UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
PROJETO DE GRADUAÇÃO

Lígia Iunes Venturott

ANÁLISE DE SENTIMENTOS UTILIZANDO REDES NEURAIS PROFUNDAS

Vitória-ES

Novembro/2018

Lígia Iunes Venturott

ANÁLISE DE SENTIMENTOS UTILIZANDO REDES NEURAIS PROFUNDAS

Parte manuscrita do Projeto de Graduação da aluna Lígia Iunes Venturott, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheira Eletricista.

Vitória-ES

Novembro/2018

Lígia Iunes Venturott

ANÁLISE DE SENTIMENTOS UTILIZANDO REDES NEURAIS PROFUNDAS

Parte manuscrita do Projeto de Graduação da aluna Lígia Iunes Venturott, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheira Eletricista.

Aprovado em 29 de Novembro de 2018.

COMISSÃO EXAMINADORA:

Prof. Dr. Jorge Leonid Aching Samatelo
Universidade Federal do Espírito Santo
Orientador

Prof. Dr. Patrick Marques Ciarelli
Universidade Federal do Espírito Santo
Examinador

Msc. Matheus Vieira Lessa Ribeiro
Universidade Federal do Espírito Santo
Examinador

Vitória-ES

Novembro/2018

Lígia Iunes Venturott

ANÁLISE DE SENTIMENTOS UTILIZANDO REDES NEURAIS PROFUNDAS

Parte manuscrita do Projeto de Graduação da aluna Lígia Iunes Venturott, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Aprovado em 29 de Novembro de 2018.

COMISSÃO EXAMINADORA:

Prof. Dr. Jorge Leonid Aching Samatelo
Universidade Federal do Espírito Santo
Orientador

Prof. Dr. Patrick Marques Ciarelli
Universidade Federal do Espírito Santo
Examinador

Msc. Matheus Vieira Lessa Ribeiro
Universidade Federal do Espírito Santo
Examinador

Vitória-ES

Novembro/2018

Ao meu sobrinho Francisco, que sempre me traz alegria.

AGRADECIMENTOS

Agradeço primeiramente a meus pais, por todo o apoio que me deram e por tudo que me ensinaram e que me levou a ser quem sou hoje.

Agradeço também aos meus irmãos, Pedro Henrique e Vitor, e a toda a minha família por todo o amor e carinho que sempre me deram.

Aos meus amigos, que tornaram possível chegar ao final do curso, por estarem comigo em todos os momentos de diversão e de dedicação, cansaço e preocupação.

Aos meus professores por compartilharem o conhecimento, me permitindo saber o que sei hoje. Em especial aos meus professores Patrick Ciarelli, por despertar meu interesse sobre esse tema, e Jorge Samatelo, por me orientar passo a passo nesse trabalho.

RESUMO

Análise de Sentimentos é o processo de retirar de um texto a emoção expressa. Esse é um campo de estudo que nos últimos anos tem se desenvolvido bastante, pois com o grande volume de textos produzido diariamente, a análise manual de textos se tornou inviável e com o avanço da Inteligência Artificial começa a ser possível transferir essa tarefa para as máquinas. A Análise de Sentimentos passou a ser de interesse de certos setores, como os da política e empresarial, que desejam saber como seus eleitores ou clientes se sentem em relação a eles. Porém ela também pode ser muito útil para ajudar a traçar perfis de usuários específicos, retirando informações de textos produzidos por eles. Os modelos de Análise de Sentimentos existentes consistem em treinar o computador a identificar sentimentos em frases a partir de outros textos. A grande oferta de textos disponibilizados pela Internet foi outro fator determinante para viabilizar essa tarefa.

Porém, a maior parte dos estudos nessa área são focados na língua inglesa, assim como as ferramentas para tratamento de texto, existindo assim poucos recursos para a língua portuguesa.

O presente trabalho tem como objetivo desenvolver um método de Análise de Sentimentos de textos da língua portuguesa utilizando como principal recurso Redes Neurais Profundas e criar ferramentas de pré-processamento de textos com foco na língua portuguesa.

Para representar os textos de maneira computacional foram utilizados *Embeddings*. Para a solução foi usada a rede neural LSTM, um tipo de RNN, devido a sua habilidade de guardar informações, essencial para o trabalho com dados sequenciais como textos. Foi usado também o mecanismo de *Attention*, desenvolvido para permitir que a rede foque nas partes mais relevantes do texto.

