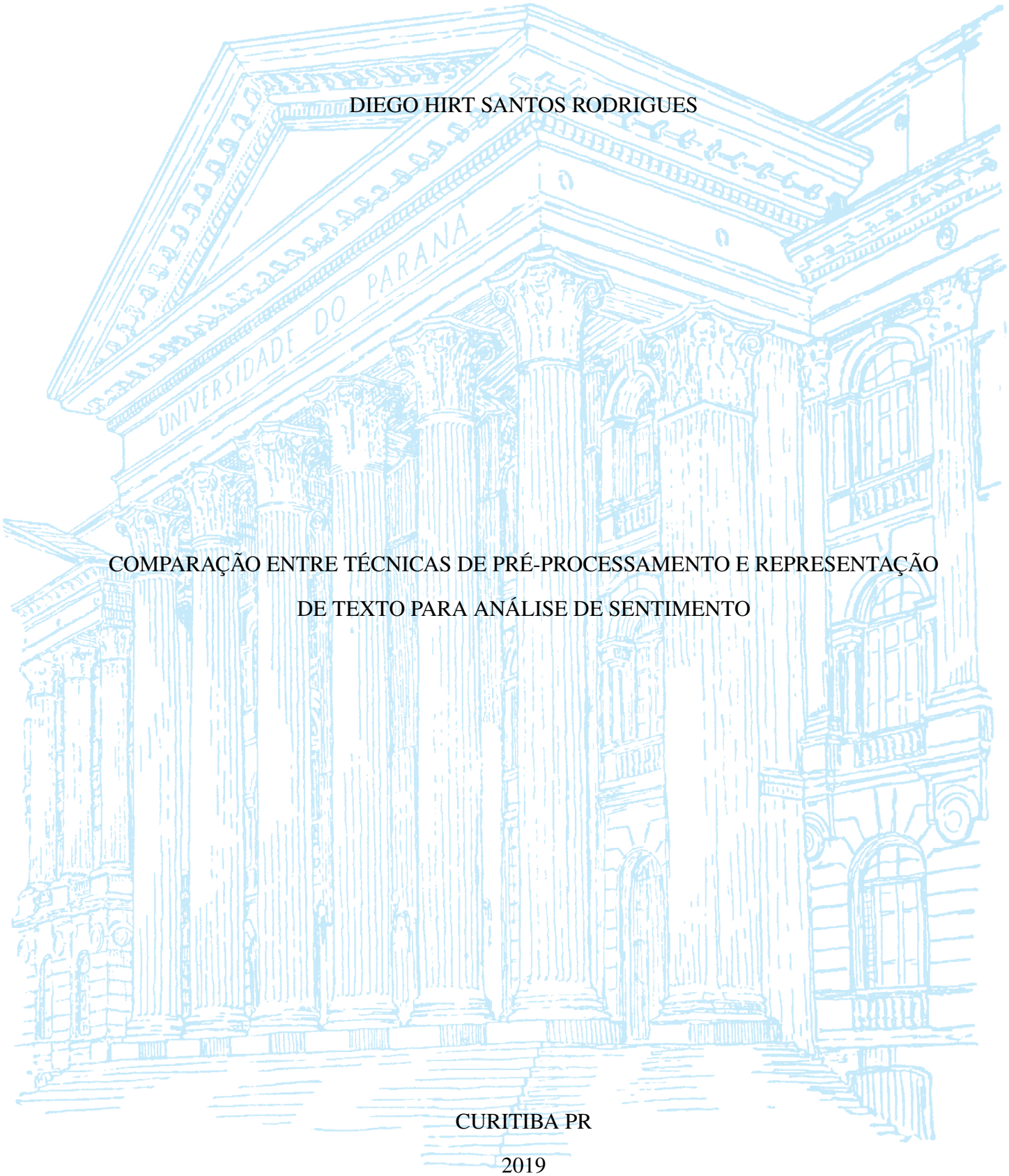
UNIVERSIDADE FEDERAL DO PARANÁ

DIEGO HIRT SANTOS RODRIGUES



COMPARAÇÃO ENTRE TÉCNICAS DE PRÉ-PROCESSAMENTO E REPRESENTAÇÃO DE TEXTO PARA ANÁLISE DE SENTIMENTO

CURITIBA PR

2019

DIEGO HIRT SANTOS RODRIGUES

COMPARAÇÃO ENTRE TÉCNICAS DE PRÉ-PROCESSAMENTO E REPRESENTAÇÃO DE TEXTO PARA ANÁLISE DE SENTIMENTO

Trabalho apresentado como requisito parcial à conclusão do Curso de Bacharelado em Ciência da Computação, Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: Eduardo Jaques Spinosa.

CURITIBA PR

2019

AGRADECIMENTOS

Agradeço aos meus pais por todo incentivo e consideração durante todos esses anos. A minha família, especialmente aos meus tios e avós por todo suporte e apoio. Ao professor Eduardo J. Spinosa por toda orientação, paciência e incentivo. Aos meus colegas de turma especialmente Lucas Ikeda, Caio Rodrigues e Rafael Castro, por toda ajuda durante o curso.

RESUMO

Com o surgimento das redes sociais muitas pessoas expressão suas opiniões e visões sobre diversos assuntos. Com o crescimento dessas opiniões disponíveis na internet, criou-se um interesse para estudar e processar essas opiniões. Esse sub-problema do processamento de linguagem natural é conhecido como análise de sentimento. Este trabalho tem como objetivo analisar o efeito de diferentes técnicas de pré processamento e representações para o problema de análise de sentimento.

