# COMPARAÇÃO DE TÉCNICAS DE *WORD EMBEDDING* NA ANÁLISE DE SENTIMENTOS

Gustavo Ziger*
Fernando Roberto Pereira*

**Resumo**

A quantidade expressiva de dados e informações que os usuários da internet geram diariamente possui potencial de uso em atividades de mineração de dados e reconhecimento de padrões. Considerando que a tomada de decisão humana é altamente influenciada por opiniões externas, as informações geradas pelos usuários de blogs, sites de avalição e redes sociais podem ser utilizadas em tarefas de análise de sentimentos. Este trabalho compara a acurácia de diferentes técnicas para representação de sentenças em números, chamadas de *word embeddings*, utilizando uma rede neural convolucional como classificador. Os experimentos realizados utilizaram o conjunto de avaliações de filmes em língua inglesa chamado "*Large Movie Review Dataset v1.0*". Com a abordagem proposta obtivemos 90,30% de acurácia com o melhor experimento.

**Palavras-Chave:** Processamento de linguagem natural. Mineração de opinião. Análise de sentimento. *Word embedding.*

## 1 INTRODUÇÃO

Nos últimos anos, as áreas de mineração de dados, mineração de opinião e análise de sentimentos têm recebido atenção dos pesquisadores. Estas são áreas de pesquisa que estão sendo aplicadas no meio corporativo para a análise de tendência do mercado e também para a identificação de perfis de usuários em compras no e-commerce. De acordo com BBC News Brasil (2018), estas técnicas podem ter sido

* Acadêmico do curso de Análise e Desenvolvimento de Sistemas do Instituto Federal de Santa Catarina. gustavo.ziger@ifsc.edu.br
* Professor Orientador do curso de Análise e Desenvolvimento de Sistemas do Instituto Federal de Santa Catarina. fernando.pereira@ifsc.edu.br

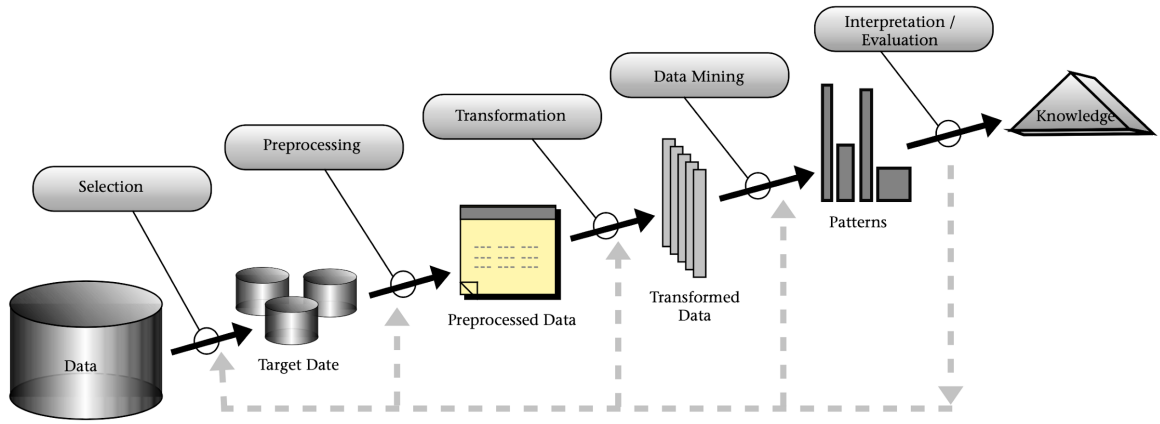
utilizadas como meio de polarizar e moldar pensamentos, caso que pode ter sido determinante na eleição de 2016 dos Estados Unidos da América.

Com o avanço tecnológico das últimas décadas e o crescimento do volume de dados gerados por meio da Internet, sejam elas nos meios acadêmicos, científicos, corporativos ou até mesmo nas mídias sociais, é evidente o valor que os dados gerados e armazenados diariamente possuem para pesquisas e análises (MARQUESONE, 2016).

Para que ocorra a análise sentimental de um texto é necessário que etapas bem definidas sejam seguidas. Fayyad, Piatetsky-Shapiro e Smyth (1996) introduzem a definição destas etapas com o conceito de *Knowledge Discovery in Databases (KDD)*. Os autores apresentam cinco passos, conforme a Figura 1: seleção, pré-processamento, transformação, mineração de dados e interpretação. Os passos descritos acima possuem como finalidade a geração de conhecimento.

A definição de conhecimento, no que tange a apresentação dada no contexto do KDD, apresentado também por Fayyad, Piatetsky-Shapiro e Smyth (1996), é delimitada por qualquer função e domínio escolhidas pelo usuário.

Figura 1 – Passos que compõem o processo do KDD




Fonte: From Data Mining to Knowledge Discovery in Databases (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996)

Este artigo trata do processamento de linguagem natural e da mineração de opinião, chamada também de análise de sentimentos – quando o escopo da pesquisa

é centrado e definido pela polaridade dos textos analisados –, mais precisamente na comparação das técnicas de *word embedding*, ou seja, representação de texto por meio de um espaço multidimensional, e o seu impacto na classificação dos textos.

Os experimentos foram realizados com o conjunto de dados compilado e utilizado no trabalho acadêmico de Maas *et al.* (2011), encontrado nos domínios da Universidade Stanford, contendo cinquenta mil avaliações de filmes em língua inglesa, classificadas entre positivas e negativas. Foram utilizadas técnicas de pré-processamento e técnicas de representação de texto baseadas em *word embedding* para gerar três modelos distintos. Também foi treinada uma rede neural convolucional e posteriormente rodados testes de classificação nesta mesma rede utilizando a métrica de acurácia.