Tanto para o treinamento da rede como para a validação da rede foi utilizada a base de dados ReLi, uma base de dados estruturada formada a partir de resenhas de livros. As métricas usadas para validar o desempenho da técnica proposta serão Acurácia Simples e Acurácia Combinada (BRUM, 2015).

A solução proposta neste trabalho apresentou bons resultados quando comparada com um trabalho similar, obtendo a Acurácia Simples de $77,97\%$ e a Acurácia de Combinada $49,36\%$.

Palavras-chave: *Deep-learning*; Redes neurais, LSTM,*Attention Mechanism*, *Embeddings*, Análise de sentimentos.

ABSTRACT

Sentiment Analysis is the process of taking the expressed emotions from a text. This field of studies had a great development in the last years because, due to the great amount of texts being produced daily, manual analysis has become impracticable and, with the recent advances in Artificial Inteligence, it has become possible to assign this tasks to machines. Sentiment Analysis has become an object of interest to some sectors of society, such as companies and politicians that want to know what its electorate and consumers think about them, but it can also be used to draw profiles of specific individuals, by collecting information from texts produced by them. The existing Sentiment Analysis approaches consist in teaching the computers to identify sentiments in phrases based on other texts. The great volume of texts available on the Internet was a crucial factor in making this task possible.

However the major part of this studies focuses on the English language, as well as the tools for word processing, thus there are few resources made for the Portuguese language.

The objective of this work is to develop a method of Sentiment Analysis of text in the Portuguese language, using as its main resource Deep Neural Networks and to develop text preprocessing tools with a focus on the Portuguese language.

To represent text in a way that could be understood by computers, Embeddings were used. In the solution of the problem it was used an LSTM network, a type of RNN, due to its ability to store memory, which is essential when working with sequential data such as texts. An Attention Mechanism was also used because it was developed to allow the network to focus on the most relevant parts of the text.

The database ReLi, a structured database built from book reviews, was used for both for the network training and validation. The metrics used to validate the performance of the proposed method are the Simple Accuracy and the Combined Accuracy (BRUM, 2015).

The method proposed in this paper showed good results when compared to a similar study, having obtained $77,97\%$ in Simple Accuracy and $49,36\%$ in Combined Accuracy.

Keywords: Deep learning; neural networks, LSTM, attention mechanism, embeddings, sentiment analysis .

LISTA DE FIGURAS

Figura 1 – Arquitetura de uma RNN. 17
Figura 2 – Neurônio de uma RNN. 18
Figura 3 – Neurônio de uma LSTM. 19
Figura 4 – Fluxo principal C. 19
Figura 5 – Primeiro *gate* 20
Figura 6 – 2º e 3º *gates* 20
Figura 7 – Saída da Célula 21
Figura 8 – LSTM *Encoder-Decoder* 22
Figura 9 – LSTM Bidirecional para NLP 23
Figura 10 – Rede com *Attention* 27
Figura 11 – Rede com *Attention* para classificação de comentários tóxicos 27
Figura 12 – Etapa *Embed* 30
Figura 13 – Etapa *Encode* 31
Figura 14 – Etapa *Attend* 31
Figura 15 – Etapa *Predict* 31

LISTA DE TABELAS

Tabela 1 – Exemplo de *Embedding* de 4 palavras e 2 dimensões 25
Tabela 2 – Exemplos de frases da base de dados e suas polaridades 29
Tabela 3 – Resultados para a Análise de Sentimentos da base de dados *corpus* ReLi. 36
Tabela 4 – Comparação entre performances dos trabalhos. 37

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
CNN	*Convolutional Neural Network*
DL	*Deep Learning*
LM	*Language Modeling*
LSTM	*Long Short Term Memory*
ML	*Machine Learning*
NN	*Neural Network*
NILC	*Núcleo Interinstitucional de Linguística Computacional*
RNN	*Recurrent Neural Network*
SA	*Sentiment Analysis*
TDS	*Towards Data Science*
UFES	*Universidade Federal do Espírito Santo*

SUMÁRIO

1 INTRODUÇÃO . . . 12
1.1 Apresentação e Objeto de Pesquisa . . . 12
1.2 Justificativa . . . 14
1.3 Objetivos . . . 14
1.4 Estrutura do Texto . . . 15

