



Universidade Federal de Pernambuco
Centro de Informática



Bacharelado em Ciência da Computação

Estudo Comparativo dos Métodos de *Word Embedding* na Análise de Sentimentos

MATHEUS HERMÍNIO DE CARVALHO

Recife
2018

Bacharelado em Ciência da Computação

MATHEUS HERMÍNIO DE CARVALHO

**Estudo Comparativo dos Métodos de *Word Embedding* na
Análise de Sentimentos**

Monografia apresentada ao Centro de Informática (CIN) da Universidade Federal de Pernambuco (UFPE), como requisito parcial para conclusão do Curso de Ciência da Computação, orientada pelo professor Cleber Zanchettin.

Recife
2018

Resumo

Word Embeddings são métodos que fornecem boas representações vetoriais contínuas de baixa dimensão para conjuntos de textos não estruturados, quando combinados com classificadores melhoram o desempenho do modelo. *Word Embedding* é muito usado em diferentes tarefas de processamento de linguagem natural tais como reconhecimento da linguagem natural, similaridade entre palavras, classificação de documentos, *parsing*, análise de sentimentos e etc. Neste trabalho, foram comparados os métodos LSA, Word2Vec (CBOW), GloVe e SSWE com as configurações de dimensionalidade 50, 100 e 300. A qualidade destas representações foi avaliada pela tarefa de classificação da polaridade de *reviews* utilizando a base de dados da IMDb contendo cerca de 25.000 *reviews* sobre filmes, as comparações ocorreram observando os resultados obtidos da rede neural perceptron. Os experimentos realizados evidenciaram que o classificador perceptron com o método SSWE obteve a melhor acurácia média de 82,12% na tarefa de classificação da polaridade.

Palavras-chave: *Word Embedding*, Análise Sentimentos, Word2vec, LSA, GloVe, SSWE, aprendizagem de máquina.

Abstract

Word Embeddings are methods that provide good low-dimensional continuous vector representations for unstructured text sets, when combined with classifiers improve the performance of the model. Word Embedding is widely used in different tasks of natural language processing such as natural language recognition, word similarity, document classification, parsing, sentiment analysis and so on. In this work, the methods compared were LSA, Word2Vec (CBOW), GloVe and SSWE with the dimensionality configurations 50, 100 and 300. The quality of these representations was evaluated by the task of classifying the polarity of reviews using the IMDb database containing about of 25.000 movie reviews, comparisons occurred observing the results obtained from the neural network perceptron. The experiments showed that perceptron classifier with the method SSWE obtained the best average accuracy of 82.12% in the task of polarity classification.

Keywords: Word Embedding, Sentiment Analysis, Word2vec, LSA, GloVe, SSWE, machine learning.

Agradecimentos

Gostaria de agradecer a minha família por serem um pilar essencial ao incentivo aos meus estudos e momentos de felicidades. Gostaria de agradecer a instituição de ensino, funcionários e aos professores pela qualidade de ensino. Gostaria de agradecer ao orientador Cleber em que contribuiu na organização deste trabalho.

Gostaria de agradecer a todos os colegas do CIN pelos momentos de felicidades e dificuldades vividos, em especial a nossa turma 2014.2 por propiciar momentos únicos.

Sumário

1 Introdução

1.1 Objetivo.....	10
1.2 Organização do Trabalho.....	10

2 Análise de Sentimentos

2.1 Níveis de Análise de Sentimentos.....	12
2.2 Pré-processamento de Texto.....	12
2.2.1 Remoção Caracteres Especiais e Normalização.....	13
2.2.2 Tokenizer.....	13
2.2.3 Stemming.....	13
2.2.4 Remoção de Stop Words.....	14
2.3 Tipos de Abordagem Geração de Lista	14
2.3.1 Abordagem Manual.....	14
2.3.2 Abordagem Baseada em Dicionário	15
2.3.3 Abordagem Baseada no Corpus.....	15
2.4 Classificação na Análise de Sentimentos.....	16
2.4.1 Rede Neural Artificial	17

3 Word Embedding

3.1 LSA	19
3.1.1 Term Frequency-Inverse Document Frequency.....	20
3.1.2 Singular Value Decomposition.....	21
3.2 Word2Vec.....	21
3.2.1 Continuous Bag-of-Words	23
3.2.2 Skip-Gram.....	24
3.3 GloVe.....	25
3.2.1 Global Matrix Factorization.....	27
3.2.2 Local Context Window.....	27
3.4 Sentiment Specific Word Embedding.....	27
3.4.1 SSWEh.....	29
3.4.2 SSWEr.....	29
3.4.3 SSWEu.....	30

4 Experimentação

4.1 Configurações do Ambiente.....	31
4.2 Base de Dados.....	32
4.3 Metodologia.....	34
4.3.1 Pré-processamento IMDb	35
4.3.2 Configurações <i>Word Embeddings</i>	35
4.3.3 Mapeamento das Sentenças em Vetores.....	36
4.3.4 Avaliação.....	37
4.3.5 Classificador.....	38
4.4 Resultados.....	39

5 Conclusão

Lista de Figuras

2.1	<i>Tokenizer</i> aplicado em uma sentença.....	13
2.2	Arquitetura rede neural artificial com duas camadas escondidas.....	17
3.1	Embedding Space Representação em R^3	18
3.2	Representação da arquitetura CBOw.....	23
3.3	Representação da arquitetura Skip-Gram.....	24
3.4	GloVe Embedding Space avaliação grau de similaridade.....	26
4.1	Distribuição das reviews por número de palavras em trainLabeled.tsv.....	33
4.2	<i>Overview</i> Metodologia Experimentos.....	34
4.3	<i>Reviews</i> IMDb antes e depois do pré-processamento.....	35
4.4	Representação vetorial da <i>review</i> mapeada por LSA.....	36

Lista de Tabelas

2.1	Lista de palavras com seu peso sentimental.....	14
2.2	Abordagem baseado na lista para sentenças usando pesos tabela 2.1.....	16
3.1	Representação de co-ocorrência com TF.....	20
4.1	Configurações do sistema utilizado.....	31
4.2	Estrutura dos dados da IMDb.....	32
4.3	Frequência das palavras.....	34
4.4	Configurações <i>Word Embedding</i> baseados em Redes neurais.....	35
4.5	Matriz confusão classificador binário.....	37
4.6	Resultados das métricas <i>Word Embedding</i>	39
4.7	Tempo de treinamento dos modelos neurais de <i>word embedding</i>	40

INTRODUÇÃO

Word Embeddings são métodos que fornecem boas representações vetoriais contínuas de baixa dimensão para conjunto de textos não estruturados. Esses vetores são muito úteis, pois possuem um forte poder de generalização. Essas representações contribuem para uma melhora no desempenho em classificadores supervisionados em tarefas de processamento da linguagem natural (PLN) [15].

Muitas empresas usam opinião e sentimento como parte de suas pesquisas para tomadas de decisões, podemos observar essa preocupação em sistemas de feedbacks sobre *reviews* de produtos e coleta de mensagens em redes sociais para serem usados em algoritmos de publicidade [3]. A empresa Facebook publicou que no WhatsApp cerca de 65 bilhões de mensagens são enviadas diariamente [16], desse modo analisar todas estas mensagens é uma tarefa humanamente desgastante e custosa. Uma abordagem interessante para analisar o grande volume de textos é automatizar o processo, usando o método de mapeamento *word embedding* combinado com algum classificador do tipo rede neural. Estes modelos apresentam atualmente bons resultados em PLN [15]. Os métodos de *word embeddings* constroem eficientes vetores a partir de uma larga quantidade de textos. As características dos vetores são alta densidade e baixa dimensão, porque reduzem o custo computacional e funcionam melhores em redes neurais [15].

Além disto, o maior benefício de um modelo de *word embedding* é o seu forte poder de generalização conseguindo extrair informação semântica e sintática dos textos. Como exemplo os vetores das palavras “vermelho” e “azul” estão próximos, pois as duas palavras são referidas como cores. Dessa maneira, os modelos são capazes de extrair informação de contextos específicos independente da linguagem do texto.

O estudo foi delimitado na comparação destes métodos aplicados na análise de sentimento, especificamente na tarefa de classificação da polaridade de *reviews*. A metodologia adotada para comparar a qualidade dos *word embeddings* será através dos resultados alcançados no classificador perceptron. Os seguintes métodos foram selecionados. O método *Latent Semantic Analysis* (LSA) escolhido por ser um modelo antigo [12]. Foram escolhidos Word2vec [9] e Global Vectors for Word Representation (GloVe) [10]

por serem populares em PLN. Por fim, escolhido *Sentiment Specific Word Embedding* (SSWE) [11] devido aos seus resultados alcançados na área de análise de sentimento.

1.1 Objetivo

Observando os diferentes modelos existentes de *word embedding*, em diferentes trabalhos publicados, o objetivo da pesquisa é estudar, comparar e analisar os métodos de *word embedding* aplicados na análise de sentimentos. A comparação será realizada através da tarefa de classificação da polaridade, sendo esta última um dos problemas recorrentes na análise de sentimentos. Para avaliar estes diferentes métodos será utilizada a base de dados sobre *reviews* de filme da IMDb. A base de dados possui *reviews* acompanhados pela sua classe representada pelo *Score*, a pesquisa foi realizada como sendo um problema de classificação binária conforme sugerido no trabalho de Tang et al. [11].

No contexto deste trabalho de graduação foram avaliados os métodos Word2vec (CBOW), GloVe, LSA e SSWE. Portanto, o objetivo é expor e explicar quais dos *word embeddings* comparados obteve o melhor resultado sobre a base de dados escolhida na tarefa de análise de sentimentos sobre *reviews* usando o classificador perceptron.

1.2 Organização do trabalho

No capítulo 2 será explorada a área de análise de sentimento descrevendo alguns conceitos e técnicas aplicadas na área. No capítulo 3 *word embedding* e os métodos selecionados da pesquisa são apresentados. Capítulo 4 os experimentos e os resultados obtidos são descritos. No capítulo 5 finalizamos com a conclusão do trabalho.

