria própria.

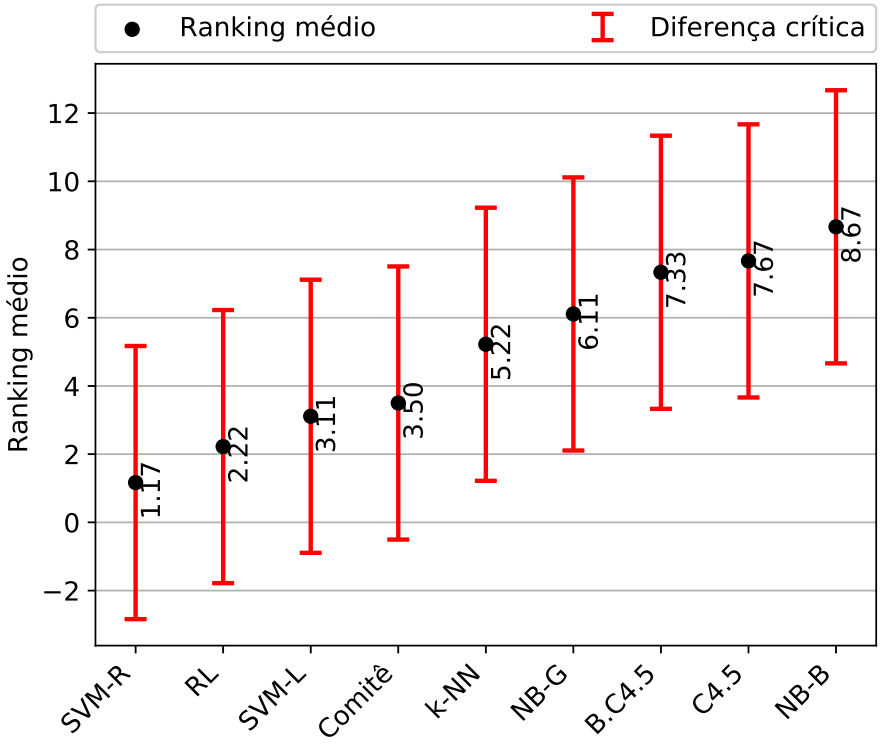
Figura 17 – Diferença crítica entre os diferentes modelos de linguagem, obtida no teste *post-hoc* Nemenyi.



Fonte: Autoria própria.

Uma vez determinado o melhor modelo de linguagem (melhor *ranking* médio: Skip-Gram 100M), o próximo passo foi determinar se algum método de classificação foi estatisticamente superior aos demais. De forma análoga, foi utilizado o teste não-paramétrico de Friedman (F_F), que mostrou que os métodos não são estatisticamente equivalentes, com grau de confiança de 95%. Com o mesmo grau de confiança, foi realizada análise posterior utilizando o método *post-hoc* Nemenyi. As Figuras 18 e 19 permitem visualizar com mais clareza os *rankings* médios e diferenças críticas entre os diferentes métodos e quais são estatisticamente equivalentes entre si. É possível observar que apesar do método SVM-R ter obtido melhor *ranking* médio, ele foi estatisticamente equivalente a RL, SVM-L, Comitê e k-NN, e estatisticamente superior aos métodos NB-B, C4.5, B.C4.5, NB-G.

Figura 18 – *Ranking* médio e diferença crítica dos diferentes métodos de classificação usando o Skip-gram gerado com 100 milhões de amostras, obtidos pelo teste *post-hoc* Nemenyi.

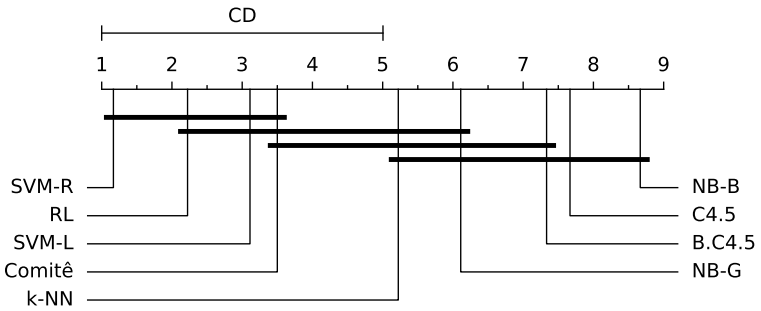


Fonte: Autoria própria.

Finalmente, uma vez encontrado o melhor modelo de linguagem (Skip-Gram 100M) e melhor método de classificação (SVM-R), foi utilizado o teste de Wilcoxon (DEMŠAR, 2006) para comparar os melhores resultados obtidos por Lochter (2015), utilizando comitê de classificadores.

O teste de Wilcoxon é uma alternativa não paramétrica ao teste-T pareado, que classifica as diferenças de desempenho de dois métodos para cada conjunto de dados. O teste calcula a soma dos *rankings* dos valores absolutos das diferenças positivas e negati-

Figura 19 – Diferença crítica dos diferentes métodos de classificação para o Skip-gram gerado com 100 milhões de amostras, obtida pelo teste *post-hoc* Nemenyi.



Fonte: Autoria própria.

vas entre os métodos. Se a menor das somas estiver acima do valor crítico, determinado pelo número de conjuntos de dados, a hipótese nula é descartada. A Tabela 15 mostra as diferenças de desempenho (F-medidas) para o teste em questão. A soma dos *rankings* das diferenças positivas é 37 e das negativas 8. Como foram utilizados 9 conjuntos de dados, a diferença crítica para esse teste, com grau de confiança de 95%, é igual 6. Como a menor das diferenças no nosso caso foi 8, isso indica que a hipótese nula pode ser descartada. Como a soma dos *rankings* das diferenças positivas é maior do que das negativas, isso indica que seguramente os resultados obtidos pela melhor combinação obtida neste trabalho (*i.e.,* SVM-R + Skip-Gram 100M) foram **estatisticamente superiores** ao SentMiner (Comitê + BoW).