2 EMBASAMENTO TEÓRICO . . . 16
2.1 Introdução . . . 16
2.2 Análise de Sentimentos . . . 16
2.3 RNN . . . 16
2.4 LSTM . . . 18
2.4.1 Componentes Básicos de uma LSTM . . . 18
2.4.2 Tipos de LSTM . . . 21
2.5 *Word Embedding* . . . 23
2.5.1 Algorítimos de *Word Embedding* . . . 24
2.5.2 Uso de *Word Embeddings* pré-treinados . . . 25
2.6 *Attention Mechanism* . . . 26
2.6.1 Implementação . . . 26
2.7 Resumo . . . 28

3 SOLUÇÃO PROPOSTA . . . 29
3.1 Introdução . . . 29
3.2 Pré-processamento . . . 29
3.3 Classificador de sentimentos . . . 30
3.4 Resumo . . . 32

4 RESULTADOS . . . 33
4.1 Introdução . . . 33
4.2 Alocação de Recursos . . . 33
4.2.1 Base de dados *corpus* ReLi . . . 33
4.2.2 Recursos Computacionais . . . 34
4.3 Metodologia de Testes . . . 35
4.3.1 *10-fold Cross-Validation* . . . 35
4.3.2 Métricas . . . 35
4.4 Experimentos . . . 36
4.4.1 Comparação de resultados . . . 36

4.5 Resumo 37

5 CONCLUSÕES E PROJETOS FUTUROS 38
5.1 Conclusões 38
5.2 Contribuições e trabalhos futuros 38

REFERÊNCIAS 40

1 INTRODUÇÃO

1.1 Apresentação e Objeto de Pesquisa

Com a expansão das redes sociais houve um grande aumento na quantidade de textos publicados. Em janeiro de 2017 foi registrada uma média de 500 milhões de *tweets* publicados por dia (OMNICORE, 2018). As redes sociais permitiram que qualquer indivíduo pudesse expressar suas opiniões e pensamentos para o resto do mundo. Essa grande massa de textos contém diversas informações que podem ser de interesse a pessoas como políticos ou empresas, que desejam saber como seus eleitores e consumidores se sentem em relação a seus produtos, campanhas, e outros. Porém, devido ao grande volume de dados, a análise manual de textos se tornou inviável, sendo necessário assim a automatização do processo via técnicas de Inteligência Artificial - IA.

A área de estudo da IA focada na compreensão de linguagem humana é denominada como *Natural Language Processing* - NLP (tradução livre, processamento de linguagem natural). Essa área objetiva a compreensão do sentido de frases humanas por computadores (BIRD EWAN KLEIN, 2009). Por exemplo, ao digitar em uma ferramenta de busca "qual a temperatura em vila velha - ES" a ferramenta devolve a temperatura na cidade desejada, como também a previsão do tempo do dia e dos próximos. Essa ferramenta de busca teve que interpretar o sentido da frase para entender o que o usuário desejava, isso é um uso de NLP.

Entre as diferentes áreas de estudo em NLP, um tema de relevância é *Sentiment Analysis* - SA (tradução livre, análise de sentimentos), o qual é definido como o processo de retirar de textos a emoção expressa neles (GUPTA, 2018). Esse é um campo de estudo que nos últimos anos tem se desenvolvido devido às necessidades já citadas e também ao avanço de áreas da IA relacionadas ao processamento de dados, especificamente a área denominada *Machine Learning* - ML (tradução livre, Aprendizado de Máquina) que consiste em "ensinar" a uma máquina a realizar tarefas como classificação ou previsão através de uma grande quantidade de dados já conhecidos. Nesse contexto, os modelos de SA baseados em ML tem como objetivo a elaboração de um modelo computacional que permita identificar sentimentos em frases a partir de outros textos. A grande oferta de textos disponibilizados pela Internet e a possibilidade de acesso a esses textos torna essa tarefa possível.