2 Análise de Sentimentos

Análise de sentimento, ou mineração de sentimento, é um conjunto de técnicas que visa identificar a opinião expressa sobre um determinado objeto como *reviews*, mensagens, comentários e etc. Antigamente o principal recurso de acesso a informação era obtido através de amigos e veículos de comunicação, porém o advento da criação das redes sociais na Web alavancou a circulação mais livre das opiniões. Diferentes trabalhos vêm estudando sentimentos de usuários em redes sociais, trabalhos desde ações em bolsa de valores [1] à política [2]. Opinião é um bom indicador para representar e avaliar futuras ações humanas, pois o ser humano age muito baseado em experiências passadas que influenciam nas atuais tomadas de decisões.

Dessa maneira, tendo interesse em capturar essas informações importantes, não estruturadas, subjetivas e normalmente incompressíveis para linguagem de máquina surgiu à área de análise de sentimento, focada na detecção, reconhecimento de opiniões e avaliação dos comentários [5]. Uma tarefa típica da área é a classificação da polaridade, dada uma sentença classificá-la em negativa ou positiva [3]. Existem tarefas que possuem um maior número de classes a ser avaliado, conseqüentemente incrementando o grau de complexidade do problema. Pawel et al. 2017 em seu trabalho apresentou o estudo no uso de reconhecimento facial para prever sete estados das emoções humanas (medo, diversão, tristeza, surpresa, irá, neutro e desgosto) [4]. Assim notamos que a área não é restringida apenas a textos, porém neste trabalho atentará ao aspecto textual.

A análise opinativa é benéfica em nichos distintos, nas empresas podem aumentar o número de vendas e agilizar processos de negócio. Na visão dos consumidores em uma melhor qualidade do produto. Aos pesquisadores traz ganho de inúmeros estudos a serem feitos como melhor maneira de representar os dados, entendimento da subjetividade humana ao computador, administrar grande volume dos dados e identificar casos de usos em proveito da sociedade.

Além disso, diferentes tipos de granularidade são estabelecidos na área de análise de sentimentos para avaliação do texto desde macro (documentos) à micro (entidades). O uso de pré-processamento em textos geralmente torna os dados mais sólidos em processamento de linguagem natural [6], classificadores com aprendizagem supervisionada possuem atualmente os melhores desempenhos na área [15] e abordagens para geração de lista são

aplicadas para analisar textos [3]. Portanto, a área possui fundamentos obtidos através de estudos e existe interesse em produzir novas pesquisas para a sociedade.

2.1 Níveis de Análise de Sentimentos

Existem três níveis de granularidade pesquisados na análise de sentimento segundo Bing 2012 [3]. Os níveis mais tradicionais conhecidos de granularidade são nomeados em documento, sentença e entidade.

No nível de documento o conteúdo do documento é considerado uma unidade básica de informação sendo o todo classificado em positivo ou negativo. Dado um documento formado de um conjunto de *reviews* por diferentes usuários opinando sobre um determinado assunto único (exemplo, o livro Titanic) é definida a classe do documento.

A granularidade a nível sentença procura dividir frases que expressam opinião ou não. Em seguida existe a classificação das sentenças opinativas em sentenças positivas e negativas, a adição da classe neutra pode ser atribuída caso o problema seja estendido para as três classes neutra, negativa e positiva.

A granularidade de entidade é um tipo de classificação mais complexo, devido a sua natureza específica de identificação da entidade e sua classe. A classificação ocorre a nível de “Entidade”, pode-se exemplificar na seguinte sentença “A pintura do carro é elegante, contudo o som da buzina estava horrível” na sentença é identificado a entidade “carro” que possui 2 opiniões a respeito sobre seus componentes. A ideia é produzir a partir dos textos não estruturados uma lista de entidades e sumarizar a opinião a respeito dela.

2.2 Pré-processamento de Texto

Escolhida uma base textual a próxima etapa é realizar um pré-processamento. Realizar esta etapa normalmente minimiza as distorções provocadas pelo texto, conseqüentemente diminuindo a taxa de erro no classificador. Aplicar o pré-processamento na análise de sentimentos resulta em entradas mais sólidas e compreensíveis para o computador [6].

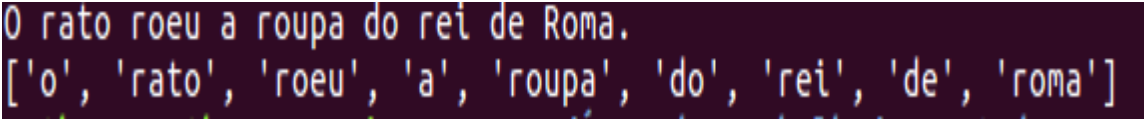
2.2.1 Remoção Caracteres Especiais e Normalização

Removemos caracteres especiais tais como pontuações, símbolos matemáticos e numéricos, pois não existindo um tratamento adequado para estes caracteres, criamos valores incompatíveis na representação vetorial para o modelo. A normalização textual é um processo que uniformiza um conjunto de palavras [6]. Erros de digitação como “poucoo” ou palavras contendo caracteres maiúsculos em início de frases “Pouco” são prejudiciais para a representação do modelo, porque o computador precisa compreender que estas palavras são a mesma palavra “pouco”, pois devemos manter a coerência na representação vetorial. Esse problema é reduzido com a normalização textual, pois o processo diminui o número de palavras distintas do conjunto. Usando as funções de *misspelling* e *lowercase* na normalização, a lista “poucoo”, “Pouco” e “poUco” é convergida para a mesma palavra “pouco”.

2.2.2 Tokenizer

O Processo visa transformar textos em *tokens* [6]. Em análise sentimento normalmente é considerado um *token* como sendo uma palavra, porém existem variações para definir o seu tamanho. Sendo uma estratégia simples separar palavras por espaços em brancos ou pontuações sendo os delimitadores. Na **figura 2.1** podemos visualizar o processo.

Figura 2.1 – *Tokenizer* aplicado em uma sentença



```
O rato roeu a roupa do rei de Roma.  
['o', 'rato', 'roeu', 'a', 'roupa', 'do', 'rei', 'de', 'roma']
```

Fonte: Autor

2.2.3 Stemming

Stemming é o procedimento que reduz palavras flexionadas ao tronco morfológico da palavra [6], como exemplo “precisamente” torna-se “precisa”, assim criando uma representação única baseado em um conjunto de palavras flexionadas. Este processo é dependente da estrutura morfológica do idioma devido à variação de regras.

2.2.4 Remoção de *Stop words*

As *stop words* são palavras muito comuns na linguagem [6], como exemplo para a língua portuguesa o artigo “a” e a preposição “de” são necessárias nas montagens de frases. Assim como as palavras flexionadas usadas no *stemming* a lista de *stop words* varia pela estrutura morfológica do idioma. Removemos os *tokens* com uma lista contendo as *stop words* a serem removidas, porque não possuem um forte valor sentimental. Existem casos fora da curva como o “não gosto” que quando removido “não” muda o sentido da palavra adjacente, porém para a maioria das situações é recomendado aplicar um filtro para as *stop words*.

2.3 Tipos de Abordagem Geração de Lista

Alguns tipos abordagens tradicionalmente usados em análise de sentimento são estabelecidos para formar o conjunto da polaridade das palavras em uma lista, sendo os tipos de abordagens conhecidos manual, baseado em dicionário e baseado no corpus [3]. Palavras “bom” e “ruim” possuem pesos opostos, pois os valores dos pesos são atribuídos de acordo com o valor sentimental, caso positivo +1 ou caso negativo -1. A lista serve como suporte para mapear palavras ou sentenças em uma representação vetorial para problemas de classificação de polaridade.

Tabela 2.1 - lista de palavras com seu peso sentimental

Palavra	bom	bonito	forte	ruim	feio	fraco
Peso	+1	+1	+1	-1	-1	-1

Fonte: Autor

2.3.1 Abordagem Manual

A abordagem manual consiste de criar uma lista de palavras que expressam sentimentos manualmente, etiquetando as palavras com valores positivo e negativos. Este método geralmente não é usado sozinho devido ao consumo de tempo para sua criação. Desse modo, duas abordagens automatizadas podem ser utilizadas em conjunto com está

técnica, porém tendo como *tradeoff* um aumento da taxa de erro a depender da qualidade das regras e dados.

2.3.2 Abordagem Baseada em Dicionário

A priori utilizamos a abordagem manual criando uma lista com as principais palavras polarizadas. Em seguida é usado um dicionário em que cada palavra contém uma lista com seus sinônimos, automaticamente os sinônimos daquela palavra são mapeados atribuindo o mesmo peso da palavra da lista. Em seguida é anexado a nova palavra a lista, o objetivo é expandir o vocabulário e generalizar a classe das palavras tendo o suporte um dicionário [3].

Contudo, os problemas desse tipo de abordagem incluem a dependência no dicionário que pode criar generalizações muito simplórias acarretando erros e não considera o contexto em que as palavras estão inseridas tendo em vista que a palavra “fria” pode variar “Comi a pizza fria.” é interpretada sendo ruim, porém “Bebi a Fanta fria.” algo positivo.

2.3.3 Abordagem Baseada no Corpus

Baseado no domínio do corpus descobre-se o sentimento da palavra em positivo e negativo. Esta abordagem é boa quando existe um corpus com qualidade possuindo um vocabulário grande e diversificado, pois será mais eficiente a captura do contexto. Uma ideia aplicada nesta abordagem é criar algumas generalizações pelo contexto. Como exemplo de generalização palavras adjacentes a conjunção “e” fornece uma ideia de continuidade. Deduzida que na frase “A mulher é bonita e elegante” os elementos de ligação de pares utilizando a conjunção “e” indicam que o par (bonita, elegante) é algo positivo, criando uma espécie de “consistência de sentimento” segundo Bing [3]. Contudo, o problema é que até mesmo no domínio do corpus palavras iguais podem diferenciar pelo seu contexto e regras.