Tabela 15 – Dados usados no teste de Wilcoxon, usado para comparar o desempenho do SentMiner com a melhor combinação encontrada neste trabalho.

Conjunto	**SentMiner**	**SVM-R + SG 100M**	**Diferença**
Archeage	0,871	**0,873**	0,002
HCR	0,746	**0,773**	0,027
Hobbit	**0,924**	0,904	-0,020
OMD	**0,826**	0,795	-0,031
SS-Tweet	0,612	**0,800**	0,188
STS-Test	0,870	**0,900**	0,030
Sanders	0,751	**0,836**	0,085
UMICH	0,968	**0,992**	0,024
iPhone6	0,755	**0,823**	0,068

O resultado da análise estatística confirma a hipótese de que o uso de modelos de representação distribuída de texto pode contornar os problemas e desvantagens da *bag of words* em tarefas de análise de sentimento em men-

sagens curtas e ruidosas, mantendo a qualidade preditiva. Mais especificamente, conclui-se que é possível atualizar o SentMiner, substituindo o uso da *bag of words* pelo Skip-gram, e substituindo o Comitê de classificadores pelo SVM-R. Essa substituição faz com que não seja mais necessário o emprego de técnicas de normalização léxica e indexação semântica, diminuindo assim o custo computacional para classificar novas amostras. Outra vantagem é que não será mais necessário usar dicionários, cuja manutenção é problemática diante do surgimento de novas palavras com cada vez mais frequência.

Uma desvantagem da abordagem proposta é que, além de ter sido necessário gerar o modelo de linguagem antes do treinamento do classificador, é preciso transformar cada amostra de acordo com o modelo de linguagem, tanto para o treinamento, quanto para a classificação de amostras ainda não vistas. Contudo, a transformação das amostras demanda poucos recursos computacionais e pode ser realizada de maneira *online*. No caso do Skip-gram, por exemplo, consiste em apenas substituir cada palavra pelo vetor correspondente e calcular a média desses vetores. Entretanto, gerar o modelo de linguagem demandou bastante processamento e memória. Embora essa tarefa precise ser feita apenas uma vez, uma estratégia interessante é atualizar o modelo de tempos em tempos para que novas palavras possam ser incorporadas, e isso é possível com todos os modelos estudados neste trabalho.

Levando em consideração os resultados obtidos e que a geração do modelo é uma tarefa não supervisionada, que demanda apenas grandes corpora de texto, as vantagens desta proposta superam as desvantagens da proposta de Lochter (2015), tais como a dependência de dicionários e de técnicas auxiliares para normalização léxica e indexação semântica.

6 Conclusões

Este trabalho foi motivado pelas dificuldades existentes em realizar análise de sentimento em mensagens curtas e ruidosas. O emprego de métodos de classificação de texto, que constitui uma das abordagens mais utilizadas nesse tipo de problema (PANG; LEE, 2008), tradicionalmente faz uso de *bag of words*. O uso deste tipo de representação traz uma série de deficiências, que se agravam no caso de mensagens curtas e ruidosas. Tradicionalmente, para abordar essas deficiências são empregadas técnicas de normalização léxica e indexação semântica, que fazem uso de dicionários, a manutenção dos quais constitui um problema em si. Este trabalho investigou a hipótese de que o uso de modelos de representação distribuída de texto pode contornar esses problemas, dispensando a necessidade de técnicas tradicionais de normalização léxica e indexação semântica, eliminando a dependência da manutenção de dicionários, preservando a qualidade preditiva e reduzindo o esforço computacional na etapa de treinamento dos classificadores.

Foi apresentado o funcionamento dos modelos de representação distribuída que são mais frequentemente empregados na literatura, e foi suscitada a questão sobre qual o impacto no desempenho da análise de sentimento de mensagens curtas e ruidosas, que é causado pela escolha do corpus utilizado na geração do modelo. Para abordar essa questão, uma série de experimentos foram planejados e executados, e os resultados confirmaram a hipótese de que o uso de corpora com as mesmas características do domínio de aplicação proporciona um desempenho melhor. Esses resultados foram publicados em Lochter et al. (2018).

Na etapa final, foram planejados e executados os experimentos para verificar a hipótese principal de que é possível substituir *bag of words* aliado a técnicas de normalização léxica e indexação semântica, por modelos de representação distribuída de texto. Esses experimentos foram planejados para que seus resultados pudessem ser comparados com os dos experimentos realizados por Lochter (2015), e portanto os mesmos conjuntos de dados e métodos de classificação foram utilizados.

Para avaliar a hipótese, foram gerados dois conjuntos de modelos de linguagem. O primeiro usando 25 milhões de amostras, e o segundo usando 100 milhões. Os modelos do primeiro conjunto foram treinados usando 20 épocas, e os do segundo usando 40 épocas de treinamento. Durante a geração desses modelos verificou-se que uma das grandes dificuldades é o custo computacional necessário. A geração do modelo mais simples levou muitas horas, sendo que o mais complexo levou uma semana para ser gerado. Em termos de espaço em disco, os tamanhos variaram de 6 GB até 78 GB. Essa dificuldade limitou a capacidade de explorar os diferentes parâmetros para geração desses modelos, e o impacto

causado no desempenho da subsequente tarefa de análise de sentimento.

Uma vez gerados os modelos e realizados os experimentos, uma série de conclusões puderam ser tiradas a partir dos resultados. A principal conclusão é que a hipótese investigada pela pesquisa foi confirmada. Foi comprovado, através da análise estatística, que o uso de Skip-gram com SVM Radial obteve um desempenho melhor do que o uso de *bag of words* aliado a técnicas de normalização léxica e indexação semântica. Assim, é possível substituir *bag of words* por pelo menos um dos modelos de representação distribuída, mantendo a qualidade preditiva e com a vantagem de não depender de técnicas baseadas em dicionários.

Comparando todos os resultados, a diferença de desempenho dos modelos gerados por CBOW, Skip-gram e Glove não foi estatisticamente significante, tanto para os modelos gerados com o corpus com 100 milhões de amostras, quanto com 25 milhões de amostras.