Mas, para a elaboração de modelos de ML orientados a SA é necessário uma etapa de pré-processamento dos textos usando técnicas proprias de NLP como Lematização [1], lista

[1] É o processo de reduzir palavras flexionadas à sua raiz. Exemplos: brincando - brincar, gatinho - gato.

de *Stop Words* [2], ferramentas de classificação gramatical ou semântica, entre outros (BIRD EWAN KLEIN, 2009). Essas técnicas variam de língua para língua, não sendo possível reaproveitar os já criados para outra língua. A maior parte dos estudos sobre SA são focados na língua inglesa, assim existe um déficit desses recursos para a língua portuguesa. Há algumas dessas ferramentas para o português, mas elas ainda não apresentam resultados comparáveis em desempenho às desenvolvidas para a língua inglesa.

Nesse contexto, com o intuito de superar as restrições de não contar com uma etapa sofisticada de pré-processamento para fala portuguesa, neste trabalho é proposto o uso de modelos bio-inspirados, denominados redes neurais profundas, para o problema de SA. A área de ML focada no estudo deste tipo de modelos é denominada *Deep Learning* - DL (tradução livre, aprendizado profundo). Em DL, entende-se por uma rede neural profunda como um modelo computacional composto por várias camadas de processamento não linear dos dados, onde a saída de cada camada inferior alimenta a entrada da sua camada imediatamente superior (DENG, 2014). O uso de múltiplas camadas permite que uma rede neural seja capaz de, a partir de características de baixo nível (como palavras), alcançar níveis mais altos de abstração (como identificação de sentimentos) através de uma série de estágios de processamento (WU; RAZAVI, 2015). Por esse motivo, este tipo de modelo pode atenuar o problema de uma etapa de pré-processamento não sofisticada.

Dentre as arquiteturas existentes de DL, uma das mais apropriadas para NLP são as *Recurrent Neural Network* - RNN (tradução, redes neurais recorrentes) que, por possuir uma memória de curto-prazo, se mostra uma opção melhor que outras redes mais convencionais para tarefas focadas em dados sequenciais como reconhecimento de fala e interpretação textual. Dentre os vários modelos de RNN se destaca a rede *Long Short Term Memory Networks* - LSTM, pois, diferentemente de uma RNN comum, a rede LSTM possui a capacidade de armazenar uma memória de longo-prazo, fazendo conexões entre informações mais espaçadas, como por exemplo palavras que se localizam no início e fim de uma frase, mas ainda assim compõe juntas um significado relevante. O desenvolvimento do *hardware* permitiu a aplicação de arquiteturas de Redes Neurais profundas complexas, essa evolução quando aplicada a SA produziu resultados que ultrapassaram os de métodos de ML mais antigos usados no problema.

[2] São palavras removidas na fase de pré-processamento de NLP por não conter significância na frase.

1.2 Justificativa

Aqui são descritos os trabalhos de maior importância lidos relacionados ao problema de SA. Primeiro estão descritos os artigos que fazem uso de redes RNN e, seguidamente, alguns trabalhos que fazem uso da rede LSTM para outros problemas de NLP.

Análise de sentimentos para o português usando redes neurais recursivas (BRUM, 2015) é um trabalho recente no qual o autor apresenta um modelo de treinamento para SA em sentenças do idioma português brasileiro. Nesse trabalho é implementada uma RNN tensorial desenvolvida pela Universidade de Stanford (AL, 2013).

Análise de Sentimento usando a Representação Distribuída de Parágrafos para o Português (ARAUJO, 2015) é um trabalho que tem como meta replicar os experimentos de (MIKOLOV et al., 2013) de Representação Distribuída para o português brasileiro usando como base de dados o *córpus* ReLi. Esse trabalho também implementa uma RNN tensorial.

Em ***Attention-based LSTM for Aspect-level Sentiment Classification*** (WANG MINLIE HUANG, 2016) os autores não se preocupam em achar o sentimento da frase em geral, mas o sentimento relacionado a cada aspecto da frase. Para isso é proposta uma *Attention-based LSTM* (AT-LSTM), uma LSTM com um mecanismo que captasse as partes mais importantes da frase, dado um aspecto que se deseja analisar.

LSTM Neural Networks for Language Modeling (MIKOLOV et al., 2010)é um trabalho no qual os autores utilizam uma rede LSTM para um outro problema de NLP, a modelagem de uma linguagem. Nesse trabalho eles tentam realizar a *Language Modeling* - LM (tradução livre, modelagem de linguagem) para o inglês e o francês.