Palavras-chave: PLN. Análise de Sentimento. Pré Processamento de Texto. Representação de Texto.

ABSTRACT

With the emergence of social media, many people are expressing their opinions and views about several topics.With the growth of these opinions available on the internet, an interest was created to study and process this opinions. This subproblem of natural language processing is known as sentiment analysis. This work aims to analyze the effect of different preprocessing techniques and different text representation for the sentiment analysis problem

Keywords: NLP. Sentiment Analysis. Text Preprocessing. Text Representation.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ACRÔNIMOS

PLN	Processamento de Linguagem Natural
BoW	Bag of Words
Semeval	Semantic Evaluation
NLTK	Natural Language Toolkit
RE	Regular expression operations

SUMÁRIO

1 INTRODUÇÃO

Muitas das pesquisas existentes sobre processamento de texto é focada em mineração e recuperação de informações factuais, classificação de texto, agrupamento de texto e mineração de textos e tarefas de processamento de linguagem natural. Poucos trabalhos foram realizados sobre o processamento de opiniões até recentemente. No entanto, opiniões são tão importantes que quando necessitamos tomar alguma decisão nós queremos a opinião de outras pessoas. Isso é verdade não apenas para indivíduos mas também para organizações. A internet mudou drasticamente a forma que as pessoas expressam suas opiniões. Atualmente elas podem postar análises de produtos no site do vendedor, expressar suas visões sobre quase qualquer assunto em forúms, grupos de discussão, e blogs, que são chamados coletivamente de conteúdo gerado por usuários (Indurkhya e Damerau, 2010).

Pré processamento e representação do texto, são duas etapas importantes do problema da análise de sentimento.Esse trabalho tem como objetivo analisar o impacto de diferentes técnicas de pré processamento e diferentes representações.

O trabalho é dividido em 5 capítulos. O capítulo 2 conceitua o processamento de linguagem natural e seu domínio. O capítulo 3 apresenta a fundamentação teórica necessária para compreender o trabalho. O capítulo 4 apresenta os experimentos realizados bem como os resultados obtidos. Finalizando no capítulo 5 com as conclusões finais sobre o tema.

2 PROCESSAMENTO DE LINGUAGEM NATURAL

Processamento de linguagem natural (PLN) é um ramo da inteligência artificial que estuda como computadores podem analisar e processar a linguagem natural.

É um conjunto de técnicas computacionais para analisar e representar textos em um ou mais níveis de análise linguística com o propósito de obter o processamento de linguagem semelhante ao processo humano para uma série de tarefas ou aplicações (Liddy, 2001).

2.1 NÍVEIS DE LINGUAGEM

Linguagem é um sistema de comunicação. A linguagem humana é composta por um conjunto de símbolos e de regras, onde símbolos são combinados para transmitir informações e regras dão ordem para esses símbolos.

Para um conjunto de símbolos terem um sentido, existe um conjunto de níveis. Para cada nível extrai-se um significado, sendo que os humanos utilizam de todos esses níveis para ter uma compreensão dos símbolos. Quanto mais níveis um sistema PLN utilizar, mais próximo de um entendimento humano ele terá. Um conceito importante é a análise semântica, que é o processo de reconhecer os possíveis significados de uma sentença.

2.2 CLASSIFICAÇÃO DE TEXTO

Classificação de texto é um subproblema de PLN, que, dado um texto e um conjunto de categorias, determina a qual categoria o texto pertence. Existem diversas aplicações para classificação de texto, como determinar se um e-mail é spam ou, categorizar um artigo jornalístico, bem como realizar uma análise de sentimento, dentre outros exemplos. A figura 2.1 mostra um exemplo de categorizar um artigo jornalístico.

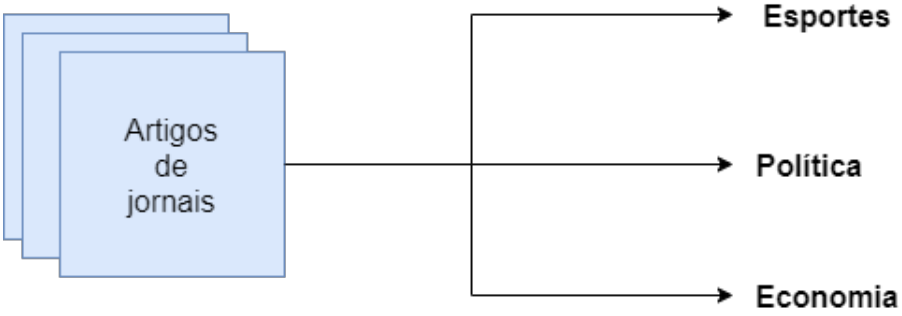


Figura 2.1: Classificação de Texto

3 ANÁLISE DE SENTIMENTO

Análise de sentimento é o estudo das opiniões, sentimentos e emoções expressadas em um texto (Indurkhya e Damerau, 2010). O objetivo da análise de sentimento é estimar a polaridade do sentimento de um texto, baseado em seu conteúdo. A polaridade do sentimento pode ser representada como positivo, negativo e neutro, dependendo da escala utilizada. Diversas etapas compõem o fluxo da análise de sentimento. Primeiro é necessário coletar um conjunto de textos que serão o alvo da análise. Em seguida a fase de pré-processamento visa a remoção de informações desnecessárias do texto. A terceira etapa é a extração de características, que transforma cada texto em uma representação numérica. Por último o classificador, que determina o sentimento de cada texto.A figura 3.1 esquematiza as etapas da análise de sentimento.