Verificou-se também, que para os métodos Naive Bayes Bernoulli e C4.5, os experimentos que usaram representação distribuída obtiveram resultados inferiores, se comparados com os experimentos que usam *bag of words*. Usando representações distribuídas, quando as amostras são convertidas em médias de vetores de palavras (Skip-gram, CBOW e GloVe), ou quando é inferido um vetor que representa a amostra como um todo (vetor de parágrafo), não é viável tratar cada dimensão especificamente como sendo um atributo. A ocorrência de uma palavra é refletida em mudanças nos valores de várias dimensões, o que pode explicar o desempenho ruim desses métodos.

Finalmente, foi verificado que os modelos de vetores de parágrafos obtiveram desempenho inferior na maioria dos métodos e na maioria dos conjuntos de dados. Devido ao tempo proibitivamente longo necessário para gerar esses modelos, não foi possível gerar modelos com parâmetros diferentes, para verificar seu impacto no desempenho. Contudo, a literatura indica que esses modelos dependem de uma quantidade enorme de dados para ajustarem adequadamente os pesos das redes neurais.

Em trabalhos futuros, pretende-se estudar abordagens baseadas em *Deep Learning* aplicadas em problemas de análise de sentimento. Neste trabalho, não foram utilizadas redes neurais artificiais como método de aprendizado, pois pretendia-se comparar os resultados com os obtidos por Lochter (2015) e Lochter et al. (2016). Nos últimos anos, entretanto, muitos artigos na área de aprendizado de máquina têm se concentrado em *Deep Learning*. Esse termo se refere aos métodos geralmente baseados em redes neurais artificias, compostas de vários níveis hierárquicos, onde cada nível aprende a transformar os dados de entrada em uma representação um pouco mais abstrata e composta. Além de serem especialmente capazes de capturar mais distinções a partir de uma quantidade maior de dados de entrada, essas redes neurais artificiais têm a capacidade de executar a extração automática de atributos a partir de dados brutos (BENGIO; COURVILLE, 2013). Usando tais técnicas, o processo de gerar um modelo preditivo capaz de classificar

novas amostras é realizado ao mesmo tempo em que se aprende representações de entrada. Como foi discutido no Capítulo 2, análise de sentimento é um problema um pouco mais complexo do que a tradicional tarefa de classificação do texto em tópicos. Diante do potencial das técnicas de *Deep Learning* de extraírem automaticamente os atributos que melhor contribuem na tarefa de classificação, essa área de pesquisa se mostra promissora.

6.1 Limitações

A principal limitação desta pesquisa foi o custo computacional necessário para gerar os modelos de linguagem. Usando um servidor com 40 núcleos e 364 Gb de memória RAM, o tempo necessário para gerar cada modelo de vetor de parágrafos a partir de um corpus com 100 milhões de *tweets* e 40 épocas de treinamento, foi de aproximadamente uma semana. Além disso, cada modelo resultante ocupa aproximadamente 78 Gb de espaço em disco. Embora os outros modelos demandem menos poder computacional e espaço, esses foram fatores limitantes que impediram o estudo do impacto de diferentes parâmetros na geração desses modelos.

6.2 Publicações

J. VON LOCHTER, P.R. PIRES, C.A. BOSSOLANI, A. YAMAKAMI and T.A. ALMEIDA. Evaluating the impact of corpora used to train distributed text representation models for noisy and short texts. In *Proceedings of the 31st IEEE International Joint Conference on Neural Networks (IJCNN'18)*, 315-322, Rio de Janeiro, Brazil, July, 2018.

Referências

AGARWAL, A. et al. Sentiment analysis of Twitter data. In: *Proceedings of the Workshop on Languages in Social Media - LSM' 11*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011. p. 30–38. Citado 2 vezes nas páginas 65 e 73.

ALMEIDA, T. A. et al. Text normalization and semantic indexing to enhance Instant Messaging and SMS spam filtering. *Knowledge-Based Systems*, v. 108, p. 25–32, 2016. Citado na página 40.

ARSLAN, Y.; KÜÇÜK, D.; BIRTURK, A. Twitter sentiment analysis experiments using word embeddings on datasets of various scales. In: SILBERZTEIN, M. et al. (Ed.). *Natural Language Processing and Information Systems*. Cham: Springer International Publishing, 2018. p. 40–47. Citado na página 61.

ASSIS, M. D. *Obra completa*. Rio de Janeiro: Nova Aguilar, 1994. Citado 2 vezes nas páginas 13 e 45.

BENGIO, Y. Learning deep architectures for AI. *Foundations and Trends® in Machine Learning*, v. 2, n. 1, p. 1–127, 2009. Citado na página 57.

BENGIO, Y.; COURVILLE, A. Deep Learning of Representations. *Intelligent Systems Reference Library*, v. 49, p. 1–28, 2013. ISSN 18684394. Citado na página 86.

BENGIO, Y. et al. A neural probabilistic language model. *The Journal of Machine Learning Research*, v. 3, p. 1137–1155, 2003. Citado 3 vezes nas páginas 25, 46 e 60.

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. New York, NY, USA: ACM, 1992. (COLT '92), p. 144–152. Citado na página 74.

BREIMAN, L. et al. *Classification and regression trees*. New York: Routledge, 1984. Citado na página 74.

BURNAP, P. et al. Tweeting the terror: modelling the social media reaction to the Woolwich terrorist attack. *Social Network Analysis and Mining*, v. 4, n. 1, p. 1–14, 2014. Citado na página 30.

CAMBRIA, E. et al. An ELM-based model for affective analogical reasoning. *Neurocomputing*, Elsevier, v. 149, n. Part A, p. 443–455, 2015. Citado na página 30.

COLLOBERT, R.; WESTON, J. A unified architecture for natural language processing: deep neural networks with multitask learning. In: *Proceedings of the 25th International Conference on Machine Learning - ICML' 08*. Helsinki, Finland: ACM Press, 2008. p. 160–167. Citado na página 46.