1.3 Objetivos

Objetivo Geral

- O desenvolvimento de um método para a Análise de Sentimentos de textos da língua portuguesa. Para isso, pretende-se aplicar Redes Neurais Profundas.

Objetivos Específicos

- Realizar uma revisão bibliográfica de trabalhos que estudam o problema de Análise de Sentimentos;
- Estudar o estado da arte em Redes Neurais Profundas aplicadas a dados sequenciais;
- Obter um banco de textos para estudo e análise do problema;
- Implementar um sistema que permita a classificação de frases em polaridades: positiva e negativa;
- Validar e testar o sistema implementado.

1.4 Estrutura do Texto

O presente trabalho está estruturado da seguinte maneira:

- **Introdução**: este capítulo inicial tem como objetivo contextualizar o trabalho e o problema aqui estudado, apresentando o problema e ideias iniciais sobre a solução do mesmo. Além disso, são apresentados os objetivos da realização deste projeto de graduação;
- **Embasamento Teórico**: aqui serão apresentados os assuntos tratados neste trabalho, tais como: SA, as redes RNN e LSTM e o *Attention Mechanism* ;
- **Solução Proposta**: este capítulo é dedicado à apresentação da solução proposta para o problema apresentado e como foi desenvolvida.
- **Experimentos**: neste capítulo são descritos os recursos necessários para o desenvolvimento deste projeto de graduação e como esses recursos foram disponibilizados. Seguidamente são descritos os experimentos realizados e as métricas utilizadas como também os resultados desses experimentos.
- **Conclusões e Projetos Futuros**: por fim, nesse capítulo são expostas as conclusões retiradas dos trabalho realizado e possíveis projetos futuros.

2 EMBASAMENTO TEÓRICO

2.1 Introdução

O objetivo deste capítulo é estabelecer conceitos teóricos utilizados durante o trabalho, como Análise de Sentimentos e as redes RNN e LSTM.

2.2 Análise de Sentimentos

Análise de Sentimentos é a área que estuda a extração de informação subjetiva de textos, os sentimentos. O principal problema dessa tarefa é a grande complexidade que apresentam as linguagens naturais. Elas são naturalmente entendida por nós humanos, porém apresentam uma estrutura sem regras tão rígidas e sempre em mutação, o que as torna, em alguns aspectos, até ilógicas.

Algumas características presentes em todas as línguas, como a ambiguidade de palavras ou as múltiplas maneiras de escrever uma mesma expressão, tornam a linguagem natural um objeto de difícil interpretação por um modelo algorítmico.

Existem diferentes técnicas de SA. uma delas consiste em identificar palavras como positivas e negativas, atribuir-lhes um valor, e somar o valor das palavras em uma frase para identificar a polaridade da frase. A IA nesse caso é aplicada para descobrir os melhores valores para cada palavra.

Outra técnica apresentada por Pang (2002) usa ML para a tarefa de classificação. A técnica anterior se torna não funcional em frases que não utilizam palavras explicitamente negativas ou positivas para expressar sentimento, como por exemplo “Como alguém consegue assistir esse filme?”, que não possui nenhuma palavra negativa mas claramente expressa um sentimento negativo. Atualmente esse segundo método apresenta melhores resultados que o primeiro.

2.3 RNN

As *Recurrent Neural Networks* - RNN (tradução livre,redes neurais recorrentes) são redes desenvolvidas para lidar com problemas envolvendo sequências. Elas se diferenciam das

redes neurais comuns por possuir um tipo de memória de curto-prazo. Isso é feito com uma realimentação de dados das camadas através do tempo.

Uma rede neural comum é formada por camadas de neurônios. Cada neurônio possui um conjunto de entradas ligadas aos neurônios da camada anterior e um conjunto de saídas ligadas ao neurônio da camada seguinte. Desta forma cada camada realiza uma operação com suas entradas uma vez e passa o resultado para a camada seguinte.

Já numa RNN a camada recursiva repete a operação n vezes antes de passar o resultado para a camada seguinte. Cada uma das vezes a saída da camada recursiva é concatenada com a saída da camada anterior. o vetor resultante dessa concatenação volta para a camada recursiva como entrada, e esse processo é repetido n vezes, sendo n o comprimento da sequência.