## 2 OBJETIVOS

### 2.1 Objetivo geral

O trabalho compara diferentes técnicas de *word embedding* na classificação de avaliações de filmes em língua inglesa e avalia os resultados por meio da métrica acurácia.

### 2.2 Objetivos específicos

1. Pré-processar o conjunto de dados "*Large Movie Review Dataset v1.0*" com a finalidade de remover ruídos indesejáveis, separar e preparar para treinamento e teste da rede neural.
2. Gerar *word embeddings* com as técnicas *Word2Vec*, *Doc2Vec* e *fastText*, pela biblioteca *Gensim*.
3. Utilizar uma Rede Neural Convolucional (RNC) para realizar a extração de características e posterior classificação das sentenças.
4. Avaliar o desempenho das *word embeddings* com a métrica acurácia e comparar os resultados de cada abordagem com o estado da arte.

## 3 FUNDAMENTAÇÃO TEÓRICA

A fundamentação teórica do presente artigo foi estruturada de forma a introduzir conceitos essenciais para o entendimento da obra, no que segue.

### 3.1 Mineração de Opinião e Análise de Sentimentos

A análise de sentimentos, também chamada de mineração de opinião, segue os passos da mineração de dados, definida como “a aplicação de algoritmos específicos para a extração de padrões dos dados” (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996, tradução do autor).

A área da análise de sentimentos busca encontrar a subjetividade dos dados analisados, podendo ser utilizada para obter posicionamentos, visões, sentimentos e opiniões nos textos pesquisados. Segundo o que afirma Naganna e Tripathi (2014), a mesma teve crescimento rápido a partir dos anos 2000, impulsionada pelo volume de dados criados na forma de comentários, debates e avaliações em geral.

A mineração de opinião pode ser classificada em três níveis de atuação, sendo eles: nível de documento, sentença e aspecto. No nível de documento a classificação sentimental ocorre em sua totalidade, assumindo um único sentimento para todo o texto analisado, já para o segundo nível, cada sentença individual do texto é analisada. Para o nível de aspecto, as características de um determinado produto são extraídas do texto (NAGANNA; TRIPATHI, 2014).

### 3.2 Representação de Texto

Para que consigamos realizar qualquer processamento de linguagem natural é necessário que seja utilizada uma técnica de representação de texto.

Uma abordagem muito simples é a de associar cada palavra a um índice no vocabulário do conjunto de dados utilizado. Outra técnica, a *one-hot encoding*, associa cada palavra como um valor 1 em um vetor de zeros de tamanho ***n*** diretamente proporcional ao tamanho do vocabulário, onde somente os índices das palavras correspondentes é igual a 1.

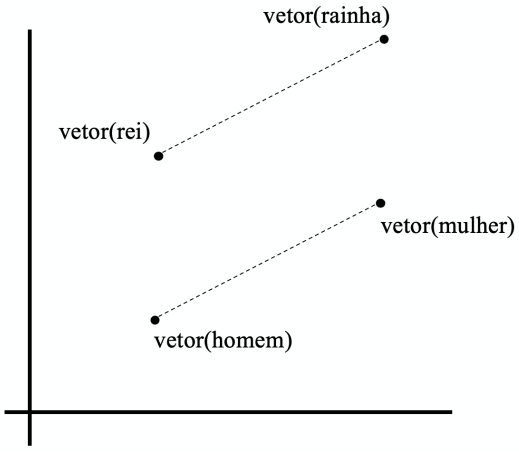
Neste trabalho foram utilizadas *word embeddings*, também chamadas de *dense word vectors*. Cada palavra é representada por um vetor de tamanho fixo capaz de capturar o seu significado semântico. Os vetores são construídos com base nos dados fornecidos e o contexto que as palavras do conjunto estão inseridas.

Com as técnicas de *word embedding* as palavras que possuem sentidos similares tendem a ficar próximas umas das outras, já que a codificação numérica é semelhante. Com esta abordagem e com operações algébricas é possível calcular a correlação de palavras. A Equação (1) exemplifica a forma como o cálculo de uma correlação ocorre.

$$vetor(rei) - vetor(homem) + vetor(mulher) = vetor(rainha) \quad (1)$$

A palavra mostrada em um cálculo similar ao realizado pela Equação (1) é a que possui a representação vetorial mais próxima do valor obtido pela operação. A Figura 2 apresenta o exemplo anterior, retirado dos trabalhos de Mikolov, Yih e Zweig (2013), colocando as palavras em um gráfico.

Figura 2 – Representação cartesiana dos vetores utilizados na Equação (1).




Fonte: O autor.

### 3.3 Algoritmos de representação de palavras

Estes algoritmos são os responsáveis por gerar as representações numéricas das

palavras de um conjunto de texto e armazená-las em vetores. Considerando o conceito de *word embedding*, são apresentados três algoritmos distintos capazes de gerar as representações citadas acima.

### 3.3.1 *Word2Vec*

Proposto por Mikolov *et al*. (2013), este algoritmo trabalha com as arquiteturas de *bag-of-words* e *skip-gram* contínuos, sendo ambos modelos log-lineares. No primeiro caso, o algoritmo tenta prever a palavra alvo, ou central, com base no contexto em que ela está inserida. Já no modelo *skip-gram* a predição é feita de maneira oposta, as palavras do contexto é que são previstas com base na palavra central. A técnica possui uma complexidade computacional bem menor comparada a outras baseadas em redes neurais presentes na literatura. Consegue computar *word embeddings* muito precisas em grandes conjuntos de dados e em pouco tempo.