CORTES, C.; VAPNIK, V. Support-vector networks. *Machine Learning*, v. 20, n. 3, p. 273–297, sep 1995. ISSN 1573-0565. Citado na página 74.

DAVE, K. et al. Mining the peanut gallery: opinion extraction and semantic classification of product reviews. *Proceedings of the 12th International Conference on World Wide Web*, p. 519–528, 2003. Citado 4 vezes nas páginas 28, 29, 32 e 35.

DAVENPORT, J. M. Approximations of the critical region of the friedman statistic. *Communications in Statistics - Theory and Methods*, v. 9, n. 6, p. 571–595, 1980. Citado na página 79.

DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, v. 7, p. 1–30, 2006. Citado 2 vezes nas páginas 79 e 81.

GIATSOGLOU, M. et al. Sentiment analysis leveraging emotions and word embeddings. *Expert Systems with Applications*, Elsevier Ltd, v. 69, p. 214–224, 2017. Citado na página 60.

GRINTER, R.; ELDRIDGE, M. Wan2tlk?: everyday text messaging. *Proceedings of the Conference on Human Factors in Computing Systems CHI'03*, n. 5, p. 441–448, 2003. Citado na página 30.

HAN, B.; BALDWIN, T. Lexical normalisation of short text messages : makn sens a # twitter. *Computational Linguistics*, v. 5, n. 212, p. 368–378, 2011. Citado na página 39.

HATZIVASSILOGLOU, V.; MCKEOWN, K. R. Predicting the semantic orientation of adjectives. *Proceedings of the 35th Annual Meeting on Association for Computational Linguistics*, p. 174–181, 1997. Citado 2 vezes nas páginas 28 e 29.

HIDALGO, J. M. G.; RODRÍGUEZ, M. d. B.; PÉREZ, J. C. C. The role of word sense disambiguation in automated text categorization. In: *Natural Language Processing and Information Systems*. Alicante, Spain: Springer Berlin Heidelberg, 2005. v. 3513, n. June, p. 298–309. Citado 2 vezes nas páginas 40 e 42.

HOGENBOOM, A. et al. Multi-lingual support for lexicon-based sentiment analysis guided by semantics. *Decision Support Systems*, Elsevier B.V., v. 62, p. 43–53, 2014. Citado na página 30.

HORRIGAN, J. B. Online Shopping. *Pew Internet & American Life Project*, 2008. Citado na página 28.

JAIN, T. I.; NEMADE, D. Recognizing contextual polarity in phrase-level sentiment analysis. *International Journal of Computer Applications*, v. 7, n. 5, p. 12–21, 2010. Citado na página 32.

JOULIN, A. et al. Bag of Tricks for Efficient Text Classification. *arXiv preprint arXiv:1607.01759*, jul 2016. Disponível em: <http://arxiv.org/abs/1607.01759>. Citado na página 71.

JOULIN, A. et al. FastText.zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*, dec 2016. Disponível em: <http://arxiv.org/abs/1612.03651>. Citado na página 71.

KALFSBEEK, H. W.; DUBOIS, D. Elicitation, assessment, and pooling of expert judgments using possibility theory. *IEEE Transactions on Fuzzy Systems*, v. 3, n. 3, p. 313–335, 1995. Citado na página 28.

KAUFMANN, M. Syntactic normalization of Twitter messages. *International Conference on Natural Language Processing*, 2010. Citado na página 39.

KIRITCHENKO, S.; ZHU, X.; MOHAMMAD, S. M. Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research*, v. 50, p. 723–762, 2014. Citado na página 32.

LE, Q.; MIKOLOV, T. Distributed representations of sentences and documents. In: *Proceedings of the 31st International Conference on Machine Learning, ICML'14*. Beijing, China: [s.n.], 2014. v. 32, p. 1188–1196. Citado 4 vezes nas páginas 25, 55, 56 e 57.

LIPSMAN, A. Online Consumer-Generated Reviews Have Significant Impact on Offline Purchase Behavior. *http://www. comscore. com/por/Insights/-Press_Releases/2007/11/Online_Consumer_Reviews_Impact_Offline_Pur chasing_Behavior*, 2007. Citado na página 28.

LIU, B. Sentiment analysis and subjectivity. *Handbook of Natural Language Processing*, 2010. Citado na página 28.

LIU, B. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, v. 5, n. 1, p. 1–167, 2012. Citado 2 vezes nas páginas 28 e 30.

LOCHTER, J. V. Máquinas de classificação para detectar polaridade de mensagens de texto em redes sociais. *Dissertação (Mestrado em Ciência da Computação) - Universidade Federal de São Carlos*, p. 85, 2015. Citado 15 vezes nas páginas 15, 24, 26, 40, 41, 42, 73, 76, 77, 79, 81, 83, 85, 86 e 95.

LOCHTER, J. V. et al. Short text opinion detection using ensemble of classifiers and semantic indexing. *Expert Systems with Applications*, Elsevier, v. 1, p. 1–28, 2016. Citado 4 vezes nas páginas 40, 73, 74 e 86.

LOCHTER, J. V. et al. Evaluating the impact of corpora used to train distributed text representation models for noisy and short texts. In: *International Joint Conference on Neural Networks (IJCNN'18)*. [S.l.: s.n.], 2018. v. 1, p. 315–322. Citado 2 vezes nas páginas 69 e 85.

MÄNTYLÄ, M. V.; GRAZIOTIN, D.; KUUTILA, M. The evolution of sentiment analysis - a review of research topics, venues, and top cited papers. *Computer Science Review*, v. 27, p. 16–32, dec 2016. Disponível em: <http://arxiv.org/abs/1612.01556>. Citado na página 28.

MCCALLUM, A.; NIGAM, K. A comparison of event models for naive Bayes text classification. *AAAI-98 Workshop on Learning for Text Categorization*, v. 752, n. 1, p. 41–48, 1998. Citado na página 74.

MCKINNEY, W. Data structures for statistical computing in Python. *Proceedings of the 9th Python in Science Conference*, v. 1697900, p. 51–56, 2010. Citado na página 74.