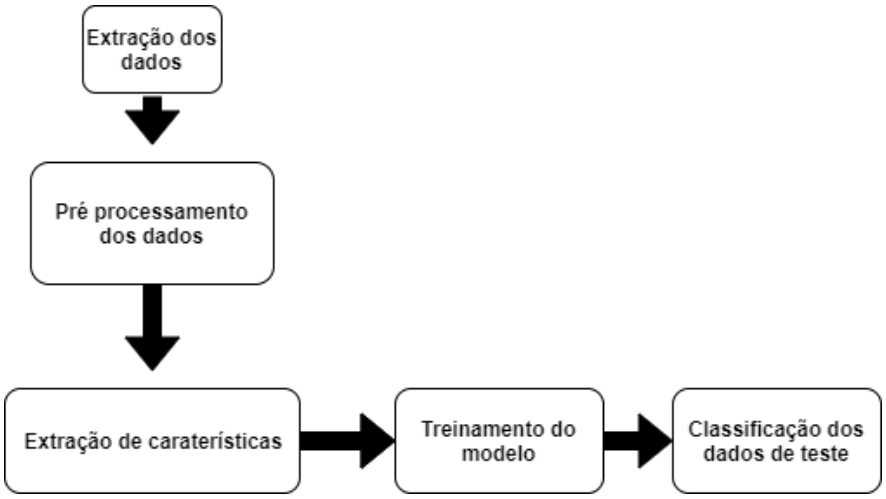


Figura 3.1: Etapas da Análise de Sentimento

3.1 PRÉ PROCESSAMENTO DO TEXTO

É a fase que visa a remoção de informações desnecessárias do texto, permitindo que o computador "entenda"de forma mais clara possível o mesmo, para a fase de extração de características.

3.1.1 Stop Words

Stop Words são palavras que são filtradas por serem comuns na linguagem e que geralmente se repetem muito em um texto . Pronomes, artigos são exemplos de *stop words*.

3.1.2 Stemming

Stemming é uma técnica para reduzir uma palavra ao seu radical, utiliza de uma heurística que tenta remover sufixos e prefixos não necessariamente importando-se com regras gramaticais. Pode-se ver pela tabela 3.1 como o *stemming* não considera as regras gramaticais, removendo o sufixo es da palavra *studies* produz o radical *studi* porém, gramaticalmente a palavra deveria ser *study*.

Forma	Sufixo	Radical
Studies	-es	Studi
Studying	-ing	Study

Tabela 3.1: Stemming

3.1.3 Lemmatization

Lemmatization também é uma técnica para reduzir uma palavra ao seu radical, porém não é agressiva como o *stemming* pois ela se baseia nas regras do dicionário, ficando limitada a regra implementada. A tabela 3.2 mostra que a *lemmatization* utiliza morfologia para produzir o radical, ou seja, o resultado é gramaticalmente correto.

Forma	Morfologia	Radical
Studies	Terceira pessoa do singular, tempo presente do verbo study	Study
Studying	Gerundio do verbo study	Study

Tabela 3.2: Lemmatization

3.2 EXTRAÇÃO DE CARACTERÍSTICAS

É a fase que transforma o texto em forma de vetor mas de maneira que ainda possua alguma informação útil do texto. Existem diversas formas para fazer essa transformação.

3.2.1 Bag of Words

O modelo *Bag of Words*(BoW) é uma técnica popular para representação de texto onde cada texto é representado pelo seu conjunto de palavras, em que conta-se quantas vezes a palavra está presente no texto, considerando um vocabulário pré definido . Por exemplo, considere os seguintes textos:

1. This is a good cat and this is a bad dog.
2. This is a bad day.
3. I like dogs.

O vetor que representa cada texto seria da forma:

	this	is	a	good	cat	and	bad	dog	day	I	like	dogs
Texto 1	2	2	2	1	1	1	1	1	0	0	0	0
Texto 2	1	1	1	0	0	0	1	0	1	0	0	0
Texto 3	0	0	0	0	0	0	0	0	0	1	1	1

Tabela 3.3: Vetores do Bag of Words

O modelo de n-gramas tenta solucionar o problema dos modelos de contagem de palavras que não consideram a ordem das palavras no texto, porém pode aumentar muito a dimensionalidade da representação.

3.2.2 N-Gramas

Um n-grama é uma sequência de N palavras: 2-grama (bigrama) é uma sequência de duas palavras como "eu estou", "o gato", e um 3-grama (trigrama) é uma sequência de três palavras como "eu estou aqui", "o gato fugiu"(Jurafsky e Martin, 2008).

3.2.3 Word2Vec

O *Word2Vec*, palavra para vetor, foi proposto por Tomas Mikolov com o intuito de resolver o problema de outros modelos que consideram palavras como unidades atômicas, onde não existe noção de similaridade entre as palavras (Mikolov et al., 2013). No *Word2vec* cada palavra é representada por um vetor, que utiliza uma rede neural de duas camadas para treinar e construir esses vetores. Existem dois algoritmos que implementam esse modelo, o primeiro é o *Continuous Bag-of-Words*(CBOW) que tenta predizer uma palavra dadas as palavras que estão próximas. O segundo algoritmo é o *Skip-Gram* que funciona de maneira contrária ao CBOW, tenta predizer as palavras de contexto (palavras próximas) dada uma palavra central.