### 3.3.2 *Doc2Vec*

Técnica proposta por Le e Mikolov (2014) que possui inspiração no *Word2Vec* e outros algoritmos relacionados e permite a análise do documento como um todo. Dado o tamanho de entrada variável que o algoritmo permite utilizar, os autores buscaram estender a sua aplicação para representações de sentenças, parágrafos e documentos completos, e não somente a de uma palavra individual.

### 3.3.3 *fastText*

Biblioteca desenvolvida pelo laboratório de inteligência artificial do *Facebook* e discutida nos trabalhos de Bojanowski *et al*. (2017). É descrita como uma extensão do modelo *Word2Vec*, mas trata cada palavra como uma sequência de ***n*** itens, chamados n-grama. Os itens podem ser letras, sílabas, fonemas ou até palavras completas. A técnica promete aumentar a precisão das *embeddings* geradas, inclusive em conjuntos de dados pequenos.

## 3.4 Rede Neural Artificial (RNA)

Para Haykin (2002), uma rede neural artificial é um processador distribuído e paralelo que possui semelhanças com o cérebro humano. A RNA apresenta a habilidade de armazenar conhecimento experimental por meio de unidades computacionais interligadas chamadas de neurônios. As semelhanças ocorrem na forma em como o conhecimento é adquirido – por meio do ambiente, em um processo de aprendizagem – e em como os neurônios são conectados.

As redes neurais convolucionais (RNC), conforme a definição de O'Shea e Nash (2015), são semelhantes às RNAs convencionais, já que possuem neurônios que utilizam o aprendizado para se adaptar e fazer o próprio ajuste. Uma RNC é uma rede do tipo *feed-forward*, com conexões não cíclicas, são inspiradas nos processos biológicos e geralmente são aplicadas em análises de imagens e reconhecimento de voz.

São utilizados alguns tipos específicos de camadas na arquitetura, sendo estas as camadas de *embedding*, de *dropout*, de convolução, de *pooling* (ou agrupamento) e densas (ou totalmente conectadas). A seguir, uma descrição da arquitetura da RNC utilizada no desenvolvimento dos experimentos.

### 3.4.1 Entrada ou *Input*

Responsável pela entrada de dados e serve como ponto de partida da rede neural. Recebe os textos já vetorizados para que seja realizado o processamento e treino nas camadas seguintes.

### 3.4.2 *Embedding*

Camada que ajusta os pesos da rede neural e gera os vetores de espaço multidimensional fixo para representar a entrada.

### 3.4.3 *Dropout*

É uma forma de regularização proposta por Srivastava *et al*. (2014). É uma camada responsável por ‘desligar’ algumas unidades, ou neurônios, de entrada de maneira aleatória conforme o valor do parâmetro aplicado. Estas unidades desligadas são ignoradas durante o treino e contribuem com a redução do *overfitting* – situação que ocorre quando o modelo aprendido corresponde quase que exatamente ao conjunto de treino, se tornando ineficaz no conjunto de validação e teste.

### 3.4.3 Camada de Convolução

Na matemática, é um operador linear que mede a soma do produto de duas funções dadas, resultando em uma terceira. Aplicado no contexto das redes neurais, para O’Shea e Nash (2015), a convolução trata de aplicar filtros às entradas da rede que resultam na extração das informações. Para realizar a extração são utilizados *kernels* que filtram os dados e são responsáveis por encontrar informações ou características específicas da entrada, como resultado são gerados mapas de ativação.

### 3.4.4 *Pooling*

Possui como foco a redução da complexidade computacional do modelo para as camadas seguintes (ALBAWI; MOHAMMED; AL-ZAWI, 2017). Reduz o número de parâmetros utilizados nas camadas anteriores e a sua dimensionalidade. A técnica diminui o mapa de ativação de acordo com o tamanho do *kernel* utilizado (O’SHEA; NASH, 2015).

### 3.4.5 Camada Densa

Também chamada de Camada Totalmente Conectada, é organizada de forma que cada neurônio é inteiramente conectado às suas camadas imediatamente anteriores e posteriores no modelo. Nas redes neurais tradicionais, essa organização acontece de forma análoga (ALBAWI; MOHAMMED; AL-ZAWI, 2017).

### 3.5 Trabalhos Relacionados

A seguir são apresentados trabalhos relacionados e os resultados alcançados em cada obra com o mesmo conjunto de dados escolhido para o desenvolvimento do presente artigo.

O trabalho proposto por Maas *et al*. (2011) contempla a metodologia para elaboração da base de dados denominada "*Large Movie Review Dataset v1.0”*, constituída de avaliações de filmes em língua inglesa. Os autores propõem um modelo definido como log-bilinear para captura semântica e sentimental nas relações entre palavras e também no consequente cálculo das *word embeddings,* e utilizam um *Support Vector Machine* (SVM) linear para avaliar a eficácia da abordagem na classificação de sentenças e documentos. Nos seus testes de classificação obtiveram 88,89% de acurácia no conjunto de teste.

Na obra de Camacho-Collados e Pilehvar (2017), os autores avaliam diversas técnicas de pré-processamento de texto e o impacto do uso na classificação e análise de sentimentos. Na melhor abordagem, sem pré-processar o texto, obtiveram a acurácia de 88,90% utilizando uma rede neural convolucional. A RNC opera com *word embeddings* pré-treinadas pelo algoritmo *Word2Vec* na camada de Embedding e também faz uso da arquitetura *long short-term memory* (LSTM).