MIKOLOV, T. et al. Distributed representations of words and phrases and their compositionality. In: *Advances in Neural Information Processing Systems 26, NIPS'13*. [S.l.]: Curran Associates, Inc., 2013. p. 3111–3119. Citado 6 vezes nas páginas 52, 53, 60, 63, 64 e 66.

MIKOLOV, T. et al. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. Disponível em: <http://arxiv.org/abs/1301.3781>. Citado 4 vezes nas páginas 25, 47, 60 e 63.

MIKOLOV, T.; YIH, W.-t.; ZWEIG, G. Linguistic regularities in continuous space word representations. In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. [S.l.: s.n.], 2013. p. 746–751. Citado na página 57.

MILLER, G. A. WordNet: a lexical database for english. *Communications of the ACM*, v. 38, n. 11, p. 39–41, 1995. Citado na página 39.

NASSIRTOUSSI, A. K. et al. Text mining for market prediction: a systematic review. *Expert Systems with Applications*, Elsevier, v. 41, n. 16, p. 7653–7670, 2014. Citado na página 30.

NAVIGLI, R.; PONZETTO, S. P. BabelNet: building a very varge multilingual semantic network. In: *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL' 10*. Uppsala, Sweden: [s.n.], 2010. p. 216–225. Citado na página 39.

O 'Connor, B. et al. From Tweets to polls: linking text sentiment to public opinion time series. *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media*, v. 11, n. 122-129, p. 1–2, 2010. Citado na página 31.

PANG, B.; LEE, L. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, v. 2, n. 1-2, p. 1–135, 2008. Citado 5 vezes nas páginas 23, 27, 28, 35 e 85.

PANG, B.; LEE, L.; VAITHYANATHAN, S. Thumbs up?: sentiment classification using machine learning techniques. *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing, EMNLP'02*, Stroudsburg, PA, USA, v. 10, p. 79–86, 2002. Citado 3 vezes nas páginas 29, 32 e 35.

PATODKAR, V. N.; I.R, S. Twitter as a corpus for sentiment analysis and opinion mining. *LREc*, v. 10, p. 1320–1326, 2010. Citado na página 32.

PEDREGOSA, F. et al. Scikit-learn: machine learning in python. *CoRR*, v. 12, p. 2825–2830, 2012. Disponível em: <http://arxiv.org/abs/1201.0490>. Citado na página 74.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. GloVe: global vectors for word representation. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP'14*. Doha, Qatar: [s.n.], 2014. p. 1532–1543. Citado 5 vezes nas páginas 25, 58, 59, 63 e 72.

REXHA, A. et al. Polarity classification for target phrases in tweets: A word2vec approach. In: *International Semantic Web Conference*. Cham: Springer, 2016. p. 217–223. Citado na página 60.

REYES, A.; ROSSO, P. On the difficulty of automatically detecting irony: beyond a simple case of negation. *Knowledge and Information Systems*, v. 40, n. 3, p. 595–614, 2014. Citado na página 30.

SAHLGREN, M. The distributional hypothesis. *Italian Journal of Linguistics*, v. 20, n. 1, p. 33–54, 2008. Citado na página 45.

SAIF, H. et al. Evaluation datasets for Twitter sentiment analysis a survey and a new dataset, the STS-Gold. In: *1st Interantional Workshop on Emotion and Sentiment in Social and Expressive Media: Approaches and Perspectives from AI (ESSEM 2013)*. Turin, Italy: [s.n.], 2013. Citado na página 65.

SALTON, G.; MCGILL, M. J. *Introduction to modern information retrieval*. 1983. Citado na página 74.

SCHAPIRE, R. E. A brief introduction to boosting. In: *IJCAI International Joint Conference on Artificial Intelligence*. [S.l.: s.n.], 1999. Citado na página 74.

SHAMMA, D. a.; KENNEDY, L.; CHURCHILL, E. F. Tweet the debates: understanding community annotation of uncollected sources. In: *Proceedings of the First SIGMM Workshop on Social Media, WSM'09*. Beijing, China: [s.n.], 2009. p. 3–10. Citado 2 vezes nas páginas 65 e 73.

SITARAM, A.; HUBERMAN, B. A. Predicting the future with social media. In: *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology-Volume 01*. [S.l.: s.n.], 2010. Citado na página 31.

SOCHER, R.; LIN, C. Parsing natural scenes and natural language with recursive neural networks. *Proceedings of the 28th international conference on machine learning (ICML-11)*, p. 129–136, 2011. Citado na página 55.

SPERIOSU, M. et al. Twitter polarity classification with label propagation over lexical links and the follower graph. In: *Proceedings of the First Workshop on Unsupervised Learning in NLP, EMNLP'11*. Edinburgh, Scotland: [s.n.], 2011. p. 53–63. Citado 2 vezes nas páginas 65 e 73.

STAGNER, R. The cross-out technique as a method in public opinion analysis. *The Journal of Social Psychology*, Taylor & Francis, v. 11, n. 1, p. 79–90, 1940. Citado na página 28.

THELWALL, M.; BUCKLEY, K.; PALTOGLOU, G. Sentiment strength detection for the social web. *Journal of the American Society for Information Science and Technology*, v. 63, n. 1, p. 163–173, 2012. Citado 2 vezes nas páginas 65 e 73.

THELWALL, M. et al. Sentiment strength detection in short informal text. *Journal of the American Society for Information Science and Technology*, v. 61, n. 12, p. 2544–2558, 2010. Citado na página 31.

THURLOW, C. Generation txt? the sociolinguistics of young people's text-messaging. *Discourse Analysis Online*, n. January 2003, p. 1–31, 2003. Citado na página 30.

TUMASJAN, A. et al. Predicting elections with Twitter: what 140 characters reveal about political sentiment. *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media*, p. 178–185, 2010. Citado na página 31.

TURNEY, P. D. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL'02*. Philadelphia, PA, USA: [s.n.], 2002. p. 417–424. Citado 2 vezes nas páginas 29 e 32.