2.4 Classificação na Análise de Sentimentos

Aprendizagem de máquina pode ser definida como a capacidade de um programa de computador melhorar seu desempenho em uma tarefa definida através de uma experiência prévia. Na aprendizagem de máquina supervisionada é aprendido um modelo a partir de dados de treinamento rotulados que nos permite fazer previsões mais precisas sobre dados desconhecidos. O classificador é uma implementação concreta do algoritmo de aprendizagem de máquina, dado um elemento, classifica-se sua classe. A classificação de polaridade da sentença é um tipo de problema de classificação.

Devido à limitação dos tradicionais classificadores de aprendizagem de máquina supervisionada não serem adaptados em interpretar diretamente entradas por texto, há uma conversão do texto não estruturado em uma representação vetorial, esta pode ser do tipo binária, inteira ou real [6]. Exemplificando a conversão textual utilizando a abordagem de lista manual “bom” e “ruim” na sentença, resulta em uma representação vetorial na **tabela 2.2**:

Tabela 2.2: Abordagem baseado na lista para sentenças usando pesos tabela 2.1

Sentença	bom	ruim
O bom homem.	1	0
O sabor estava ruim e o atendimento do restaurante era ruim.	0	-2

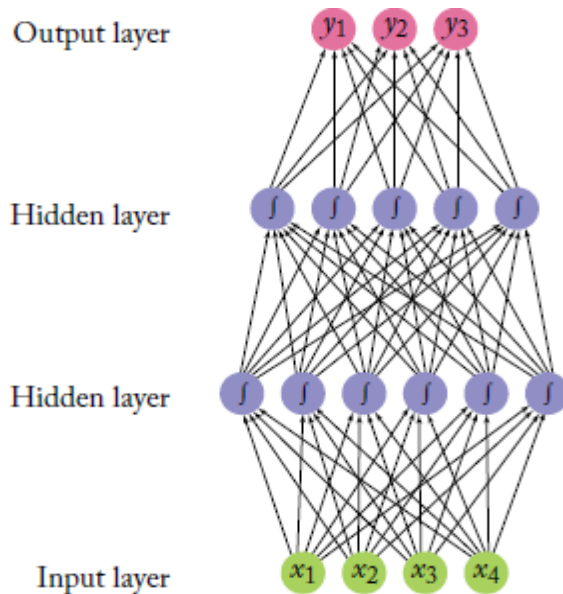
Fonte: Autor

As sentenças são representadas pelos vetores [1,0] e [0,-2]. Sendo estes vetores os padrões a serem classificados em positivo ou negativo. Os métodos de classificação supervisionados possuem destaque especial na classificação de texto pelo seu desempenho excepcional, principalmente os modelos de redes neurais que são os classificadores com os melhores resultados nos últimos anos em PLN [15].

2.4.1 Rede Neural Artificial

O modelo que vêm sido mais usado em problemas de pesquisa na análise de sentimentos é a rede neural artificial devido a novas descobertas [15]. A rede neural artificial é um modelo probabilístico, a ideia para a criação da rede foi baseada no funcionamento dos neurônios que transmitem informações quando certo tipo de sinal for atendido, provocando o repasse da informação [15]. De modo análogo este comportamento é reproduzido pela função de ativação em uma rede neural.

Figura 2.2: Arquitetura rede neural artificial com duas camadas escondidas



Fonte: [15]

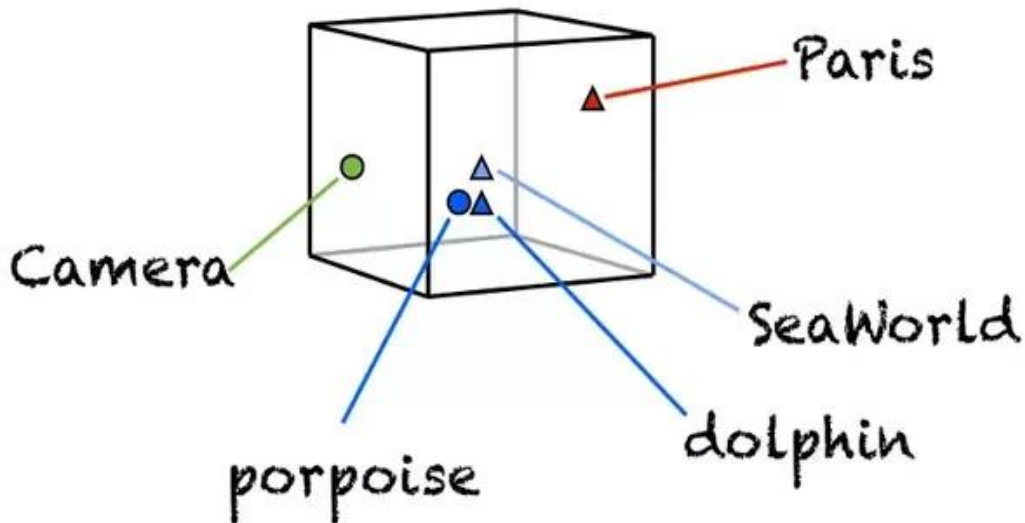
A camada de entrada será responsável por receber os padrões, estes são repassadas para que a camada oculta possa capturar/extrair a informação. A camada de saída exibirá a distribuição da probabilidade do padrão, podendo ser interpretado em um problema de classificação como sendo a classe pertence aquele padrão.

3 WORD EMBEDDING

Podemos definir *Word Embedding* como sendo um conjunto de técnicas que mapeia a semântica e sintática de uma linguagem natural em um espaço real utilizando estatísticas. Dessa forma, palavras de um conjunto de texto são mapeadas para vetores reais. O espaço destes vetores é denominado *embedding space* [15].

Palavras como “Paris” e “França” são mapeadas em vetores próximos devido ao seu grau de similaridade. Assim como sinônimos como “cão” e “cachorro” deveriam estar pouco distantes em uma boa representação de *word embedding*. Estes vetores podem ser utilizados como entrada para diferentes tipos de tarefas de processamento de linguagem natural tais como reconhecimento da linguagem natural, similaridade entre palavras [10], recuperação de informação [12], classificação de documentos, análise de sentimentos [11], *parsing* e etc.

Figura 3.1: Embedding Space representação em R^3



Fonte: <https://www.kaggle.com/sbongo/do-pretrained-embeddings-give-you-the-extra-edge>

Word Embedding é computado aplicando técnica de redução de dimensionalidade a matriz de co-ocorrência que é produzida a partir do texto. Existem diferentes abordagens

como a fatoração de matriz em *Latent Semantic Analysis* [12] ou através de redes neurais como apresentado em Word2vec [9].

Bengio et. al 2003 propuseram um dos primeiros modelos neurais de linguagem que inspirou diferentes arquiteturas. Dado um conjunto de palavras, predizer a probabilidade de uma próxima palavra ocorrer em determinada sentença. Modelos de *n-gram* eram sucedidos para realizar tal tarefa, porém observando o problema *curse of dimensionality* [7] outro tipo de abordagem deveria ser definido. Isto pelo fato de que o conjunto de combinações das palavras crescia exponencialmente prejudicando o tempo de execução significativamente, sendo o treinamento dos modelos com milhões de parâmetros um desafio importante a ser abordado [7].

Segundo Bengio et. al a “modelagem de variáveis contínuas, obtém uma generalização mais facilmente” [7]. Devido a isto, para enfrentar o problema de *curse of dimensionality* os autores propuseram generalizar a representação das palavras em vetores de dimensão n , sendo n a dimensão do *embedding space*. Essa representação foi combinada com uma função de probabilidade para palavras sequenciais criando um modelo de rede neural. Desse modo, ganhando como benefício o poder de representação de generalização dos vetores que irá associar melhor as palavras similares. Os vetores possuem como características alta densidade e baixa dimensão, porque reduzem o custo computacional e funcionam melhores em redes neurais [15].

3.1 Latent Semantic Analysis

Latent Semantic Analysis [12] é um método que procura ser eficiente para extrair e representar o contexto do texto com vetores de dimensões reduzidas. Esse é comumente usado na recuperação de informação. O método pode ser explicado como a combinação de *Term Frequency–Inverse Document Frequency* (TF-IDF) [13] e *Singular Value Decomposition* (SVD) [12], porém existem variações do LSA no uso de diferentes algoritmos para a criação da matriz de co-ocorrência.

Em Edgar et. al 2014 [14] foi testado a categorização semântica e similaridade entre palavras pares, no artigo é comparado os modelos LSA e Skip-Gram Word2Vec. Os autores constataram que para um conjunto pequeno de dados LSA obteve uma melhor performance que o Skip-Gram, sendo os benefícios de Skip-Gram serem notáveis apenas em conjuntos

iguais ou superiores a 10 milhões de palavras segundo os autores. Neste trabalho de graduação, descreve-se para primeira etapa o uso do TF-IDF para gera a matriz de co-ocorrência a partir dos documentos, em seguida para esta matriz é aplicado o SVD para criar um vetor reduzido em que consiste de uma representação dos documentos e termos representado pela matriz.