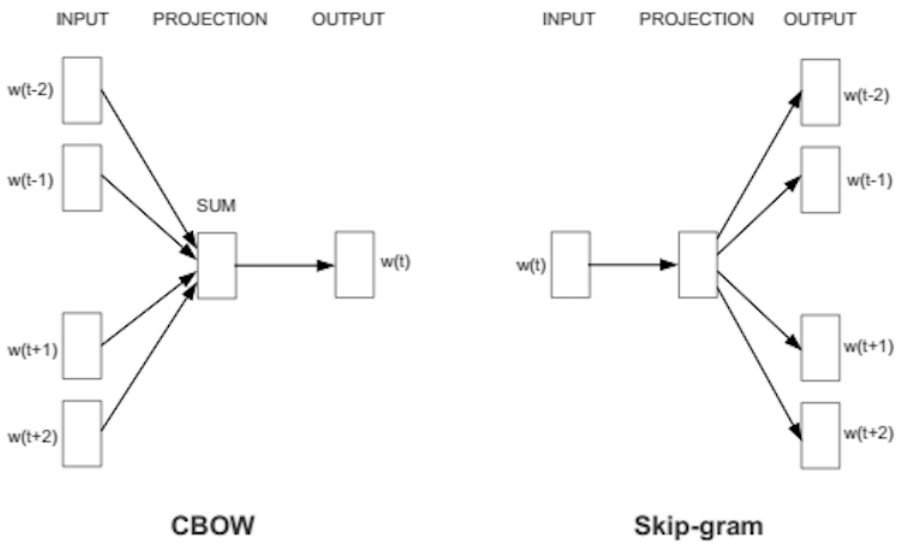


Figura 3.2: CBOW Skip-Gram
fonte:(Mikolov et al., 2013)

O resultado do *Word2vec* é um espaço onde cada ponto é uma palavra, ou seja, agora tem-se relação entre as palavras e com isso obtêm-se um entendimento semântico. Pode-se ver pela figura 3.3 que palavras com contexto parecidos estão próximas, e com isso operações maemáticas são possíveis. Por exemplo com a operação $king - man + woman$ resultaria em *queen*.

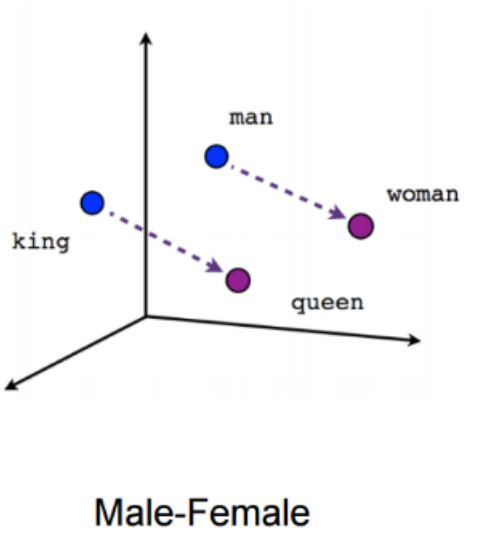


Figura 3.3: Espaço Semântico
fonte: (Shperber, 2017)

3.2.4 Doc2Vec

O *Doc2vec*, documento para vetor, é uma extensão do *Word2Vec*. É um algoritmo não supervisionado que aprende representações de tamanho fixo a partir de textos de tamanho variável como sentenças, parágrafos e documentos, onde cada documento é representado por um vetor denso que é treinado para predizer palavras no documento (Mikolov e Le, 2014).

O modelo usa como base o *Word2Vec* porém adiciona um novo vetor, o *Paragraph ID* que é uma identificação do documento. A figura 3.4 mostra como o vetor de palavras é aprendido. Dado o contexto de três palavras(*"the"*, *"cat"* e *"on"*) são usadas para predizer a quarta palavra (*"on"*)

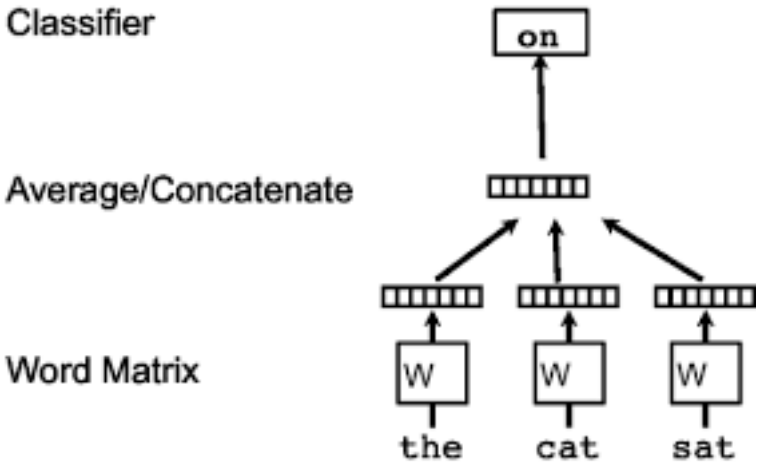


Figura 3.4: Vetor De Palavras
fonte(Mikolov e Le, 2014)

A figura 3.5 mostra como o vetor de parágrafos (documentos) funciona. Pode-se ver que o esquema é parecido com a figura 3.4 a única mudança é a adição da identificação do parágrafo (documento).