TURNEY, P. D.; LITTMAN, M. L. Measuring praise and criticism. *ACM Transactions on Information Systems*, v. 21, n. 4, p. 315–346, 2003. Citado 2 vezes nas páginas 29 e 32.

TURNEY, P. D.; PANTEL, P. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, v. 37, p. 141–188, 2010. Citado na página 46.

WALKER, S. H.; DUNCAN, D. B. Estimation of the probability of an event as a function of several independent variables. *Biometrika*, 1967. Citado na página 74.

WANG, Y. et al. Word vector modeling for sentiment analysis of product reviews. In: *Natural Language Processing and Chinese Computing: Third CCF Conference, NLPCC 2014*. Shenzhen, China: [s.n.], 2014. p. 168–180. Citado na página 60.

WIEBE, J. et al. Learning subjective language. *Computational Linguistics*, v. 30, n. 3, p. 277–308, 2004. Citado na página 30.

WIEBE, J. M. *Recognizing subjective sentences: a computational investigation of narrative text*. Tese (Doutorado), Buffalo, NY, USA, 1990. Citado na página 28.

WIEBE, J. M.; BRUCE, R. F.; O'HARA, T. P. Development and use of a gold standard data set for subjectivity classifications. *Proceedings of the Association for Computational Linguistics*, p. 246–253, 1999. Citado na página 28.

XIA, R.; ZONG, C.; LI, S. Ensemble of feature sets and classification algorithms for sentiment classification. *Information Sciences*, Elsevier Inc., v. 181, n. 6, p. 1138–1152, 2011. Citado na página 74.

YANG, J.; LESKOVEC, J. Patterns of temporal variation in online media. In: ACM. *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*. [S.l.], 2011. p. 177–186. Citado na página 64.

ZHANG, L.; WANG, S.; LIU, B. Deep learning for sentiment analysis : a survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, v. 8, n. 4, p. 1253, 2018. Citado na página 33.

Apêndice A – Resultados detalhados

A Tabela 16 apresenta todos os resultados obtidos por Lochter (2015). As Tabelas 17 a 30 apresentam os resultados de cada um dos experimentos realizados neste trabalho de pesquisa, com cada modelo de linguagem × método de classificação × base de dados.

Tabela 16 – F-medida obtidos para cada método avaliado nos experimentos realizados por Lochter (2015) usando amostras expandidas sem aplicar seleção de atributos.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	0,860	0,716	0,862	0,646	0,787	0,688	0,516	0,821	0,951
B.C4.5	0,758	0,656	0,825	0,686	0,703	0,655	0,573	0,680	0,925
C4.5	0,746	0,627	0,838	0,687	0,707	0,651	0,573	0,736	0,907
Comitê	**0,871**	**0,746**	**0,924**	**0,755**	**0,826**	**0,751**	**0,612**	**0,870**	**0,968**
NB-G	0,798	0,629	0,592	0,612	0,684	0,634	0,517	0,765	0,778
k-NN	0,700	0,642	0,828	0,650	0,680	0,667	0,591	0,788	0,948
RL	0,855	0,733	0,914	0,715	0,806	0,706	0,603	0,807	0,959
NB-M	0,848	0,705	0,862	0,696	0,790	0,710	0,545	0,833	0,941
SVM-L	0,846	0,722	0,913	0,733	0,802	0,705	0,593	0,818	0,962
SVM-R	0,842	0,732	0,909	0,692	0,781	0,698	0,515	0,845	0,948

Tabela 17 – F-medida para cada método avaliada nos experimentos realizados utilizando PVDBOW gerado com 100 milhões de amostras e 40 épocas de treinamento.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	0,579	0,623	0,678	0,697	0,627	0,527	0,547	0,563	0,571
B.C4.5	0,587	0,578	0,640	0,645	0,578	0,547	0,535	0,613	0,663
C4.5	0,608	0,545	0,628	0,652	0,552	0,551	0,557	0,591	0,663
Comitê	**0,690**	**0,637**	**0,738**	**0,726**	**0,646**	0,620	**0,654**	0,646	0,766
NB-G	0,637	0,611	0,713	0,692	0,605	**0,634**	0,644	0,677	0,687
k-NN	0,612	0,623	0,682	0,699	0,536	0,599	0,590	0,568	**0,816**
RL	0,586	0,623	0,678	0,697	0,627	0,632	0,625	**0,702**	0,738
SVM-L	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,723
SVM-R	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,570

Tabela 18 – F-medida para cada método avaliada nos experimentos realizados utilizando PVDM-Concat gerado com 100 milhões de amostras e 40 épocas de treinamento.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,570
B.C4.5	0,605	0,554	0,590	0,615	0,583	0,558	0,539	0,518	0,779
C4.5	0,601	0,557	0,598	0,656	0,546	0,544	0,517	0,574	0,781
Comitê	**0,708**	0,623	0,693	0,692	0,653	0,632	0,570	0,618	0,816
NB-G	0,612	0,582	0,613	0,647	0,570	0,612	0,542	0,588	0,694
k-NN	0,626	**0,633**	**0,713**	**0,707**	0,558	0,577	0,572	0,571	**0,911**
RL	0,678	0,631	0,682	0,695	**0,662**	**0,658**	**0,592**	**0,643**	0,787
SVM-L	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,630
SVM-R	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,570

Tabela 19 – F-medida para cada método avaliada nos experimentos realizados utilizando PVDM-Mean gerado com 100 milhões de amostras e 40 épocas de treinamento.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,570
B.C4.5	0,582	0,574	0,630	0,658	0,562	0,567	0,529	0,538	0,778
C4.5	0,590	0,550	0,628	0,628	0,568	0,548	0,525	0,599	0,775
Comitê	0,742	0,661	0,757	0,709	0,659	0,698	0,631	0,677	0,925
NB-G	0,643	0,614	0,709	0,637	0,567	0,621	0,580	0,588	0,714
k-NN	0,581	0,604	0,738	**0,711**	0,502	0,556	0,550	0,574	0,922
RL	0,763	0,647	0,747	0,701	**0,668**	**0,722**	**0,706**	0,669	0,923
SVM-L	0,769	0,655	0,774	0,699	0,659	0,703	0,658	0,641	0,921
SVM-R	**0,774**	**0,661**	**0,797**	0,703	0,664	0,718	0,645	**0,680**	**0,929**