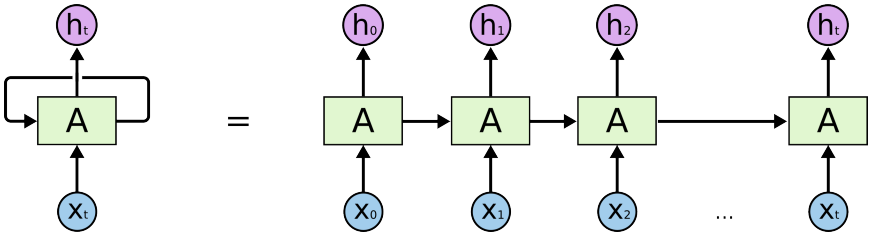
Para exemplificar, imagine uma sequência de entrada composta por cinco instantes e uma rede neural composta pelas camadas A, B e C:

$$Sequência = \{t_1, t_2, t_3, t_4, t_5\}$$

Primeiramente a camada A processa toda a sequência de entrada e passa o seu resultado para B. Inicialmente a camada B receberá como entrada a saída de A relativa à t_1, $A(t_1)$, e gerará o vetor $B(t_1)$. O vetor $B(t_1)$ não será a entrada de C. Ao invés disso $B(t_1)$ será concatenado com $A(t_2)$ e voltará para a entrada de B, que gerará $B(t_2)$. Isso se repetirá até que seja gerado o vetor $B(t_5)$ e esse sim, por fim, será a entrada de C.

Para um melhor entendimento a recursão na camada RNN pode ser representada como uma sequência de camadas, como mostra a Figura 1, em que as entradas são os $x(t)$ e as saídas os $h(t)$.

Figura 1 – Arquitetura de uma RNN.

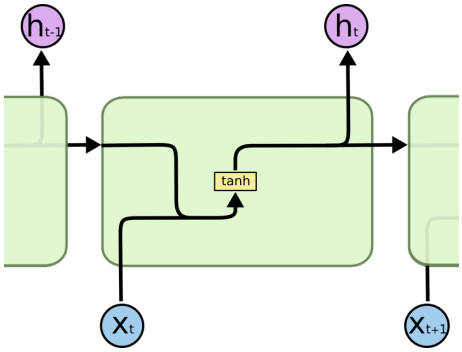


Fonte: <http://colah.github.io/posts/2015-08-Understanding-LSTMs/> em 11/05/18.

Uma célula de uma RNN funciona de maneira similar a um neurônio de uma rede neural padrão, o diferencial esta em que as entradas provindas do instante anterior são consideradas

como uma entrada a mais, tal como é mostrado na Figura 2. Esse mecanismo é chamado de memória de curto-prazo.

Figura 2 – Neurônio de uma RNN.



Fonte: <http://colah.github.io/posts/2015-08-Understanding-LSTMs/> em 11/05/18.

Dessa maneira o conhecimento do evento passado mais recente é utilizado para o cálculo presente. Essa habilidade é útil para tarefas chamadas sequenciais, como reconhecimento de voz ou análise textual, onde dados passados interferem no resultado. Por exemplo, na análise de textos, enquanto as palavras contêm certa informação individualmente, a sequência dessas palavras contêm outros tipos de informação extremamente relevantes para a compreensão textual. Sendo assim é importante fazer uma associação das palavras com as que vieram anteriormente.

2.4 LSTM

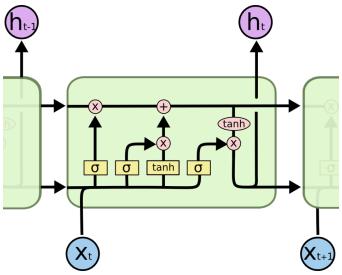
Um problema que até mesmo uma RNN tem dificuldade em resolver é o problema de dependência de longo prazo, quando o *link* de informação necessário não é entre dados sequenciais e sim entre dados mais espaçados. Esse problema é solucionado pelas redes LSTM. já que este tipo de redes são capazes de aprender informações largamente espaçadas. Esses tipos de redes relembram informações por longos períodos de tempo.

2.4.1 Componentes Básicos de uma LSTM

Assim como as RNN, explicadas na seção anterior, uma rede LSTM também possui um mecanismo de repetição no tempo, porém a estrutura interna do neurônio é construída de uma forma diferente. Essa estrutura é construída de uma maneira que a informação do instante anterior possa passar para o próximo com pequenas alterações. A célula de uma

LSTM controla essas alterações através dos *gates*. A estrutura completa de uma LSTM pode ser observada na Figura 3, onde $h(t)$ é a saída da célula LSTM em cada instante t, e $x(t)$ é a entrada vinda da camada anterior para cada instante.