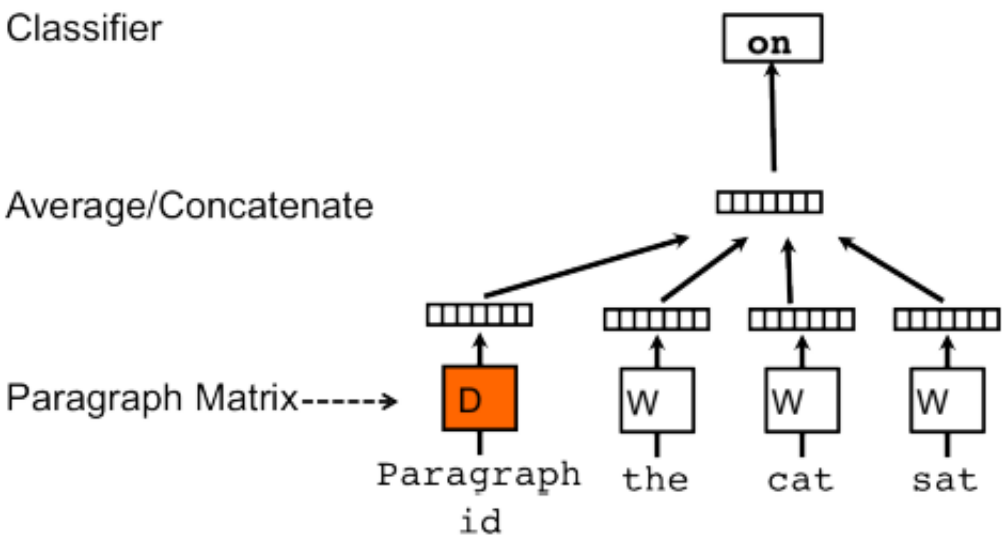


Figura 3.5: Doc2Vec
fonte(Mikolov e Le, 2014)

4 EXPERIMENTOS

A proposta do trabalho é analisar os efeitos do pré-processamento e da representação de texto para o problema de análise de sentimento.

4.0.1 Base de Dados

Para o desenvolvimento desse trabalho foi utilizada a base de dados da tarefa 4-A do SemEval de 2017 (Rosenthal et al., 2017). O SemEval(*Semantic Evaluation*) é uma competição para avaliar sistemas de análise semântica computacional, onde a tarefa 4-A de 2017 era uma análise de sentimento no *twitter* em que dado um *tweet* pretende-se determinar se ele expressa um sentimento positivo, negativo ou neutro. Foram disponibilizados os dados de todos os anos anteriores para fazer o treino, onde qualquer combinação desses dados era permitida, e um novo arquivo de teste foi criado que contém *tweets* diferentes dos encontrados nos arquivos de treino. O arquivo de treino é uma união de todos os arquivos apresentados na tabela 4.1 com suas respectivas distrubuições de classes. O arquivo de testes disponibilizado pelo SemEval está distribuido conforme a tabela 4.2.

Dataset	Positivos	Neutro	Negativo
twitter-2013train-A	3640	4586	1458
twitter-2013dev-A	575	739	340
twitter-2014test-A	982	669	202
twitter-2015train A	170	253	66
twitter-2015test-A	1038	987	365
twitter-2016train-A	3017	2001	850
twitter-2016dev-A	829	746	391
Total	10251	9981	3672

Tabela 4.1: Distribuição do Dataset de Treino

Dataset	Positivos	Neutro	Negativo
SemEval2017-task4-test.subtask-A.english	2352	5743	3811

Tabela 4.2: Distribuição do Dataset de Teste

Para trabalhar com o *dataset* de treino, foram removidos os *tweets* repetidos. Verificou-se que os dados de treino estavam desbalanceado com relação a quantidade de exemplos por classe, onde há poucos dados da classe negativa. Para contornar esse problema foi utilizada a técnica de *undersampling* onde foram selecionadas de forma aleatória dados das classe positivo e neutro de forma que possuíssem a mesma quantidade de dados da classe negativo.

4.1 MÉTRICAS

As métricas escolhidas foram as mesmas utilizadas pelo SemEval: *average recall* e *macro-average F1*. *Average recall* é a média do recall entre as classes Positivo, Negativo e Neutro,

e é calculado conforme a equação,onde R^P R^N R^U são o *recall* da classe positivo, negativo e neutro respectivamente. 4.1

$$AvgRec = \frac{1}{3}(R^P + R^N + R^U) \tag{4.1}$$

Macro-average F1 é a métrica F1 calculada sobre as classes positivo e negativo, excluindo a classe neutro, é calculado conforme 4.2, onde F_1^P e F_1^N referem-se ao F1 da classe positivo e negativo respectivamente.

$$F_1^{PN} = \frac{1}{2}(F_1^P + F_1^N) \tag{4.2}$$

4.2 METODOLOGIA EXPERIMENTAL

O trabalho foi desenvolvido em Python3, utilizando para o pré-processamento a biblioteca NLTK em conjunto com RE para tratamento de expressões regulares, para os modelos Word2Vec e Doc2Vec foi utilizada a biblioteca Gensim. Tanto para os modelos de contagem de palavras, quanto para o classificador (regressão logística) utilizou-se a biblioteca scikit-learn.