Gui *et al*. (2019) introduzem e utilizam uma variação do LSTM, que emprega a chamada *dynamic skip connection*, permitindo a captura da conexão direta entre duas palavras dependentes mesmo que estejam afastadas na frase analisada. Os autores atingiram a acurácia de 90,10% na classificação utilizando uma rede neural recorrente (RNR) e a técnica de LSTM proposta. A abordagem utilizada emprega *word embeddings* pré-treinadas pelo algoritmo *Word2Vec* como forma de inicialização da rede.

Nos experimentos de Johnson e Zhang (2015), os autores comparam diferentes arquiteturas de redes neurais convolucionais e a aplicação de *bag-of-words* na camada de convolução para a classificação de textos. O melhor experimento resultou em 92,33% de acurácia na RNC. As *word embeddings* utilizadas na obra foram geradas pela própria rede neural convolucional.

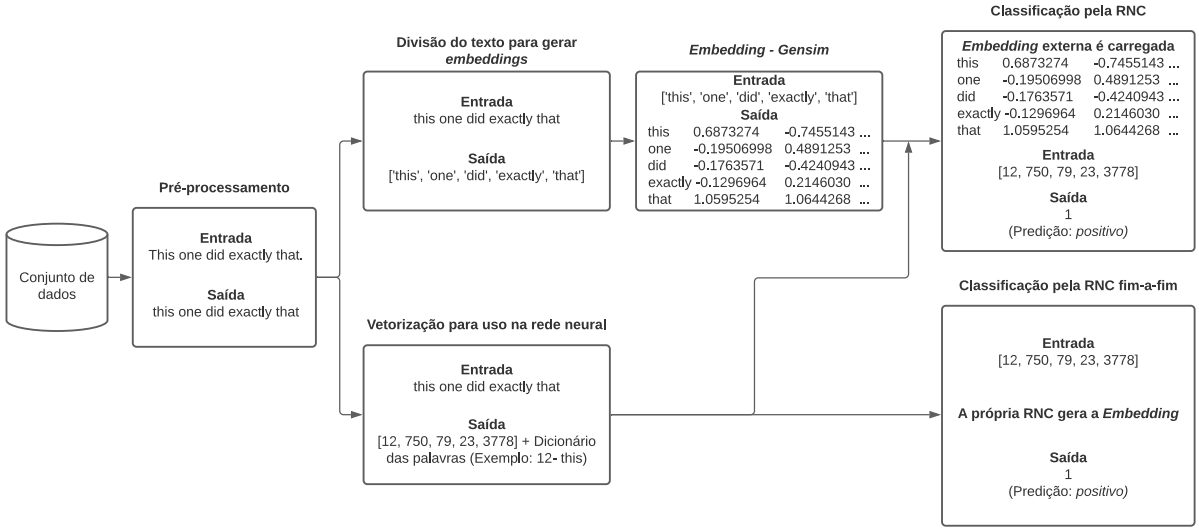
# 4 METODOLOGIA

Os experimentos realizados visam comparar o impacto de quatro técnicas de *word embedding* comumente utilizadas na tarefa de classificação de sentenças, especificamente, na análise de sentimentos. Para mensurar o desempenho das técnicas, utilizamos a métrica de avaliação acurácia no conjunto de teste.

Inicialmente, realizamos a divisão da base de dados (Seção 4.1). Em seguida, realizamos um pré-processamento no conjunto de dados e geramos *word embeddings* por meio das técnicas *Word2Vec*, *Doc2Vec* e *fastText*, pela biblioteca *Gensim*. Também, geramos *word embeddings* com a rede neural fim-a-fim, arquitetura implementada por meio das bibliotecas *Keras* e *TensorFlow (Seção 4.2)*. Por fim, avaliamos a acurácia das técnicas de *word embedding* no conjunto de testes.

A Figura 3 apresenta o fluxograma das etapas para classificação de uma sentença.

Figura 3 – Fluxograma das etapas do trabalho



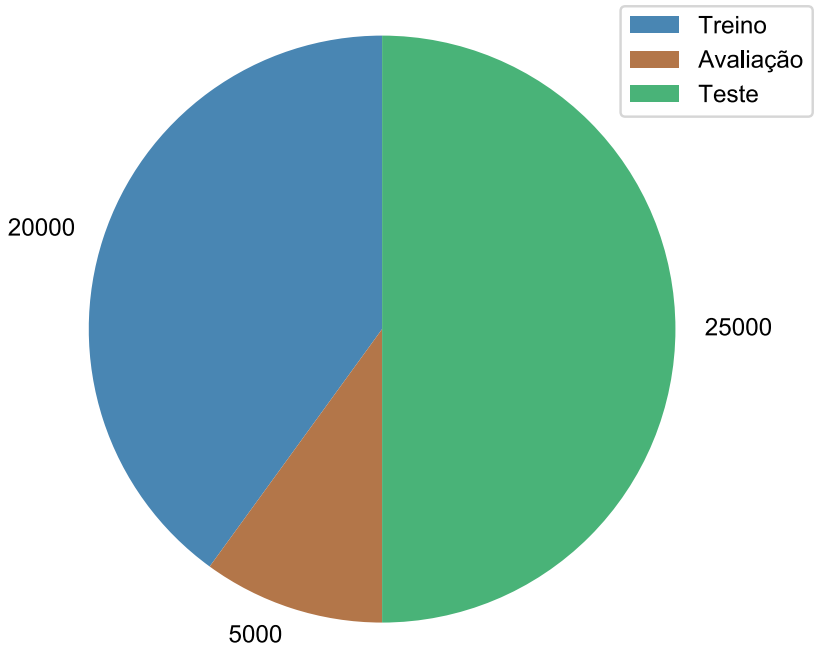

Fonte: O Autor.