3.1.1 Term Frequency-Inverse Document Frequency

O *Term Frequency–Inverse Document Frequency* pode ser dividido em duas etapas, em TF o algoritmo procura contabilizar o número de palavras pela frequência normalizada, ou seja, podemos definir dado um conjunto de documentos:

Documento 1: Futebol Brasil

Documento 2: Alemanha e Brasil e Futebol

Sendo sua representação na matriz de termo de frequência como apresenta na **tabela 3.1**:

Tabela 3.1 - Representação de co-ocorrência com TF

Termo	Documento 1	Documento 2
Brasil	1	0,5
Alemanha	0	0,5
Futebol	1	0,5
E	0	1

Fonte: Autor

Resumidamente a fórmula do termo de uma frequência será:

$$f_{(t,d)} = \frac{f_{t,d}}{f_{t',d}} \quad (3.1)$$

Sendo as variáveis t o termo e d o documento, realizamos uma normalização dividindo pela frequência do t' o termo mais frequente encontrado no documento d . Em *Inverse Document Frequency* (IDF) procuramos diminuir a influência das palavras que ocorrem muito

frequentemente em documentos. Conseqüentemente melhorando o efeito na consulta para exibição de documentos mais relevantes, IDF pode ser calculado por:

$$\log(N/nt) \quad (3.2)$$

Sendo N o número total de documentos e nt o número de documentos com o termo. A formula final TF-IDF para a matriz de co-ocorrência será definida por:

$$f_{(t,d)} * \log \frac{N}{n_t} \quad (3.3)$$

3.1.2 Singular Value Decomposition

O *Singular Value Decomposition* é uma técnica algébrica que fatoriza uma dada matriz em três matrizes:

$$M = (K)*(S)*(D)^t \quad (3.4)$$

Sendo K representando por $m \times m$ matriz ortogonal, S pela matriz $m \times n$ matriz diagonal e D $n \times n$ matriz ortogonal de n , os valores m e n são dimensões da matriz. A matriz diagonal S possui valores onde os valores na diagonal são $[d_1 \dots d_r]$. Esses valores são chamados de *singular value* sendo o processo de decomposição da matriz M para S denominada *Singular Value Decomposition* [12].

LSA utiliza SVD para reduzir a dimensionalidade da matriz de co-ocorrência gerada por TF-IDF, construindo um vetor com menor dimensão representando por uma matriz fatorada S . Essa matriz preservar características do modelo anterior fazendo uma generalização. Contudo, perde algumas características da matriz original, porque em troca a um ganho computacional pela otimização por utilizar menos vetores comparada a matriz original.

3.2 Word2vec

Mikolov et. al 2013 [9] propuseram em seu trabalho duas diferentes arquiteturas para computar palavras em representação vetorial. O objetivo era criar modelos com redes

neurais que fossem treinados mais rapidamente e tivessem melhor acurácia em tarefas de processamento da linguagem natural. Em sua pesquisa enfatizou que *word embeddings* são capazes de extrair conhecimento semântico indo além da ideia de alguns outros pesquisadores que limitavam a análise na extração sintática. Podemos justificar o poder de embutir conhecimento semântico sobre vetores com a seguinte operação mostrada [9]:

$$V(\text{"rainha"}) \approx V(\text{"rei"}) - V(\text{"homem"}) + V(\text{"mulher"}) \quad (3.5)$$

Essa fórmula apresenta uma ideia em que o *Embedding Space* pode assimilar o contexto semântico, sendo V o vetor da palavra, pois consegue interpretar que algo próximo de rei e seja mulher, mas não seja homem, será mapeado para um vetor próximo ao de rainha.

Os modelos foram construídos para serem usados em bases de dados muito grandes contendo entre milhões à bilhões de palavras. Importante destacar que os modelos existentes na época não apresentaram boa performance de tempo na construção de vetores com dimensão 50 e 100 com uma grande base de treinamento. Por isso, os autores desenvolveram formas mais eficientes para computar.

Desse modo, os autores definiram uma forma geral para o custo da complexidade de uma arquitetura qualquer, chegaram à conclusão que o custo poderia ser deduzido como:

$$O = E * T * Q \quad (3.6)$$

Sendo E *epochs* do treinamento, T é o número de palavras no conjunto de treinamento e Q como o custo total da arquitetura escolhida [9]. Portanto, diminuir o custo significava uma simplificação na arquitetura em um modelo mais antigo estudado [7], removendo camadas intermediárias percebendo que “complexidade é causada pela camada oculta não-linear nesses modelos” [9] e “depende rigorosamente da normalização eficiente do *softmax*” [9].

Em Mikolov 2013 et. al os experimentos foram limitados a comparar apenas a representação de palavras aprendidas por redes neurais devido a limitações de performance e complexidade dos tipos de aprendizagem apresentados no treinamento com larga quantidade de dados. Segundo os autores “palavras podem ter múltiplos graus de similaridade” [9]. Devido a isto a comparação ocorreu através de diferentes tarefas com

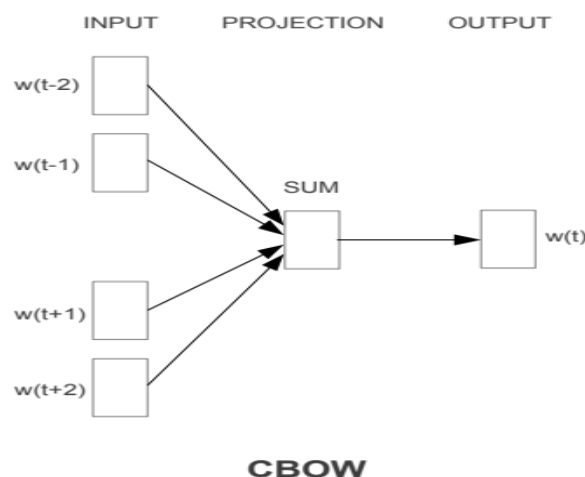
níveis de similaridade, exemplos de similaridade foram criados manualmente gerando uma enorme lista de pares a serem testados. Como exemplo os pares *big* e *bigger* sintaticamente correlacionadas, devem ser pouco distantes no espaço vetorial.

Nos resultados foram constatados que grande quantidade de dados produz vetores semânticos úteis para indicar a similaridade semântica entre duas palavras, por exemplo, França e Paris estão fortemente associado assim como Alemanha e Berlim [9]. Levando menos que um dia para criar vetores de qualidade com cerca de 1.6 bilhões de palavras. Os autores conseguiram aumentar a acurácia e diminuir o custo computacional do modelo.

3.2.1 Continuous Bag-of-Words

O *Continuous Bag-of-Words* (CBOW) [9] é uma rede neural que prediz a palavra dado um contexto, sendo o contexto interpretado como uma sentença. A rede neural proposta é similar a descrita em Bengio et al. [7], houveram alterações para otimizar o modelo, como a remoção da camada oculta e modificando a ligação na camada de projeção criando um canal compartilhado para a camada de saída. O resultado calculado na saída da projeção é a média de todos os vetores presentes no contexto da camada de entrada. Este modelo pode prever tanto palavras prévias como posteriores em um contexto.

Figura 3.2: Representação da arquitetura CBOW



Fonte: [9]

O custo do treinamento de CBOW pode ser dado pela fórmula:

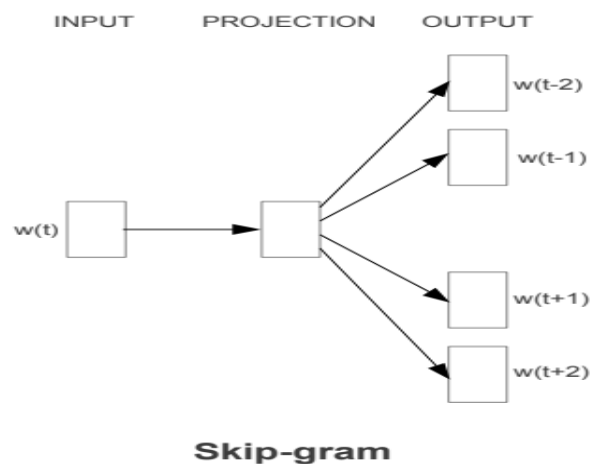
$$Q = N * D + D * \log_2(V) \quad (3.7)$$

V a quantidade de palavras distintas no vocabulário, N o tamanho da camada de projeção e D a dimensão dos vetores.

3.2.2 Skip-gram

O modelo *Skip-gram* desempenha uma função inversa do apresentado em CBOW, ou seja, dado uma palavra prediz calculando as palavras mais prováveis ao contexto.

figura 3.3: Representação da arquitetura Skip-gram



Fonte: [9]

O custo de treinamento para Skip-gram, no qual C é a distância máxima da palavra apresentada abaixo. Skip-Gram comparado a CBOW é mais custoso:

$$Q = C * (D + D * \log_2(V)) \quad (3.8)$$

3.3 GloVe

O *Global Vectors for Word Representation* [10] é um modelo global de regressão log-bilinear descrito como uma combinação dos pontos fortes dos métodos *global matrix factorization* e *local context window*. Os autores usaram propriedades de modelos existentes, pois estes segundo eles não estavam sendo aproveitados totalmente [10]. O 1º passo é definirmos os índices para a matriz de co-ocorrência, na representação de palavras, as primeiras entradas são geradas por estáticas na equação abaixo.

$$F(w_i, w_j, w_{k\sim}) = P_{ik}/P_{jk} \quad (3.9)$$

Sendo w_i , w_j e w_k palavras. P_{ik} e P_{jk} refere-se as probabilidades das palavras i e j aparecerem no contexto da palavra k .

$$F(w_i - w_j, w_{k\sim}) = P_{ik}/P_{jk} \quad (3.10)$$

Para diminuir o número do conjunto em F , encodamos a informação com a diferença dos vetores. Através de um conjunto de deduções os autores chegaram ao seguinte custo do GloVe:

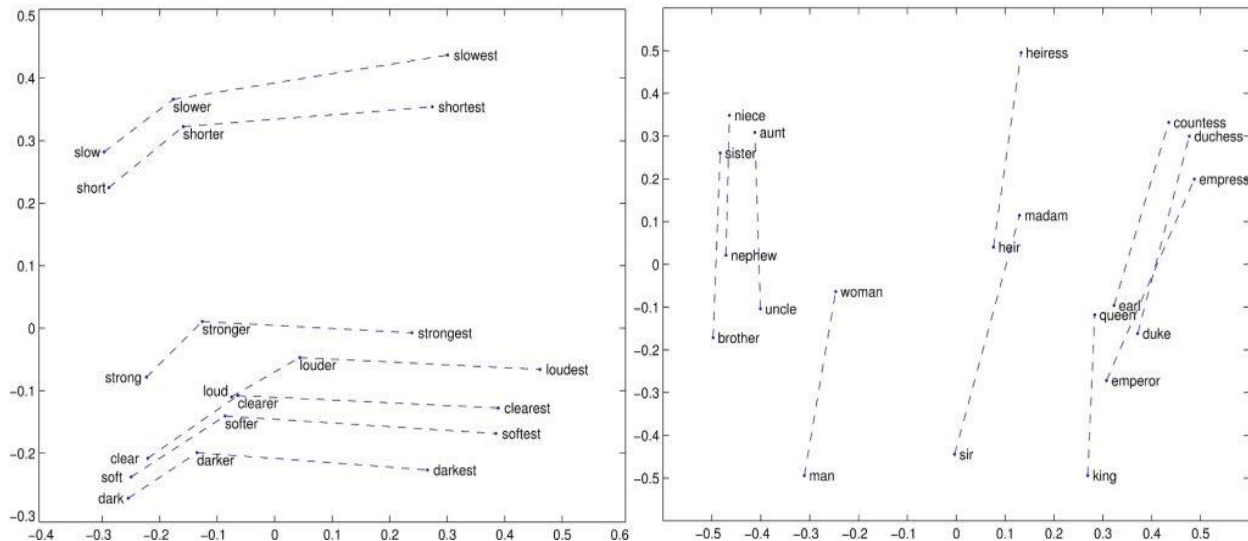
$$J = \sum_{i,j=1}^V f(X_{ij})(w_i^T \bar{w}_j + b_i + \bar{b}_j - \log X_{ij})^2 \quad (3.11)$$

Sendo V como o tamanho do vocabulário, f a função de peso, X_{ij} o número de vezes que a palavra i ocorre no contexto de j e b bias. Através de testes eles definiram os hiperparâmetros para otimizar o modelo como sendo $x_{max}=100$ usada na função de peso, $\alpha = \frac{3}{4}$ parâmetro da função de peso e AdaGrad para o treinamento. A complexidade do modelo depende da quantidade de zeros na entrada da matriz, tamanho da coleção de documentos, tamanho do vocabulário e o custo para gerar a matriz de co-ocorrência.

Em *Glove: Global Vectors for Word Representation 2014* [10] os experimentos ocorreram através de três tarefas distintas de processamento de linguagem natural para analogia das palavras, similaridade das palavras e reconhecimento do nome da entidade.

Analogia de palavras pode ser exemplificada semanticamente em responder a seguinte pergunta “Brasil está para Brasília assim como Alemanha está para?” e sintaticamente através de sufixos “Cachorro está para cachorrinho assim como gatinho está para?”. Calculada pela semelhança de cosseno.

Figura 3.4: Glove *Embedding Space* avaliação grau de similaridade



Fonte: <https://nlp.stanford.edu/projects/glove/>

Podemos observar pela **figura 3.4** o grau de similaridade do modelo em que palavras estão sintaticamente ligadas na esquerda e semanticamente na direita. Os modelos GloVe foram treinados com diferentes coleções de dados do wikipedia formando bilhões de *tokens*, existindo uma variação do conjunto pelo tamanho. Os dados passaram por um pré-processamento de texto e foram filtrados para um vocabulário com exatamente as 400,000 palavras mais frequentes. Sendo comparados com os trabalhos CBOW, SVD e Skip-Gram.

O modelo GloVe demonstrou uma melhor performance sobre os modelos comparados CBOW, SVD e Skip-Gram em todas as tarefas definidas de PLN nos experimentos, alcançando uma acurácia de 75% na tarefa de analogia das palavras. Para algumas das configurações dos experimentos GloVe demonstrou menor tempo de treinamento em relação ao Word2vec dependendo da configuração do corpus, vocabulário e dimensão do vetor.

3.3.1 Global Matrix Factorization

A vantagem da sua utilização é decompor largas matrizes em matrizes menores, dessa forma reduzindo complexidade. Este tipo de técnica também é utilizado no modelo *Latent Semantic Analysis* mostrado no trabalho [12]. Porém tendo como desvantagem baixa performance na tarefa de analogia de palavras e somente as palavras mais frequentes não é suficiente para calcular o grau de similaridade de forma satisfatória [10].

3.3.2 Local Context Window

Podemos aprender a representação das palavras usando palavras adjacentes. Este tipo de técnica também é utilizado no modelo Skip-Gram e CBOW apresentado no trabalho [7]. Contudo, tendo como limitação as operações aplicadas a nível local, não possui uma matriz de co-ocorrência do corpus do texto global, dessa forma não tira bom proveito da estatística, pois sofre com mínimos locais.

3.4 Sentiment Specific Word Embedding

Sentiment Specific Word Embedding (SSWE) [11] é um método de *word embedding* especialista na análise de sentimento codificando as palavras de forma que se possa extrair informação de sentimento. Os autores apresentaram em seu trabalho como um novo método para mapear palavras em uma representação contínua vetorial de forma a embutir informação de sentimento. Notando que os métodos existentes mapeavam palavras pelo contexto sintático em vetores contínuos, mas acabavam colocando palavras opostas como “bom” e “ruim” em vetores próximos, consequentemente resultando em uma perda semântica entre as palavras, o que acaba gerando uma perda de informação sobre os sentimentos no texto. Comparado com *word embeddings* tradicionais como Word2Vec ignoram a informação do sentimento nas sentenças. Segundos os autores “Os métodos tradicionais tipicamente modelam o contexto sintático das palavras, mas ignoram as informações de sentimento do texto” [11].

Em Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification 2014 [11] foi observando o uso crescente de pesquisas nos anos recente baseado em redes

sociais, a tarefa principal foi definida em avaliar a classificação da polaridade de dados extraídos do twitter, este conjunto de dados consiste de mensagens tendo como característica serem geralmente frases curtas limitadas a 140 caracteres que expressam uma opinião do usuário [11]. Sendo twitter uma rede social que permite enviar e receber mensagens torna uma boa fonte de referencial para capturar emoções humanas.

Para avaliar os diferentes modelos foram extraídos um conjunto massivo de mensagens sendo classificados através da técnica denominada *distant supervision* para gerar a base de treinamento e teste, sendo as entradas usadas no framework para classificação de sentimento com aprendizado não-supervisionado, dessa forma realizando a geração de *labels* através de um conjunto de dados original para a classificação de sentimentos no twitter. Este tipo de abordagem permite gerar uma maior base de dados, pois os dados serão etiquetados automaticamente, porém gera alguns ruídos através de falsos positivos.

As configurações utilizadas nos experimentos foram dimensão dos vetores de tamanho 50, camadas ocultas com tamanho 20 e *learning rate* de AdaGrad. Realizando a construção das *embedding* através de técnicas de *unigram*, *bigram* e *trigram* [7]. Utilizando mapeamentos como mínimo, média e máximo para construir para os vetores de entrada. Os vetores eram concatenados sendo as entradas representadas na seguinte forma:

$$v(tw) = [v_{\max}(tw), v_{\min}(tw), v_{\text{average}}(tw)] \quad (3.12)$$

Sendo a variável *tw* definida como a representação vetorial da sentença que serviria de entrada para o classificador. Nos resultados os autores perceberam que quanto maior (cerca de 12 milhões de tweets) o conjunto de treinamento para SSWEu os resultados obtidos ficavam mais estáveis devido ao ganho de vocabulário.

A acurácia do experimento SSWEu com a combinação unigram+bigram+trigram obteve uma acurácia de 84,98%. Dessa forma, concluiu que “O SSWEu aprende automaticamente recursos discriminativos a partir de tweets em massa e tem desempenho melhor comparado ao estado de arte” [11]. Sobressaindo ao modelo Word2vec que alcançou a precisão de 76,31% na tarefa de classificação da polaridade. Os modelos desenvolvidos são uma extensão do trabalho Collobert 2011, três redes neurais com diferentes estratégias foram desenvolvidas ao longo da pesquisa [11].

3.4.1 SSWEh

Uma abordagem não supervisionada baseado no modelo de Collobert. Esta solução prediz a distribuição de sentimento baseado na entrada do n -gram que representa um conjunto de palavras de tamanho fixo. Esse tipo de entrada é utilizado, pois o número de palavras em uma sentença é variável. O modelo desliza uma *window* de tamanho fixo gerando n -gram, desse modo a rede neural classifica a polaridade dos n -gram retirado das sentenças.

As camadas de saída de Collobert foram modificadas tendo o tamanho de neurônios alterado para K o número das *labels* de sentimento e adicionado uma camada de softmax acima. A função de *loss* de SSWEh pode ser descrita abaixo:

$$loss_h(t) = - \sum_{k=\{0,1\}} f_k^g(t) * \log(f_k^h(t)) \quad (3.12)$$

t a entrada, k o número da polaridade das *labels* de sentimento, $f^g(t)$ a distribuição de sentimento de ouro e $f^h(t)$ a distribuição de sentimento prevista. O modelo não gera n -grams corrompidos e os resultados das saídas são interpretados como [1,0] positivo e [0,1] negativo.