A primeira parte dos experimentos analisa o efeito do pré-processamento de texto. A segunda parte analisa a representação. Para os modelos BoW e n-gramas foram avaliados diferentes tamanhos do vocabulário e para os modelos Word2Vec e Doc2Vec foram avaliados os tamanhos de janela e tamanho do vetor de representação.

Os experimentos forma realizados de forma conjunta, ou seja, cada técnica de pré-processamento foi avaliada com todas as representações e variações de parâmetros de cada representação. Para avaliar os resultados, utilizou-se uma média do *Average Recall* , que dado um determinado parâmetro, calcula-se a média do *Average Recall* dentre todos os testes realizados em que esse parâmetro foi utilizado. Também foram analisados o maior e menor valor do *Average Recall* obtido pelo parâmetro, e o valor máximo de F_1 que foi métrica secundária utilizada pelo SemEval.

4.3 PRÉ-PROCESSAMENTO

Inicialmente realizou-se uma análise sobre o efeito do pré processamento do texto comparando as técnicas descritas na seção 3.1, também analisou-se o texto sem pré processamento, e um conjunto de operações denominado operação básica, essa processamento consiste em remover elementos que não são significativos na análise e podem representar alguma distorção. Foram feitas as seguintes operações.

- Negação: Todos os construtores negativos como *Can't*, *don't* foram substituídos por *not*.
- Menção de usuários: no twitter é comum mencionar outros usuários. Toda menção da forma @user foram removidas.
- Menção de websites: da mesma forma é comum mencionar outras páginas web. Palavras contendo http foram removidas.
- Foram removidos dígitos e todo texto foi transformado para letra minúscula.

Para este experimento, aplicou-se cada técnica de pré-processamento individualmente, e testou-se cada com todas as representações. Os resultados obtidos são apresentados na tabela 4.3.

Pré Processamento	Média Average Recall	Máximo Average Recall	Mínimo Average Recall
Stemming	0.518	**0.583**	0.473
Lemmatization	0.516	0.575	0.469
Básico	0.515	0.571	0.470
Sem Pré Processamento	0.511	0.562	0.433
Stop Word	0.478	0.547	0.390

Tabela 4.3: Average Recall Pré-Processamento

A técnica de *stemming* apresentou o melhor resultado com 58% de acerto. Observa-se que a técnica de remoção de *stop words* teve a menor taxa de acerto dentre todas as técnicas, até mesmo comparado a não pré-processar o texto, isso se deve ao fato que são removidas muitas palavras utilizando *stop words* como pode ser observado em 4.4.

Pre Processamento	Quantidade Palavras	Palavras por Tweet
Sem Pré Processamento	214751	19.90
Básico	210740	19.53
Stop Word	125736	11.65
Stemming	210649	19.52
Lemmatization	210649	19.52

Tabela 4.4: Quantidade de palavras no Dataset

4.4 REPRESETAÇÃO

4.4.1 Bag of Words e N-gramas

Nesta análise verificou-se o impacto do BOW e n-gramas como modelos para representação de texto. Foram testados diferentes tamanhos de vocabulário, em que quando limitado por um valor N, as palavras escolhidas para representar o vocabulário são as N palavras que possuem o maior número de repetição no dataset. Avaliou-se também a opção em que o vetor tem tamanho de todo o vocabulário em que foi treinado. Pode-se observar pela tabela 4.5 que a diferença do tamanho desse vetor para o BOW não causou um impacto na acurácia, não chegando a 1% a diferença tanto na média quanto no maior valor do *Average Recall*.

As tabelas 4.6 e 4.7 são referentes ao BOW 2-gramas e BOW 3-gramas reespectivamente, como esperado ambas obtiveram um resultado maior que o BOW 1-grama, pois considerar a ordem das palavras conseguem capturar melhor o que o texto representa, ou seja, começa ter um entendimento semântico. Em relação ao tamanho do vocabulário, ambas representações obtiveram resultado parecido.

No entanto, ao contrário da versão uni-grama do BOW, a diferença entre usar todo o vocabulário ou fixar um tamanho foi considerável, onde a média dos experimentos usando todo o vocabulário tiveram um aumento de 3% no *Average Recall*. A partir de bigramas a dimensionalidade aumenta muito, porém usar todo o vocabulário apresentou um resultado melhor.

4.4.2 Word2Vec

Para o *Word2Vec* foram testados os valores de tamanho de janela e o tamanho do vetor de caraterísticas treinados no próprio *dataset*, também foi testada uma versão pré treinada do

Tamanho Vocabulário	Média Average Recall	Máximo Average Recall	Mínimo Average Recall	Máximo F1
3000	**0.534**	0.554	0.517	0.541
2400	0.533	0.553	0.510	0.537
Todo Vocabulário	0.532	0.550	0.517	0.543
3500	0.531	0.552	0.512	0.542
1800	0.531	0.551	0.511	0.531
2100	0.531	0.552	0.509	0.535
1500	0.528	0.550	0.502	0.525

Tabela 4.5: Tamanho Vocabulário BOW

Tamanho Vocabulário	Média Average Recall	Máximo Average Recall	Mínimo Average Recall	Máximo F1
Todo Vocabulário	**0.550**	0.569	0.530	0.558
1800	0.530	0.548	0.505	0.520
3500	0.529	0.543	0.515	0.526
1500	0.529	0.548	0.509	0.519
2100	0.529	0.547	0.514	0.520
3000	0.528	0.545	0.516	0.525
2400	0.526	0.544	0.514	0.519

Tabela 4.6: Tamanho Vocabulário BOW 2-gramas

Tamanho Vocabulário	Média Average Recall	Máximo Average Recall	Mínimo Average Recall	Máximo F1
Todo Vocabulário	**0.556**	0.577	0.529	0.558
1800	0.529	0.548	0.505	0.519
2100	0.529	0.547	0.509	0.520
1500	0.528	0.548	0.511	0.522
3500	0.527	0.543	0.514	0.527
3000	0.527	0.547	0.515	0.526
2400	0.525	0.543	0.509	0.516

Tabela 4.7: Tamanho Vocabulário BOW 3-gramas

Word2Vec fornecido pela Google. Esse modelo possui um vetor de características de dimensão 300 e foi treinado em aproximadamente 100 bilhões de palavras em contraste com esse *dataset* que possui 214751 palavras.