## 4.1 Escolha e preparação do conjunto de dados

Para o desenvolvimento dos experimentos utilizamos a base de dados "*Large Movie Review Dataset v1.0*", contendo cinquenta mil avaliações de filmes igualmente divididas em positivas e negativas, ou seja, o conjunto de dados é balanceado. A base de dados proposta por Maas *et al.* (2011), contempla os conjuntos de treino e teste,

porém, dividimos de maneira arbitrária o conjunto de treinamento em dois, oitenta por cento para formar o conjunto de treino e vinte por cento para formar o conjunto de validação. Desta forma, após a divisão, possuímos vinte e cinco mil avaliações para teste, vinte mil para treino e cinco mil para validação, com demonstração feita pela Figura 4. O conjunto de treino é utilizado para gerar as *embeddings*, bem como para treinar a rede neural convolucional. O conjunto de validação é utilizado para validar o treino feito pela RNC. Por fim, o conjunto de teste avalia o desempenho da rede utilizando a métrica acurácia.

Figura 4 – Divisão da base de dados




Fonte: O Autor.

O pré-processamento realizado contempla as seguintes etapas:

1. Remove qualquer tipo de código HTML.
2. Remove links ou nomes de usuário.
3. Expande contrações.
4. Transforma todo o texto para minúsculo.
5. Remove a pontuação.
6. Remove espaços extras.

A Tabela 1 apresenta exemplos de textos na forma bruta e na forma pós-processada.

Tabela 1 – Exemplo de textos utilizados na pesquisa

| **Texto bruto** | **Polaridade** |
|---|---|
| The sign of a good movie is that it can toy with our emotions. This one did exactly that. | Positivo |
| The first 20 minutes were a little fun because I don't think I've seen a film this bad before {acting, script, effects (!), etc....} | Negativo |
| | |
| **Texto após o processamento** | **Polaridade** |
| the sign of a good movie is that it can toy with our emotions this one did exactly that | Positivo |
| the first 20 minutes were a little fun because i do not think i have seen a film this bad before acting script effects etc | Negativo |

Fonte: O Autor.

Para gerar as *embeddings* externas à rede neural, o texto passa por uma etapa adicional que divide todo o conteúdo em uma lista de palavras individuais. A Tabela 2 ilustra o processo de divisão de sentenças em lista de palavras.

Tabela 2 – Exemplo de textos com a etapa adicional aplicada

| **Texto com etapa adicional** | **Polaridade** |
|---|---|
| ['the', 'sign', 'of', 'a', 'good', 'movie', 'is', 'that', 'it', 'can', 'toy', 'with', 'our', 'emotions', 'this', 'one', 'did', 'exactly', 'that'] | Positivo |
| ['the', 'first', '20', 'minutes', 'were', 'a', 'little', 'fun', 'because', 'i', 'do', 'not', 'think', 'i', 'have', 'seen', 'a', 'film', 'this', 'bad', 'before', 'acting', 'script', 'effects', 'etc'] | Negativo |

Fonte: O Autor.

## 4.2 Codificação dos experimentos

Os experimentos foram desenvolvidos em linguagem de programação *Python*

3.8.1 e ambiente de desenvolvimento integrado *Visual Studio Code* 1.53.3. Com uma máquina *macOS Big Sur* 11.1, processador Intel Core i7-4850HQ rodando a 2.4 GHz, 16 GB de memória RAM e uma placa de vídeo externa NVIDIA GeForce GT 750M com 2 GB de memória dedicada.

Foi escolhida e utilizada a biblioteca *Gensim* na versão 3.8.3 para gerar e treinar as *embeddings* externas que foram posteriormente carregadas na rede neural. Esta biblioteca fornece algoritmos otimizados, com performance estável e que resolvem o problema proposto. Os parâmetros utilizados foram exatamente os mesmos para cada algoritmo, foram escolhidos de maneira arbitrária e estão listados conforme a Tabela 3 a seguir.

Tabela 3 – Parâmetros de treino na biblioteca Gensim

| **Parâmetro** | **Valor** |
|---|---|
| Entrada de dados | Conjunto de treino |
| Tamanho do vetor gerado | 100 |
| Mínima repetição de uma palavra | 1 |
| Épocas | 50 |

Fonte: O Autor.

Para a construção da rede neural convolucional fim-a-fim utilizamos a biblioteca *Keras* com *backend TensorFlow* na versão 2.4.0. A camada *Embedding* da rede depende que os textos estejam em forma de vetores, desta forma, utilizamos a classe *TextVectorization* que elabora, a partir do vocabulário do conjunto de treino, um dicionário que mapeia um número inteiro para cada palavra. Em seguida, cada texto ou sentença é transformado em um vetor de inteiros com base no dicionário. Com essa vetorização inicial, é possível alimentar a rede neural e iniciar os treinamentos e testes.

A Tabela 4 apresenta um exemplo da vetorização do texto em números inteiros.