Tabela 20 – F-medida para cada método avaliada nos experimentos realizados utilizando PVDM-Sum gerado com 100 milhões de amostras e 40 épocas de treinamento.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	0,579	**0,623**	0,678	0,697	0,627	0,524	0,547	0,507	0,570
B.C4.5	0,564	0,543	0,623	0,620	0,556	0,560	0,530	0,554	0,591
C4.5	0,555	0,540	0,621	0,635	0,557	0,528	0,555	0,532	0,583
Comitê	0,670	0,623	**0,715**	**0,726**	**0,637**	0,636	0,622	0,641	**0,720**
NB-G	**0,702**	0,617	0,644	0,643	0,609	**0,641**	**0,646**	**0,688**	0,672
k-NN	0,439	0,622	0,678	0,697	0,428	0,501	0,593	0,596	0,696
RL	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,671	0,570
SVM-L	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,570
SVM-R	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,570

Tabela 21 – F-medida para cada método avaliada nos experimentos realizados utilizando CBOW gerado com 100 milhões de amostras e 40 épocas de treinamento.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,570
B.C4.5	0,729	0,573	0,703	0,699	0,633	0,626	0,593	0,599	0,951
C4.5	0,692	0,595	0,720	0,680	0,629	0,603	0,607	0,669	0,954
Comitê	0,829	0,717	0,858	**0,820**	0,763	0,799	0,747	0,822	0,985
NB-G	0,718	0,628	0,692	0,684	0,664	0,667	0,640	0,719	0,762
k-NN	0,761	0,673	0,751	0,776	0,690	0,731	0,643	0,696	0,955
RL	**0,860**	0,732	0,900	0,797	0,775	**0,819**	**0,781**	**0,841**	0,989
SVM-L	0,857	**0,753**	**0,904**	0,788	0,781	0,818	0,781	0,833	**0,990**
SVM-R	0,859	0,746	0,895	0,803	**0,797**	0,814	0,765	0,833	0,989

Tabela 22 – F-medida para cada método avaliada nos experimentos realizados utilizando Skip-Gram gerado com 100 milhões de amostras e 40 épocas de treinamento.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,570
B.C4.5	0,735	0,668	0,709	0,671	0,635	0,664	0,609	0,705	0,953
C4.5	0,746	0,613	0,716	0,680	0,630	0,659	0,606	0,677	0,957
Comitê	0,855	0,744	0,870	0,821	0,765	0,808	0,779	0,894	0,988
NB-G	0,759	0,630	0,785	0,703	0,676	0,677	0,697	0,864	0,869
k-NN	0,827	0,729	0,759	0,789	0,748	0,774	0,738	0,844	0,966
RL	0,870	0,753	0,891	0,818	0,794	**0,836**	0,798	0,894	0,991
SVM-L	0,861	0,741	**0,906**	0,806	0,788	0,822	0,789	0,883	0,989
SVM-R	**0,873**	**0,773**	0,904	**0,823**	**0,795**	0,836	**0,800**	**0,900**	**0,992**

Tabela 23 – F-medida para cada método avaliada nos experimentos realizados utilizando GloVe gerado com 100 milhões de amostras e 40 épocas de treinamento.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	0,579	0,623	0,678	0,697	0,627	0,541	0,547	0,577	0,570
B.C4.5	0,740	0,606	0,707	0,669	0,613	0,656	0,622	0,657	0,951
C4.5	0,736	0,617	0,724	0,675	0,627	0,621	0,616	0,652	0,946
Comitê	0,834	0,712	0,875	0,786	0,756	0,778	0,751	0,852	0,984
NB-G	0,705	0,589	0,774	0,667	0,675	0,664	0,668	0,808	0,865
k-NN	0,786	0,684	0,755	0,761	0,722	0,717	0,685	0,758	0,962
RL	0,857	**0,742**	**0,908**	**0,806**	0,779	0,814	0,781	0,850	**0,990**
SVM-L	**0,864**	0,737	0,906	0,799	**0,782**	0,818	0,783	0,855	0,990
SVM-R	0,853	0,734	0,874	0,799	0,776	**0,825**	**0,785**	**0,869**	0,989

Tabela 24 – F-medida para cada método avaliada nos experimentos realizados utilizando PVDBOW gerado com 25 milhões de amostras e 20 épocas de treinamento.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	0,579	**0,623**	**0,678**	0,697	**0,627**	0,524	0,547	0,507	0,570
B.C4.5	0,544	0,544	0,579	0,624	0,564	0,525	0,508	0,554	0,594
C4.5	0,555	0,536	0,556	0,573	0,539	0,522	0,502	0,535	0,587
Comitê	0,596	0,600	0,649	0,675	0,595	0,544	0,549	0,563	0,694
NB-G	**0,597**	0,585	0,605	0,622	0,532	0,537	0,565	0,560	0,671
k-NN	0,582	0,597	0,657	**0,697**	0,599	**0,544**	0,519	0,549	**0,753**
RL	0,584	0,623	0,678	0,697	0,627	0,544	**0,569**	0,574	0,661
SVM-L	0,544	0,572	0,577	0,570	0,561	0,529	0,522	**0,585**	0,591
SVM-R	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,570

Tabela 25 – F-medida para cada método avaliada nos experimentos realizados utilizando PVDM-Concat gerado com 25 milhões de amostras e 20 épocas de treinamento.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,570
B.C4.5	0,573	0,544	0,672	0,639	0,566	0,567	0,526	0,526	0,813
C4.5	0,573	0,526	0,655	0,626	0,582	0,536	0,510	0,487	0,809
Comitê	0,607	0,625	**0,705**	0,679	0,645	0,580	0,546	0,510	0,885
NB-G	0,585	0,569	0,648	0,624	0,578	**0,598**	0,547	0,538	0,705
k-NN	0,583	**0,635**	0,659	**0,697**	0,597	0,512	0,557	0,535	**0,919**
RL	**0,630**	0,628	0,672	0,694	**0,651**	0,593	**0,564**	**0,596**	0,765
SVM-L	0,557	0,513	0,690	0,632	0,537	0,570	0,551	0,574	0,672
SVM-R	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,570