3.4.2 SSWEr

Este modelo é treinado para prever quando um n -gram é positivo e negativo. Podendo ser o resultado como $[Pp, Pn]$ ao qual uma distribuição como [0.7,0.3] ser interpretada como positiva e [0.2,0.8] como resultado negativo, Pp a probabilidade positiva e Pn a probabilidade negativa. Pode ser interpretado como uma rede mais simples que a SSWEh, pois não utiliza softmax em sua última camada, fazendo uso de apenas uma função de mapeamento descrita abaixo.

$$\delta_s(t) = \begin{cases} 1 & \text{if } f^g(t) = [1,0] \\ -1 & \text{if } f^g(t) = [0,1] \end{cases} \quad (3.13)$$

A função de *loss* SSWEr definida como:

$$loss_r(t) = \max(0, 1 - \delta_s(t)f_0^r(t) + \delta_s(t)f_1^r(t)) \quad \mathbf{(3.14)}$$

3.4.3 SSWEu

Observando que C&W extrai informação sintática, mas não extrai informação sobre sentimento e os novos métodos SSWEr e SSWEh apresentavam características inversas ao trabalho anterior. Observou que a solução seria combinar todas as técnicas gerando uma extração mais completa tanto na parte sintática e sentimento. Sendo a função *loss* a combinação das *loss* Collobert, *loss* SSWEr e *loss* SSWEh:

$$loss_u(t, t^r) = \alpha * loss_{cw}(t, t^r) + (1 - \alpha) * loss_{us}(t, t^r) \quad \mathbf{(3.15)}$$

4 EXPERIMENTAÇÃO

Neste capítulo descrevemos os experimentos realizados com os modelos de *word embeddings* em análise. Os vetores produzidos por LSA, Word2vec (CBOW), GloVe e SSWE são colocados como entrada em um perceptron para avaliar a qualidade dos vetores na análise de sentimentos. Os métodos são avaliados com base nas métricas extraídas do classificador descrito neste capítulo.

4.1 Configurações do Ambiente

As configurações da máquina utilizada são apresentadas na **tabela 4.1**, os experimentos foram desenvolvidos na linguagem de programação Python¹ usufruindo de suas bibliotecas.

Tabela 4.1 - Configurações do sistema utilizado

Sistema Operacional	Ubuntu 16.04.4
Memória Ram	12 GB
Processador	Intel(R) Core(TM) i5-4210U @ 1.70GHz
Tipo Sistema	64 Bits

Fonte: Autor

Python é uma linguagem de programação de alto nível lançada em 1991. Nos últimos anos está teve um ganho de popularidade, conquistando a quarta posição em 2018 de linguagem mais popular de acordo com o Índice TIOBE². Python tem como ponto forte possuir bons ecossistemas de bibliotecas *open source* para aprendizagem de máquina. As principais bibliotecas utilizadas neste trabalho são Keras³, Gensim⁴ e Scikit-Learn⁵, tornado uma linguagem completa para desenvolver e executar experimentos na área de inteligência artificial.

¹ <https://www.python.org/>

² <https://www.tiobe.com/tiobe-index/>

³ <https://keras.io/>

⁴ <https://scikit-learn.org/stable/>

⁵ <https://radimrehurek.com/gensim/>

Keras é uma API com foco em redes neurais oferecendo suporte a Python, desenvolvida com o foco em criar aplicações de aprendizagem de máquina rapidamente, deste jeito agilizando o processo de produção de pesquisa. Gensim é uma biblioteca desenvolvida em Python com implementações de modelagem semântica não supervisionada a partir de textos, contendo uma implementação de Word2Vec sendo eficiente robusta e grátis. Por fim, Scikit-Learn Ferramenta *open source* de python útil para tarefas em mineração de dados e análise de dados.

4.2 Base de Dados

O Kaggle⁶ é uma plataforma online que pertence ao Google que mantém uma comunidade de pessoas interessadas em aprendizagem de máquina e ciência dos dados. Esta disponibiliza uma grande gama de *datasets* públicos. Selecionou no Kaggle o *dataset* pertencente à Internet Movie Database⁷ (IMDb) possuindo cerca de 50.000 *reviews* no idioma inglês. Os dados estão divididos em dois arquivos *trainLabeled.tsv* e *test.tsv*, cada um deles contém 25.000 *reviews*. Nos experimentos será usado apenas o *train.tsv* devido a falta do rótulo *score* no arquivo *test.tsv* que determina a classe para validação, já que o Kaggle tem como característica manter *labels* da classe de testes privados em competições para evitar fraudes.

Tabela 4.2 - Estrutura dos dados da IMDb

Id	Review	score
11642_1	If you are in to bad movies for the entertainment of witnessing bad movies, bad acting, bad production etc... this, perhaps he may have been the next Spielberg in the making...	0
6793_10	This is the greatest movie if you want inspiration on following your heart and never giving up on your dream...	1

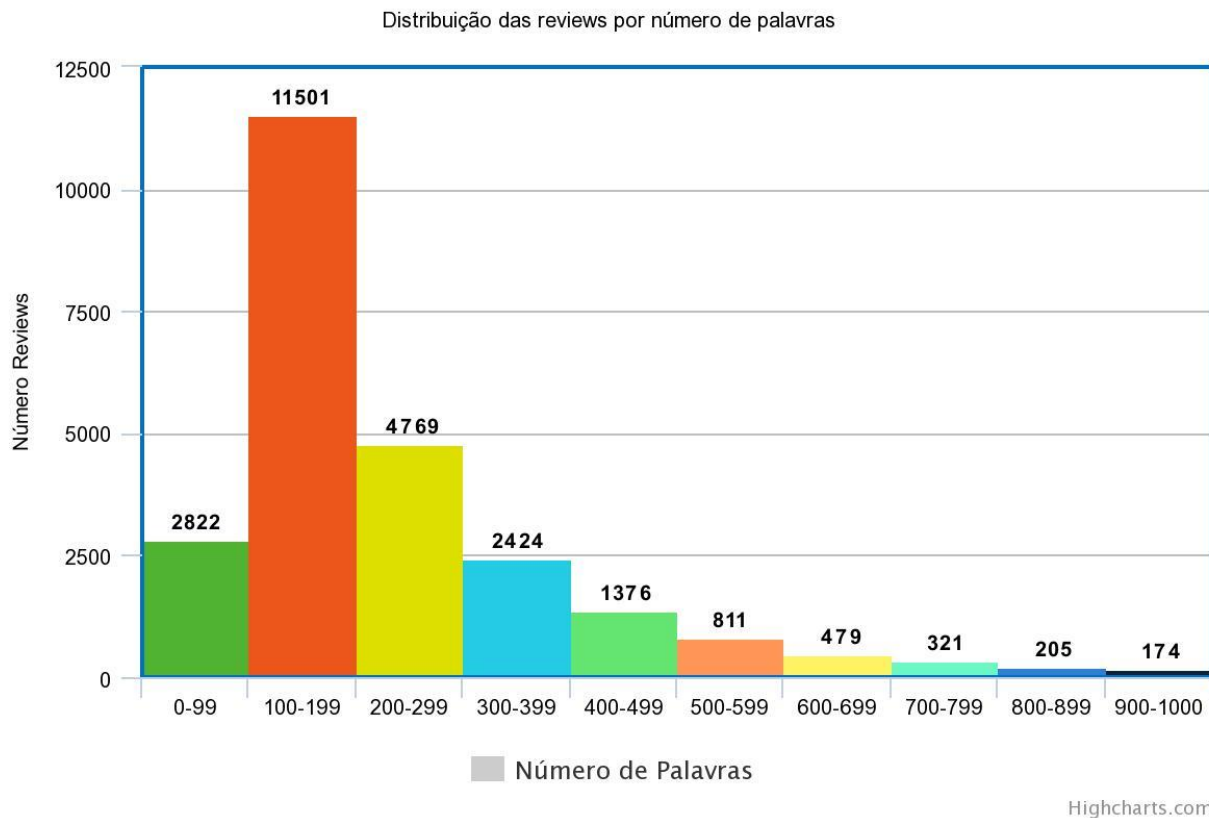
Fonte: Autor

⁶ <https://www.kaggle.com/>

Na **tabela 4.2** é observada a estrutura da base de dados com as colunas *id* o identificador único, *review* em que está contida a informação escritas dos usuários sobre os filmes e o *score* a classe pertencente. Filmes com nota igual ou maior a 7 foram etiquetadas pelo *score* 1 representando a categoria positiva e notas menores que 5 receberam *score* 0 representando a categoria negativa, *scores* neutros foram removidos para evitar ambiguidade. A *trainLabeled.tsv* está dividida em 12.500 *reviews* positivas e negativas, caracterizando como uma base de dados balanceada sendo um conjunto ideal para realizar experimentos de classificação.

O vocabulário contém em torno de 73.608 palavras únicas, a distribuição do tamanho das *reviews* por número de palavras é exibida no gráfico na **imagem 4.1**. Cerca de 11.501 *reviews* contêm entre 100 a 199 palavras.

Imagem 4.1: Distribuição das reviews por número de palavras em *trainLabeled.tsv*



Fonte: Autor

Em relação à quantidade de palavras nas *reviews*, a menor das *reviews* contém 12 palavras e a maior 2.470 palavras, a média para o conjunto é 234 palavras.