Pode-se observar pelas tabelas 4.8 e 4.9 a baixa variação da média do Average Recall, então tanto o tamanho da janela quanto vetor de características, não tiveram uma grande influência na taxa de acerto. A baixa diferença nos valores possivelmente mostra que o modelo não aprendeu a reconhecer as palavras. Como os textos do *twitter* são curtos, com poucas palavras, e muitas gírias dificultam o modelo aprender o contexto da palavra.

Tamanho da Janela	Média Average Recall	Máximo Average Recall	Mínimo Average Recall	Máximo F1
8.0	0.518	0.534	0.484	0.532
10.0	0.517	0.533	0.482	0.533
11.0	0.517	0.531	0.486	0.530
15.0	0.516	0.538	0.480	0.531

Tabela 4.8: Tamanho da Janela Word2Vec

Tamanho Vetor Características	Média Average Recall	Máximo Average Recall	Mínimo Average Recall	Máximo F1
300	0.519	0.533	0.486	0.533
250	0.518	0.534	0.487	0.532
200	0.517	0.534	0.487	0.532
150	0.516	0.533	0.485	0.528
120	0.516	0.538	0.480	0.526

Tabela 4.9: Tamanho do Vetor Word2Vec

Por mais que o Word2vec seja um modelo mais robusto que o BOW obteve um resultado pior, isso porque a quantidade de dados foi insuficiente para o modelo aprender a representar uma palavra. Podemos observar pela tabela 4.10 que o *Word2Vec* treinado pela Google obteve um resultado melhor, chegando a 60% de taxa de acerto, esse resultado também é superior ao obtido pelo BOW.

Average Recall	F1
0.602	0.617

Tabela 4.10: Word2Vec Google

4.4.3 Doc2Vec

O *Doc2Vec* foi submetido às mesmas análises do *Word2Vec*, pode-se observar pelas tabelas 4.11 e 4.12 que o resultado foi mais baixo que o obtido pelo *Word2Vec*, obtendo no máximo um *Average Recall* de 50,2%. O *Doc2Vec* sofre do mesmo problema que *Word2Vec*, em que os textos possuem poucas palavras.Além disso, o baixo volume de documentos, cerca de 10 mil, também prejudicaram no resultado final do 4.11.

A tabela 4.11 mostra que maiores tamanhos de janela obtiveram uma acurácia maior. Como os textos são curtos, possivelmente usar todo o texto para definir cada palavra do texto gera um resultado melhor.

Tamanho da Janela	Média Average Recall	Máximo Average Recall	Mínimo Average Recall	Máximo F1
15	**0.488**	0.502	0.478	0.500
11	0.485	0.501	0.475	0.498
10	0.483	0.496	0.472	0.498
8	0.478	0.494	0.468	0.496

Tabela 4.11: Tamanho da Janela Doc2Vec

Tamanho Vetor Características	Média Average Recall	Máximo Average Recall	Mínimo Average Recall	Máximo F1
200	0.480	0.502	0.455	0.500
150	0.480	0.500	0.456	0.500
120	0.480	0.502	0.456	0.498
250	0.480	0.500	0.456	0.497
300	0.480	0.500	0.457	0.496

Tabela 4.12: Tamanho do Vetor Doc2Vec

4.5 RESULTADO DO SEMEVAL

A tabela 4.13 mostra os resultados obtidos neste trabalho, em que pode-se observar que, das representações treinadas no próprio *dataset*, a versão trigrama do BOW apresentou o melhor resultado, porém dentre todos os testes, a versão do *Word2Vec* pré treinado pela Google obteve um resultado ainda melhor. A imagem 4.1 mostra o resultado da competição do SemEval para a tarefa 4A "Classificação da Polaridade da Mensagem"em inglês de 2017 e está ordenado pelo *Average Recall* em que quanto maior o valor melhor.Com o valor obtido de 0.602 com *Word2Vec* treinado pela Google esse trabalho ocuparia a posição de número 23.