Tabela 26 – F-medida para cada método avaliada nos experimentos realizados utilizando PVDM-Mean gerado com 25 milhões de amostras e 20 épocas de treinamento.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,570
B.C4.5	0,591	0,558	0,670	0,602	0,567	0,545	0,521	0,546	0,757
C4.5	0,604	0,534	0,634	0,617	0,562	0,541	0,532	0,510	0,770
Comitê	0,748	0,621	0,745	0,703	0,682	0,698	0,645	0,655	**0,925**
NB-G	0,618	0,558	0,693	0,645	0,595	0,621	0,578	0,624	0,697
k-NN	0,588	0,634	0,707	0,697	0,481	0,535	0,578	0,557	0,920
RL	**0,779**	**0,654**	0,739	0,709	**0,715**	0,708	**0,713**	**0,694**	0,918
SVM-L	0,766	0,644	0,757	0,701	0,696	**0,712**	0,689	0,671	0,910
SVM-R	0,771	0,649	**0,778**	**0,711**	0,711	0,694	0,658	0,680	0,918

Tabela 27 – F-medida para cada método avaliada nos experimentos realizados utilizando PVDM-Sum gerado com 25 milhões de amostras e 20 épocas de treinamento.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	0,579	**0,623**	0,678	0,697	0,627	0,524	0,547	0,507	0,570
B.C4.5	0,586	0,547	0,621	0,600	0,562	0,524	0,542	0,579	0,619
C4.5	0,577	0,558	0,582	0,577	0,559	0,494	0,550	0,487	0,594
Comitê	0,680	0,616	0,692	0,701	0,643	0,635	0,630	0,568	0,751
NB-G	0,617	0,599	0,669	0,656	0,581	0,621	0,601	0,638	0,627
k-NN	0,475	0,616	0,678	0,697	0,562	0,607	0,588	0,610	0,788
RL	0,579	0,623	0,678	0,697	0,627	0,552	0,547	**0,699**	0,570
SVM-L	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,599
SVM-R	**0,735**	0,618	**0,713**	**0,726**	**0,700**	**0,698**	**0,661**	0,646	**0,792**

Tabela 28 – F-medida para cada método avaliada nos experimentos realizados utilizando CBOW gerado com 25 milhões de amostras e 20 épocas de treinamento.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,507	0,570
B.C4.5	0,708	0,623	0,690	0,664	0,616	0,643	0,593	0,613	0,952
C4.5	0,693	0,573	0,674	0,712	0,636	0,624	0,594	0,641	0,948
Comitê	0,831	0,715	0,828	0,789	0,761	0,779	0,739	0,811	0,982
NB-G	0,712	0,624	0,701	0,686	0,659	0,660	0,639	0,733	0,740
k-NN	0,759	0,675	0,730	0,750	0,710	0,705	0,674	0,730	0,952
RL	0,853	0,741	0,868	**0,808**	0,772	0,809	**0,783**	**0,869**	0,982
SVM-L	0,856	0,760	**0,893**	0,801	0,779	**0,812**	0,782	0,861	0,987
SVM-R	**0,858**	**0,762**	0,891	0,803	**0,792**	0,808	0,782	0,836	**0,988**

Tabela 29 – F-medida para cada método avaliada nos experimentos realizados utilizando Skip-Gram gerado com 25 milhões de amostras e 20 épocas de treinamento.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	0,579	0,623	0,678	0,697	0,627	0,529	0,547	0,563	0,570
B.C4.5	0,732	0,628	0,720	0,709	0,634	0,645	0,606	0,696	0,951
C4.5	0,747	0,608	0,699	0,714	0,652	0,668	0,616	0,630	0,954
Comitê	0,831	0,731	0,839	**0,803**	0,738	0,769	0,751	0,855	0,976
NB-G	0,746	0,630	0,787	0,712	0,674	0,685	0,704	0,875	0,816
k-NN	0,836	0,732	0,761	0,784	0,748	0,763	0,734	0,855	0,966
RL	0,825	0,685	0,680	0,697	0,739	0,767	0,769	**0,894**	0,980
SVM-L	0,621	0,623	0,678	0,697	0,627	0,590	0,550	0,507	0,936
SVM-R	**0,879**	**0,772**	**0,912**	0,793	**0,802**	**0,839**	**0,795**	0,886	**0,991**

Tabela 30 – F-medida para cada método avaliada nos experimentos realizados utilizando GloVe gerado com 25 milhões de amostras e 20 épocas de treinamento.

Método	Archeage	HCR	Hobbit	iPhone6	OMD	Sanders	SS-Tweet	STS-Test	UMICH
NB-B	0,579	0,623	0,678	0,697	0,627	0,524	0,547	0,599	0,570
B.C4.5	0,729	0,602	0,695	0,679	0,617	0,638	0,615	0,635	0,951
C4.5	0,737	0,619	0,709	0,705	0,635	0,643	0,616	0,591	0,959
Comitê	0,816	0,708	0,864	0,801	0,743	0,770	0,736	0,794	0,980
NB-G	0,709	0,616	0,784	0,695	0,685	0,659	0,675	0,794	0,856
k-NN	0,781	0,671	0,780	0,752	0,732	0,724	0,678	0,777	0,966
RL	0,757	0,625	0,678	0,697	0,687	0,683	0,706	0,783	0,961
SVM-L	**0,854**	0,750	0,908	0,799	0,775	**0,818**	**0,781**	**0,838**	**0,991**
SVM-R	0,854	**0,760**	**0,914**	**0,801**	**0,784**	0,815	0,780	0,816	0,990