Tabela 4.3 - Frequência das palavras

	<i>Movie</i>	<i>Film</i>	<i>Story</i>	<i>Time</i>	<i>people</i>	<i>The</i>	<i>A</i>	<i>And</i>
Frequência	30856	27760	8732	7925	7663	320646	159273	158524

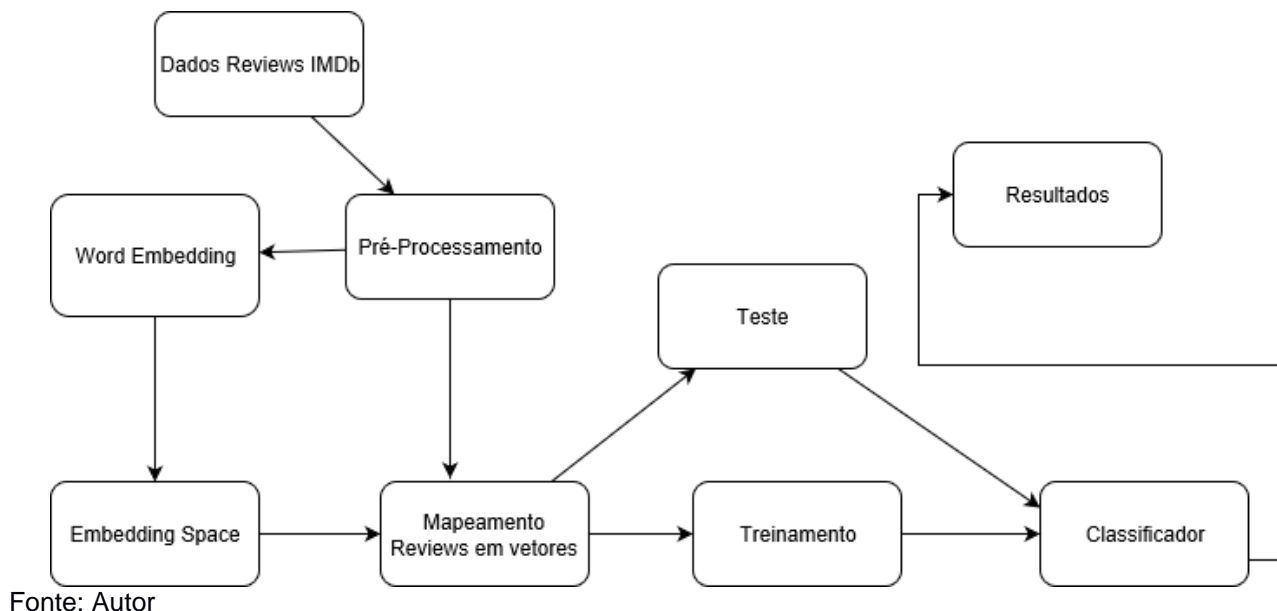
Fonte: Autor

Na **tabela 4.3** Os substantivos mais frequentes são *movie*, *film*, *story*, *time* e *people*. *Stop Words* como *the*, *a* e *and* são as três palavras mais frequentes da base IMDb.

4.3 Metodologia

Os diferentes modelos SSWE, GloVe, LSA e Word2vec (CBOW) são comparados pelos resultados das métricas do classificador binário perceptron, pois o problema é linearmente separável por apresentar duas classes de saída na tarefa de classificação de polaridade e devido as redes neurais extraírem o melhor potencial dos *word embeddings* [15]. A ideia é verificar a qualidade das entradas produzidas pelos *word embeddings* na rede neural. Podemos ver um panorama geral na **imagem 4.2** que será explicada ao longo das seções.

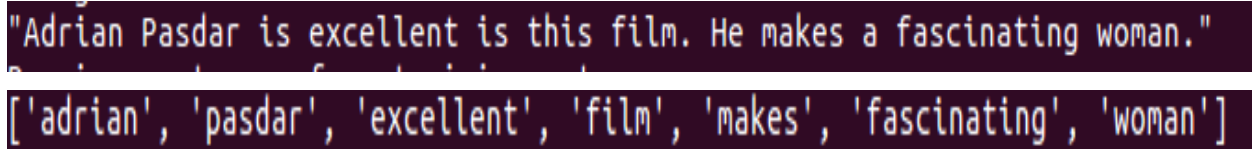
Imagem 4.2 – Overview Metodologia Experimentos



4.3.1 Pré-Processamento IMDb

Coletado o `trainLabeled.tsv` as *reviews* contidas no arquivo passaram por um pré-processamento prévio para melhorar os resultados na análise de sentimento. Foram retiradas *Tags html*, limpando o corpus. Em seguida foram removidos os símbolos que não representam letras, pois símbolos especiais ou números não serão úteis para a representação vetorial do modelo. Após esta etapa, as sentenças são normalizadas e *tokens* criados a partir das *reviews*. Concluindo com a remoção das *stop words* para diminuir as distorções nas representações vetoriais das sentenças. Visualizamos na **imagem 4.3** a transformação.

Imagem 4.3 - *Reviews* IMDb antes e depois do pré-processamento.



A imagem mostra duas linhas de texto em um fundo escuro. A primeira linha contém a frase: "Adrian Pasdar is excellent is this film. He makes a fascinating woman." A segunda linha contém a lista de tokens resultante: ['adrian', 'pasdar', 'excellent', 'film', 'makes', 'fascinating', 'woman']

Fonte: Autor

4.3.2 Configurações *Word Embeddings*

Os *word embeddings* baseado em redes neurais GloVe, a arquitetura Continuous Bag-of-Words apresentado em Word2Vec e SSWE, foram treinados com a base de dados da IMDb pré-processados para serem capazes de gerar o *Embedding Space* que consiste de um arquivo contendo uma lista de palavras e suas representações vetoriais. Sendo os parâmetros para o treinamento das redes neurais configurados na **tabela 4.4**.

Tabela 4.4 - Configurações *Word Embedding* baseados em Redes neurais

Base Dados	<i>Epochs</i>	Janela de Contexto	Filtro Palavras
IMDB	10	15	5

Fonte: Autor

Os modelos GloVe e CBOW no treinamento receberam todo o conteúdo textual das *reviews* em `trainLabeled.tsv`. Em *Sentiment Specific Word Embedding* foi treinado apenas

com 75% dos dados de trainLabeled.tsv, pois sendo um modelo que usufrui do rótulo da classe devemos limitar o seu conjunto de treinamento.

O LSA avaliado usa TF-IDF na matriz de co-ocorrência e não seguirá os padrões de configurações das redes neurais devido à sua natureza diferenciada para a criação da representação vetorial pulando para a etapa de mapeamento de *reviews*.

4.3.3 Mapeamento das Reviews em Vetores

Estabeleceu o mapeamento das sentenças na seguinte forma, cada *embedding space* representando por um arquivo contém uma representação única da palavra como vetor com a dimensionalidade configurada, sendo a *review* mapeada utilizando a concatenação do vetor mínimo, máximo e a média dos vetores contido nas sentenças como visto em [11].

$$V(R) = [\text{MAX}(R), \text{MIN}(R), \text{MÉDIA}(R)] \quad (4.1)$$

A exceção à regra para o mapeamento será em LSA, porque este mapeia a matriz de co-ocorrência da *review* diretamente usando SVD, obtendo o vetor representativo para ser usado como entrada no classificador.

Imagem 4.4 - Representação vetorial da *review* mapeada por LSA

```
[ 0.37460157 -0.09350724 0.15998079 -0.04547954 0.146719 -0.00888241
 0.03674672 0.02273276 0.03113756 -0.06891885 -0.03054728 -0.03104694
 -0.04058651 -0.01814635 0.08631623 0.01499513 0.00917019 -0.00686307
 -0.09106064 0.01138302 0.03311621 -0.05317519 0.03034381 0.00903781
 0.09217778 0.00083875 0.03158967 0.04499194 0.06138179 0.01141496
 -0.01359259 -0.00248063 0.02614738 -0.02748059 0.04838341 0.02026636
 0.05037867 -0.00829348 0.00100771 0.0550167 -0.05193633 0.04705608
 0.01156533 0.08829045 -0.00352982 -0.05470402 0.03527677 -0.07453959
 -0.0244757 -0.04414392]
```

Fonte: Autor

4.3.4 Avaliação

O desempenho é avaliado com as métricas precisão, revocação, F-measure e acurácia para a completude dos resultados. Definida a matriz de confusão na **tabela 4.5** tendo as variáveis para suportar os cálculos das métricas.

Tabela 4.5 - Matriz confusão classificador binário

	Predita Positivo	Predita Negativo
Classe Negativo	FN	VN
Classe Positivo	VP	FP

Fonte: Autor

Os conceitos relacionados à matriz de confusão são:

- VP (Verdadeiro Positivo): Quantidade de exemplos positivo predito corretamente;
- FP (Falso Positivo): Quantidade de exemplos positivo predito erroneamente;
- VN (Verdadeiro Negativo): Quantidade de exemplos negativo predito corretamente;
- FN (Falso Negativo): Quantidade de exemplos negativo predito erroneamente.

As seguintes métricas serão usadas para avaliação, sendo a precisão indicadora da corretude por categoria.

$$precisão = \frac{VP}{VP+FP} \quad (4.2)$$

Revocação indica a frequência de relevância dos resultados preditos.

$$revocação = \frac{VP}{VP+FN} \quad (4.3)$$

F-measure é a média harmônica ponderada entre precisão e revocação, medida útil em identificar um equilíbrio garantido consistência, calculados na seguinte forma.

$$F\text{-measure} = 2 \times \frac{precisão \times revocação}{precisão + revocação} \quad (4.4)$$

Acurácia é computado como a porcentagem de acerto para todas as classes, computado com.

$$acurácia = \frac{VP+VN}{VP+FN+VN+FP} \quad (4.5)$$

Os resultados das métricas são probabilidades no alcance entre 0 e 1, sendo os valores representado por porcentagens nas tabelas. Adotou-se a visão Macro sendo a média geral das classes locais.

4.3.5 Classificador

Tendo os dados da IMDb mapeados em uma representação vetorial das sentenças. Nesta etapa os 25.000 vetores produzidos são divididos na proporção 75% treinamento e 25% teste para serem usados no classificador. O classificador dos experimentos é uma rede neural do tipo perceptron. A camada de entrada possui o tamanho do vetor representativo da *review* e na camada de saída um neurônio para indicar a distribuição da probabilidade, foi usada como função de ativação Softmax por apresentar bons resultados na área de processamento de linguagem natural [15]. Finalizando com a função ReLu para regularizar os resultados sendo o inteiro 0 negativo e 1 positivo.

$$\text{ReLu}(\text{Softmax}(V(R))) \quad (4.6)$$

Sendo o classificador treinado por dez *epochs*, porque devemos generalizar rapidamente para garantir a eficácia do modelo em base de dados gigantes para economizar recursos financeiros. Para cada método de *word embedding* com diferentes dimensionalidades foi repetida a execução 5 vezes é calculada suas médias e desvios padrões.