Tabela 4 – Exemplo de texto vetorizado para utilização na RNC

| Texto pré-processado | Texto vetorizado |
|---|---|
| the sign of a good movie is that it can toy with our emotions this one did exactly that | [13, 89, 7, 3, 7803, 14023, 903, 3778, 31, 21912, 54793, 120, 370, 549, 12, 750, 79, 23, 3778] |

Fonte: O Autor.

Foi utilizada uma RNC composta pelas respectivas camadas e parâmetros:

1. ***Input***: Recebe os textos vetorizados para entrada da rede.
2. ***Embedding***: O número de entrada, ou de *tokens,* é 82032 conforme o vocabulário do vetorizador. O tamanho do vetor segue em 100 como nos outros algoritmos.
3. ***Dropout***: A taxa de *dropout* foi definida em 0,5, isto é, cinquenta por cento dos neurônios são desligados aleatoriamente em cada ciclo de atualização dos pesos do modelo.
4. ***Conv1D***: Camada de convolução de uma dimensão. Possui como parâmetros uma saída de tamanho 100 e o tamanho do *kernel* em 5, responsável por percorrer a entrada de dados. Em seguida, utilizamos a função de ativação *Rectified Linear Unit* (ReLU).
5. ***GlobalMaxPooling1D***: Reduz a dimensionalidade da entrada utilizando o maior valor possível.
6. ***Dense***: Camada totalmente conectada com 128 unidades que representam a dimensionalidade de saída. Também possui uma função de ativação ReLU.
7. ***Dense***: Complementa a camada anterior, mas possui somente 64 unidades.
8. ***Dropout***: Taxa de *dropout* novamente definida em 0,5.
9. ***Dense***: Camada densa com somente uma unidade e com função de ativação sigmoide com retorno zero ou um. Responsável por realizar a classificação de cada texto em negativo ou positivo.

Os parâmetros foram escolhidos com base em testes na implementação da rede. Para realizar o treinamento da RNC foram utilizados hiperparâmetros complementares na compilação do modelo. Para o cálculo de perda foi utilizada a função entropia

cruzada binária, retornando valores altos para previsões ruins e baixos para as boas. Foi utilizado o otimizador *RMSprop*. Por fim, seguindo a mesma linha dos trabalhos acadêmicos estudados, foi utilizada a métrica de acurácia para medir a precisão do modelo.

Para mensurar a acurácia dos modelos gerados utilizando a biblioteca *Keras*, são criadas duas variáveis, uma ***t*** para o total de textos a serem avaliados e outra ***c*** para o contador de acertos da rede. O valor final da acurácia é a divisão do contador pelo total. A Equação (2) mostra como a acurácia é calculada.

$$Acurácia = \frac{c}{t} \tag{2}$$

Todos os experimentos foram realizados com a mesma arquitetura RNC, entretanto, a camada *Embedding* da rede também foi utilizada com modelos *word embedding* provenientes das técnicas *Word2Vec*, *Doc2Vec* e *fastText*. Também avaliamos o desempenho de cada modelo em duas situações distintas, sem ajustar os pesos da camada *Embedding* e com o ajuste nos pesos.

Destacamos que em todos os experimentos os pesos das camadas de extração de características e classificação iniciaram aleatoriamente (*from scratch*), e foram ajustados durante a etapa de treinamento.

Abaixo apresentamos o resumo dos experimentos realizados:

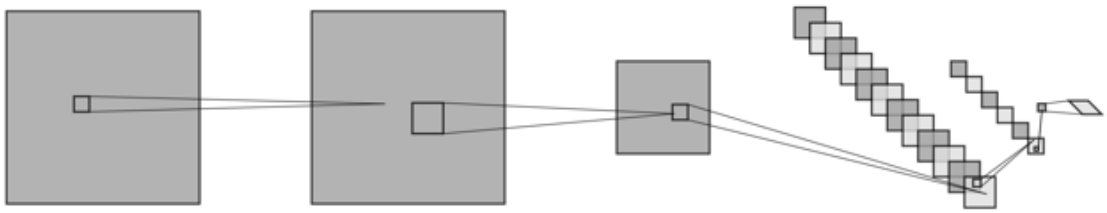
1. *Word2Vec CBOW*, com ajuste dos pesos da camada *Embedding* da RNC.
2. *Word2Vec CBOW*, sem ajuste dos pesos da camada *Embedding* da RNC.
3. *Word2Vec Skip-gram*, com ajuste.
4. *Word2Vec Skip-gram*, sem ajuste.
5. *Doc2Vec*, com ajuste.
6. *Doc2Vec*, sem ajuste.
7. *fastText*, com ajuste.
8. *fastText*, sem ajuste.
9. Rede neural convolucional, fim-a-fim.

Em síntese, a continuidade ou não do treino foi definida por um parâmetro

responsável por permitir o ajuste dos pesos carregados na camada *Embedding* da rede. Quando este parâmetro é desativado a RNC não ajusta os pesos gerados pelas técnicas *Word2Vec*, *Doc2Vec* e *fastText*. Caso o parâmetro esteja ativado, a RNC ajusta os pesos gerados pelas técnicas mencionadas acima. Quando utilizada de maneira fim-a-fim, a própria rede neural gera as *word embeddings*.

A Figura 5 exemplifica a estrutura da rede neural convolucional escolhida.