Representação	Average Recall	F1
Bow	0.554	0.543
Bow-2grama	0.569	0.558
Bow-3grama	0.577	0.558
W2V	0.538	0.505
Word2Vec-Google	**0.602**	0.617
Doc2Vec	0.502	0.481

Tabela 4.13: Resultados por representação

#	System	*AvgRec*	F_1^{PN}
1	DataStories	**0.681**$_1$	0.677$_2$
	BB_twtr	**0.681**$_1$	0.685$_1$
3	LIA	**0.676**$_3$	0.674$_3$
4	Senti17	**0.674**$_4$	0.665$_4$
5	NNEMBs	**0.669**$_5$	0.658$_5$
6	Tweester	**0.659**$_6$	0.648$_6$
7	INGEOTEC	**0.649**$_7$	0.645$_7$
8	SiTAKA	**0.645**$_8$	0.628$_9$
9	TSA-INF	**0.643**$_9$	0.620$_{11}$
10	UCSC-NLP	**0.642**$_{10}$	0.624$_{10}$
11	HLP@UPENN	**0.637**$_{11}$	0.632$_8$
12	YNU-HPCC	**0.633**$_{12}$	0.612$_{15}$
	SentiME++	**0.633**$_{12}$	0.613$_{13}$
14	ELiRF-UPV	**0.632**$_{14}$	0.619$_{12}$
15	ECNU	**0.628**$_{15}$	0.613$_{13}$
16	TakeLab	**0.627**$_{16}$	0.607$_{16}$
17	DUTH	**0.621**$_{17}$	0.605$_{17}$
18	CrystalNest	**0.619**$_{18}$	0.593$_{19}$
19	deepSA	**0.618**$_{19}$	0.587$_{20}$
20	NILC-USP	**0.612**$_{20}$	0.595$_{18}$
21	Ti-Senti	**0.607**$_{21}$	0.577$_{22}$
22	BUSEM	**0.605**$_{22}$	0.587$_{20}$
23	EICA	**0.595**$_{23}$	0.555$_{24}$
24	OMAM	**0.590**$_{24}$	0.542$_{26}$
25	Adullam	**0.589**$_{25}$	0.552$_{25}$
26	NileTMRG	**0.578**$_{26}$	0.515$_{32}$
27	Amobee-C-137	**0.575**$_{27}$	0.520$_{30}$
28	ej-za-2017	**0.571**$_{28}$	0.539$_{27}$
	LSIS	**0.571**$_{28}$	0.561$_{23}$
30	XJSA	**0.556**$_{30}$	0.519$_{31}$
31	Neverland-THU	**0.555**$_{31}$	0.507$_{33}$
32	MI&T-Lab	**0.551**$_{32}$	0.522$_{29}$
33	diegoref	**0.546**$_{33}$	0.527$_{28}$
34	xiwu	**0.479**$_{34}$	0.365$_{34}$
35	SSN_MLRG1	**0.431**$_{35}$	0.344$_{35}$
36	YNUDLG	**0.340**$_{36}$	0.201$_{37}$
37	WarwickDCS	**0.335**$_{37}$	0.221$_{36}$
	Avid	**0.335**$_{37}$	0.163$_{38}$
B1	All POSITIVE	**0.333**	0.162
B2	All NEGATIVE	**0.333**	**0.244**
B3	All NEUTRAL	**0.333**	0.000

Figura 4.1: Resultados SemEval

5 CONCLUSÃO

Este trabalho analisou o efeito de diversas técnicas de pré processamento e representação de texto para o problema de análise de sentimento. Os experimentos foram feitos foram baseados na tarefa 4A do Semeval de 2017 que consiste em classificar o sentimento de textos da rede social *twitter* em positivo, negativo ou neutro.

Em relação ao pré processamento, conclui-se que é necessário um cuidado grande, pois os textos produzidos na rede social possuem uma quantidade baixa de palavras, então técnicas que podem remover muitas palavras como *stop words* diminuem muito a acurácia.

Com a baixa quantidade de textos, foi verificado que modelos de representação mais simples apresentam um bom resultado, porém utilizando um modelo mais robusto, pré-treinado em muitos documentos mostrou que essas técnicas mais complexas se treinadas com muito mais dados tendem a apresentar um resultado melhor.

Para trabalhos futuros cabe um tratamento mais específico nos dados, por exemplo, tratar *emoticons*, ironia, onomatopeias, que são frequentemente encontradas em textos colocados em redes sociais, e treinar os modelos com uma quantidade maior de dados.

REFERÊNCIAS

Angiani, G., Ferrari, L., Fontanini, T., Fornacciari, P., Iotti, E., Magliani, F. e Manicardi, S. (2017). A comparison between preprocessing techniques for sentiment analysis in twitter.

Indurkhya, N. e Damerau, F. J. (2010). *Handbook of Natural Language Processing*. https://www.crcpress.com/.

Joulin, A., Grave, E., Bojanowski, P. e Mikolov, T. (2016). Bag of tricks for efficient text classification.

Jr, E. A. C., Marinho, V. Q. e dos Santos, L. B. (2017). Nilc-usp at semeval-2017 task 4: A multi-view ensemble for twitter sentiment analysis.

Jurafsky, D. e Martin, J. H. (2008). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall.

Liddy, E. D. (2001). Natural language processing.

Mikolov, T., Chen, K., Corrado, G. e Dean, J. (2013). Efficient estimation of word representations in vector space.

Mikolov, T. e Le, Q. (2014). Distributed representations of sentences and documents.

RaRe-Technologies (2019). Document classification with word embeddings tutorial. https://github.com/RaRe-Technologies/movie-plots-by-genre/blob/master/ipynb_with_output/Document%20classification%20with%20word%20embeddings%20tutorial%20-%20with%20output.ipynb. Acessado em 10/10/2019.

Rosenthal, S., Farra, N. e Nakov, P. (2017). Semeval-2017 task 4: Sentiment analysis in twitter.

Shperber, G. (2017). A gentle introduction to doc2vec. https://medium.com/wisio/a-gentle-introduction-to-doc2vec-db3e8c0cce5e. Acessado em 10/10/2019.