4.4 Resultados

A avaliação ocorreu com o conjunto de testes (6.250 *reviews*) no classificador apresentado em 4.3.5, as métricas coletadas para comparar são precisão, revocação, F-measure e acurácia. As médias e os desvios padrões dos resultados são mostrados em percentuais na **Tabela 4.6**.

Tabela 4.6 - Resultados das métricas *Word Embedding*

Dimensão	LSA			Word2vec(CBOW)			GloVe			SSWE		
	50	100	300	50	100	300	50	100	300	50	100	300
Precisão	73,88 ±2,17	75,19 ±0,94	76,71 ±0,70	78,10 ±0,39	78,99 ±0,30	79,83 ±0,46	80,31 ±0,10	81,09 ±0,20	81,46 ±0,50	81,39 ±0,39	81,89 ±0,81	82,79 ±0,30
Revocação	73,51 ±2,14	74,42 ±1,24	76,11 ±1,36	77,72 ±0,75	78,63 ±0,62	79,39 ±1,14	79,68 ±0,26	80,92 ±0,26	81,31 ±0,57	81,07 ±0,61	81,40 ±0,81	82,13 ±0,79
F-measure	74,81 ±2,05	76,10 ±1,05	77,30 ±0,53	77,52 ±0,75	78,76 ±0,60	78,63 ±2,58	80,87 ±0,08	81,15 ±0,39	81,22 ±0,55	80,93 ±1,55	81,07 ±1,68	82,17 ±1,59
Acurácia	73,45 ±2,14	74,34 ±1,26	76,05 ±1,40	77,72 ±0,81	78,62 ±0,67	79,41 ±1,08	79,61 ±0,27	80,90 ±0,29	81,30 ±0,60	81,07 ±0,58	81,41 ±0,48	82,12 ±0,79

Fonte: Autor

O Aumento da dimensão dos vetores das palavras e conseqüentemente das *reviews* resultou em um ganho nos percentuais das métricas para todos os modelos treinados na base IMDb, porque estas representações continuaram a extrair das *reviews* características relevantes com o ganho das dimensionalidades. À visto disso, a configuração de *embedding space* com a dimensionalidade 300 superou 100 e 50 para todos os modelos nesta base e tarefa de classificação de polaridade usando o classificador perceptron.

Baseado nos resultados dos experimentos a configuração com o método LSA apresentou o pior resultado, mesmo a configuração com 300 vetores não conseguiu superar nenhum outro modelo de predição com dimensionalidade 50. O mapeamento com apenas SVD em LSA comparando aos outros métodos demonstrou ser ineficiente, tendo como explicação que a fatoração da matriz não foi capaz de criar uma boa representação vetorial

da *review* para a rede neural perceptron. Outra justificativa é uma menor quantidade de vetores no mapeamento para representar a *review*, conseqüentemente sendo o pior método com as configurações trabalhadas.

GloVe sobressaiu a Word2vec (CBOW) na análise de sentimentos, tem como explicação que o método aproveita das boas características encontradas em LSA evitando mínimos locais e Word2vec tendo boa performance na analogia das palavras. Todos os métodos de mapeamento apresentaram uma boa absorção do contexto específico contribuindo em uma média superior a 73% de acurácia para o classificador, possuindo um bom poder de generalização. Os modelos demonstraram serem estáveis em F-measure, pois apresentaram um equilíbrio em precisão e revocação.

O destaque está em SSWE sobressaindo às métricas avaliadas, com uma dimensionalidade 300 foi capaz de superar todos os outros métodos. O motivo da melhora do desempenho no classificador é explicado devido a característica em absorver o conhecimento do sentimento das palavras em contextos específicos, mapeando palavras pelo seu valor sentimental, o que faz palavras opostas como “bom” e “ruim” ficarem distante entre si. Contudo, a diferença dos resultados em SSWE não foi tão discrepante em relação ao GloVe. Em relação ao tempo de treinamento podemos observar na **tabela 4.7**:

Tabela 4.7- Tempo de treinamento dos modelos neurais de *word embedding*

	Word2Vec(CBOW)			GloVe			SSWE		
	50	100	300	50	100	300	50	100	300
Dimensão	50	100	300	50	100	300	50	100	300
Tempo em minuto(s)	1	2	6	2	5	12	5	9	28

Fonte: Autor

Existe uma relação em que quanto maior a dimensão maior a quantidade do tempo, causando um aumento no custo. Uma das desvantagens percebida em SSWE foi o seu maior tempo de treinamento comparado a GloVe e Word2vec (CBOW). CBOW demonstrou ser o mais rápido em questão de treinamento dos modelos neurais gastando cerca de 1/5 do tempo de treinamento do SSWE. Outra desvantagem em SSWE é a necessidade de dados etiquetados prévios, enquanto outros métodos não contavam com este tipo de artifício. Contudo, *Sentiment Specific Word Embedding* com dimensão 300 obteve no classificador melhor resultado médio com 82,12% de acurácia.

5 Conclusão

Neste trabalho, as áreas de análise de sentimento e os métodos de *word embedding* foram estudadas e exploradas. O objetivo de combinar estas áreas é obter modelos otimizados capazes de automaticamente extrair informações de sentimentos em grandes bases de textos. Os métodos foram comparados pela base de dados da IMDb.

Nos experimentos foi observado que *word embedding* é capaz de capturar informações baseadas em contextos próprios, tornando-os interessante para serem aplicados em situações adversas. Os resultados obtidos no classificador com as palavras com dimensionalidade 300 mostraram serem melhores resultados quando comparados as dimensões 50 e 100, porém aumentar a dimensionalidade significa aumentar o custo total do modelo. Comparados os diferentes métodos LSA, SSWE, CBOW e GloVe pela qualidade das métricas coletadas da rede neural perceptron. O método com o melhor resultado foi *Sentiment Specific Word Embedding* usado no classificador perceptron obtendo uma acurácia média de 82,12%. Esse método de *word embedding* tem bom potencial na análise de sentimento devido ao refinamento feito no trabalho para absorver a informação de sentimento [11], porém tendo o maior custo no treinamento.

Portanto, a pesquisa evidencia que o uso de SSWE contribui para uma melhora na tarefa de classificação da polaridade quando comparando aos outros modelos LSA, Word2vec (CBOW) e GloVe. Em trabalhos futuros, seria interessante estender o número de métodos de *word embedding* comparados, diversificar os métodos de mapeamento, adicionar novos classificadores e experimentar os resultados em uma base com milhões de *reviews* em português. Destarte, a mercê dos resultados pode ser encontrada em pesquisas futuras quais dos modelos para a linguagem em português foi melhor ou criar um novo método de *word embedding* mais eficiente para o idioma estudado.

REFERÊNCIAS BIBLIOGRÁFICAS

- [1] Eric D. Brown .August (2012).**Bulls, Bears...and Birds? Studying the Correlation between Twitter Sentiment and the S&P500.** Em Research Gate.
- [2] Andranik Tumasjan, Timm O. Sprenger, Philipp G. Sandner, Isabell M. Welp (2010).**Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment.** Em AAAI Publications, Fourth International AAAI Conference on Weblogs and Social Media.
- [3] Bing Liu (2012). **Sentiment Analysis and Opinion Mining.**
- [4] Paweł Tarnowski, Marcin Kołodziej, Andrzej Majkowski, Remigiusz J. Ra.(2017).**Emotion recognition using facial expressions. International Conference on Computational Science.** Em ICCS 2017, 12-14 June 2017,Zurich, Switzerland. Procedia Computer Science 108C (2017) 1175–1184
- [5] Erik Cambria, Björn Schullerm,Yunqing Xia e Catherine Havasi.(2013).**New Avenues in Opinion Mining and Sentiment Analysis.** Em IEEE Intelligent Systems (Volume: 28 , Issue: 2 , March-April 2013).
- [6] Dursun Delen, Robert Nisbet, Thomas Hill, Andrew Fast, John Elder, Gary Miner(2012).**Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications.**
- [7] Bengio, Y., Ducharme, R., Vincent, P., & Janvin, C. (2003). **A Neural Probabilistic Language Model.** Em The Journal of Machine Learning Research, 3, 1137–1155.
- [8] Collobert, R., & Weston, J. (2008). **A unified architecture for natural language processing.** Em Proceedings of the 25th International Conference on Machine Learning – ICML '08, 20(1), 160–167.
- [9] Mikolov, T., Corrado, G., Chen, K., & Dean, J. (2013). **Efficient Estimation of Word Representations in Vector Space.** Em Proceedings of the International Conference on Learning Representations (ICLR 2013)
- [10] Pennington J, Socher R, Manning CD (2014). **Glove: Global Vectors for Word Representation.** Em EMNLP. 2014;14:1532-1543.
- [11] Duyu Tang, Furu Wei, Nan Yang , Ming Zhou , Ting Liu , Bing Qin (2014).**Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification.**
- [12] Dumais, S.T. , Furnas , G.W. , Landauer, T.K.,Deerwester,S.,e Harshman,R.(1988). **Using latent semantic analysis to improve access to textual information.** Em Proceedings of the SIGCHI conference on Human factors in computing systems, pages 281-285. ACM.
- [13] Scott Deerwester , Susan T. Dumais, George W. Furnas , Thomas K. Landaue e Richard Harshman(1990). **Indexing by latent semantic analysis.** Em Journal of the America Society for information Science, 41.

[14] Edgar Altszyler, Mariano Sigman, Sidarta Ribeiro, Diego Fernández Slezak(2014).**Comparative study of LSA vs Word2vec embeddings in small corpora: a case study in dreams database.** Em ACL 1:238-247.

[15] Yoav Goldberg (2017). **Neural Network Methods for Natural Language Processing.**

[16] Prachi Bhardwaj (2018). **The number of messages sent via WhatsApp each day has tripled since Facebook bought it four years ago.** Disponível Em: <<https://www.businessinsider.co.za/whatsapp-messages-tripled-facebook-acquisition-charts-2018-5/>>. Acesso em: 05 dez. 2018.