Figura 5 – Representação da estrutura da rede neural convolucional



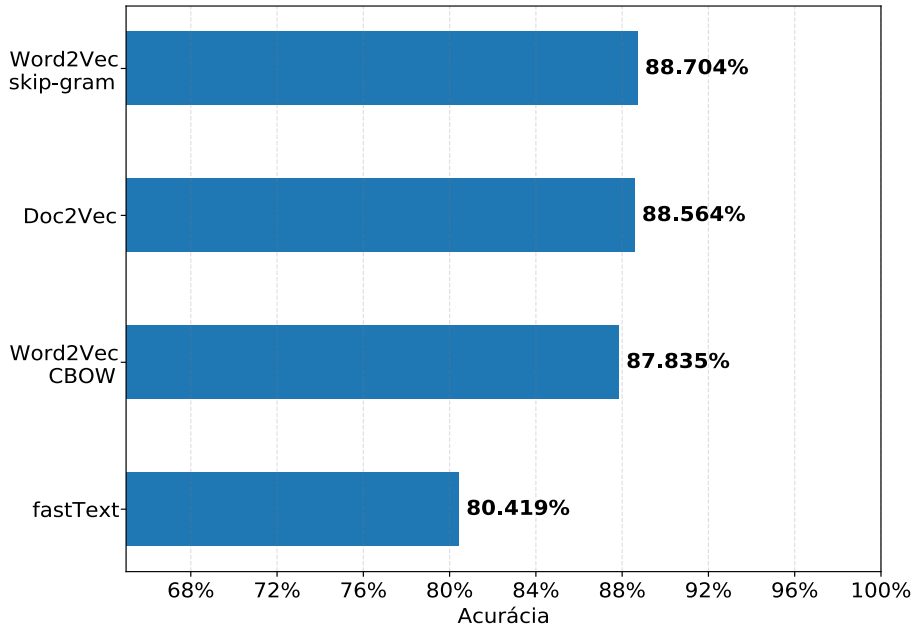
Fonte: O Autor.

## 5 RESULTADOS

Para facilitar a exposição dos resultados alcançados pelos testes na rede neural convolucional os dados foram divididos entre as Figuras 5 e 6.

Na Figura 6 é possível visualizar a acurácia obtida no conjunto de dados de validação com as técnicas *Word2Vec*, *Doc2Vec* e *fastText*, com o parâmetro de treino continuado desativado na RNC. A menor acurácia atingida foi 80,41%, pelo algoritmo *fastText*, enquanto a maior foi atingida pelo algoritmo *Word2Vec*, com o modelo *skip-gram*, com 88,70%.

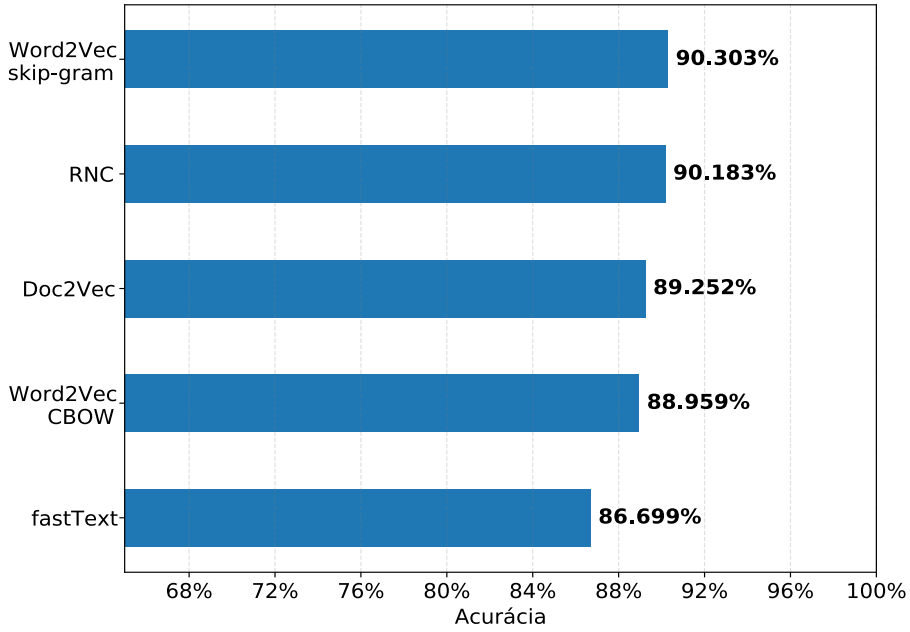
Figura 6 – Acurácia das embeddings sem ajuste pela RNC




Fonte: O Autor.

Na Figura 7, foram compilados os resultados alcançados com a utilização do ajuste de peso nas técnicas *Word2Vec*, *Doc2Vec* e *fastText* e também do resultado obtido com a *embedding* gerada com a RNC

Figura 7 – Acurácia das embeddings com ajuste pela RNC




Fonte: O Autor.

Destacamos que todos os resultados melhoraram em comparação aos expostos

na Figura 6. O menor valor continua sendo do algoritmo *fastText*, mesmo ocorrendo uma significativa melhora, com 86,69%. Novamente o algoritmo *Word2Vec* utilizando o modelo *skip-gram* atingiu a melhor acurácia, com 90,30%. A RNC fim-a-fim obteve 90,18%.

### 5.1 Comparação com o estado da arte

Como mostrado na Tabela 5, a abordagem utilizada neste trabalho superou algumas metodologias propostas na literatura para a tarefa de análise de sentimentos na base de dados "*Large Movie Review Dataset v1.0*". Os melhores resultados dos experimentos realizados (90,30% e 90,18%), utilizando a métrica acurácia, nos colocam acima dos trabalhos de Maas *et al*. (2011) e Camacho-Collados e Pilehvar (2017), com 88,89% e 88,90% respectivamente.

Tabela 5 – Comparativo dos resultados

| **Trabalho** | **Técnica** | **Acurácia** |
|---|---|---|
| Maas *et al*. (2011) | *Bag of Words* - SVM | 88,89% |
| Camacho-Collados e Pilehvar (2017) | RNC + LSTM | 88,90% |
| Nossa Proposta | RNC fim-a-fim | **90,18%** |
| Nossa Proposta | *Word2Vec Skip-gram*, com ajuste pela RNC | **90,30%** |
| Gui *et al*. (2019) | RNR + LSTM + *dynamic skip connections* | 91,10% |
| Johnson e Zhang (2015) | *seq2-bown*-RNC | 92,33% |

Fonte: O Autor.

## 6 CONCLUSÃO

Neste artigo utilizamos uma rede neural convolucional para avaliar o impacto de diferentes técnicas de *word embedding* na análise de sentimentos. Os experimentos foram realizados com os algoritmos *Word2Vec*, *Doc2Vec*, *fastText* e a própria RNC para gerar as *embeddings* e os resultados alcançados são comparados ao de trabalhos já reconhecidos no meio acadêmico.

## AGRADECIMENTOS

Agradeço a todos que direta e indiretamente contribuíram de alguma forma para que este trabalho fosse realizado, em especial aos meus pais pelas lições de vida e a todos os meus professores pelos mais diversos ensinamentos dentro e fora de sala.

## COMPARISON OF WORD EMBEDDING TECHNIQUES IN SENTIMENT ANALYSIS

**Abstract**: The significant amount of data and information that Internet users generate daily has the potential to be used in data mining and pattern recognition activities. Considering that human decision-making is highly influenced by external opinions, the information generated by users of blogs, rating and review sites and social networks can be used in sentiment analysis tasks. This paper compares the accuracy of different techniques for representing sentences in numbers, called word embeddings, using a convolutional neural network as classifier. The experiments used the set of film reviews called “Large Movie Review Dataset v1.0”. With the proposed approach, we obtained 90.30% accuracy with the best experiment.

Keywords: Natural language processing. Opinion mining. Sentiment analysis. Word embedding.

## REFERÊNCIAS

ALBAWI, Saad; MOHAMMED, Tareq Abed; AL-ZAWI, Saad. Understanding of a convolutional neural network. In: **2017 International Conference on Engineering and Technology (ICET)**. IEEE, 2017. p. 1-6.

BAKSHI, Rushlene Kaur *et al*. Opinion mining and sentiment analysis. In: **2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom).** IEEE, 2016. p. 452-455.

BBC NEWS BRASIL. Como os dados de milhões de usuários do Facebook foram usados na campanha de Trump. **BBC News Brasil**, 2018. Disponível em:

https://www.bbc.com/portuguese/geral-43705839. Acesso em: 10 jan. 2021.

BOJANOWSKI, Piotr *et al*. Enriching word vectors with subword information. **Transactions of the Association for Computational Linguistics**, v. 5, p. 135-146, 2017.

CAMACHO-COLLADOS, Jose; PILEHVAR, Mohammad Taher. On the Role of Text Preprocessing in Neural Network Architectures: An Evaluation Study on Text Categorization and Sentiment Analysis. In: **BlackboxNLP @ EMNLP**. 2018.

FAYYAD, Usama; PIATETSKY-SHAPIRO, Gregory; SMYTH, Padhraic. From data mining to knowledge discovery in databases. **AI magazine**, v. 17, n. 3, p. 37-37, 1996.

GUI, Tao *et al*. Long short-term memory with dynamic skip connections. In: **Proceedings of the AAAI Conference on Artificial Intelligence**. 2019. p. 6481-6488.

HAYKIN, Simon. **Redes neurais**: princípios e prática. Bookman Editora, 2007.

JOHNSON, Rie; ZHANG, Tong. Effective Use of Word Order for Text Categorization with Convolutional Neural Networks. In: **Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**. 2015. p. 103-112.

LE, Quoc; MIKOLOV, Tomas. Distributed representations of sentences and documents. In: **International conference on machine learning**. PMLR, 2014. p. 1188-1196.

MAAS, Andrew *et al*. Learning word vectors for sentiment analysis. In: **Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies**. 2011. p. 142-150.

MARQUESONE, Rosangela. **Big Data**: técnicas e tecnologias para extração de valor dos dados. [S.l.]: Editora Casa do Código, 2016.

MIKOLOV, Tomas et al. Efficient estimation of word representations in vector

space. **arXiv preprint arXiv:1301.3781**, 2013.

MIKOLOV, Tomas; YIH, Wen-tau; ZWEIG, Geoffrey. Linguistic regularities in continuous space word representations. In: **Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies**. 2013. p. 746-751.

O'SHEA, Keiron; NASH, Ryan. **An Introduction to Convolutional Neural Networks**. 2015.

PAK, Alexander; PAROUBEK, Patrick. Twitter as a corpus for sentiment analysis and opinion mining. In: **LREc**. 2010. p. 1320-1326.

SRIVASTAVA, Nitish *et al*. Dropout: a simple way to prevent neural networks from overfitting. **The journal of machine learning research**, v. 15, n. 1, p. 1929-1958, 2014.

TRIPATHI, Gautami; NAGANNA, S. Opinion mining: A review. In: **International Journal of Information & Computation Technology**. International Research Publications House, 2014. p. 1625-1635.

ZIGER, Gustavo. **Word Embeddings Comparison**. GitHub, 2021. Disponível em: https://github.com/zi-ger/word-embeddings-comparison. Acesso em: 26 mar. 2021.