Figura 3 – Neurônio de uma LSTM.

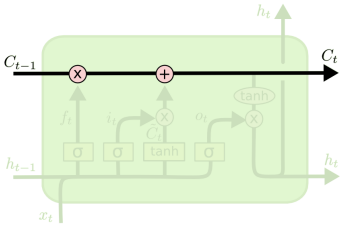


Fonte: <http://colah.github.io/posts/2015-08-Understanding-LSTMs/> em 11/05/18.

Em uma LSTM duas correntes de informação são passadas de um instante para o outro (ver Figura 3), ao contrário de uma RNN que há unicamente uma informação, a de saída das células, sendo passada.

A primeira ligação, destacada na Figura 4 é denominada C e permite que a informação passe por toda a célula e chegue na próxima célula. Esse fluxo de informação é controlado pelo *gates*, que podem ou não alterar a informação. A segunda ligação, destacada na Figura 5 é a ligação normal h_t de uma RNN.

Figura 4 – Fluxo principal C.

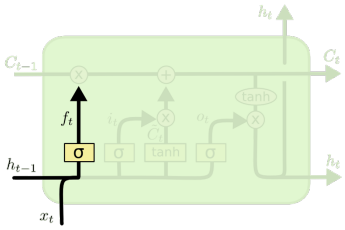


Fonte: <http://colah.github.io/posts/2015-08-Understanding-LSTMs/> em 11/05/18.

Uma célula LSTM tem três *gates*, que controlam tanto o fluxo de informação C_t para o próximo instante quanto a saída h_t do instante atual. Como é mostrado na Figura 3, cada célula tem: (i) como entrada três vetores, x_t, C_{t-1} e h_{t-1}, onde x_t é o vetor de entrada comum as redes neurais, C_{t-1} é o vetor do fluxo de informação principal, e h_{t-1} é a saída do instante anterior; (ii) como saída dois vetores, h_t e C_t, onde, h_t é o vetor de saída comum a uma rede neural qualquer e C_t é a saída do fluxo de informação.

O primeiro *gate* (ver Figura 5) multiplica C_{t-1} por uma função $f_1(x_t, h_{t-1}) = \sigma(W_f.[h_{t-1}, x_t] + b_f)$, onde $[h_{t-1}, x_t]$ são os vetores h_{t-1} e x_t concatenados. A f_1 regula quanto de C_{t-1} passará dependendo dos valores de x_t e h_{t-1}.

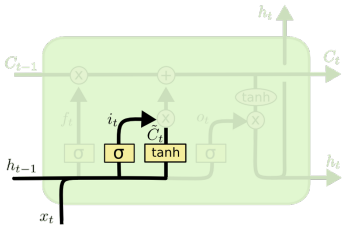
Figura 5 – Primeiro *gate*



Fonte: http://colah.github.io/posts/2015-08-Understanding-LSTMs/ em 11/05/18

O segundo e terceiro *gates* (ver Figura 6) são responsáveis por adicionar novas informações ao fluxo principal. O vetor concatenado $[h_{t-1}, x_t]$ de entrada passa pelas funções tanh e σ paralelamente. A função tanh cria um vetor de novos candidatos a serem adicionados ao fluxo principal. Já a função σ decide quais desses valores serão atualizados. O segundo *gate* multiplica os dois vetores resultantes das funções tanh e σ. O terceiro *gate* adiciona o resultado ao fluxo principal.

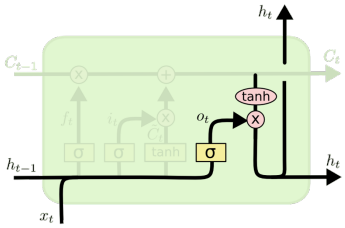
Figura 6 – 2º e 3º *gates*



retirado de http://colah.github.io/posts/2015-08-Understanding-LSTMs/ em 08/05/18

Por fim, deve-se decidir qual será a saída h_t da célula. A saída é baseada no C_t construído e no vetor de entrada (ver Figura 7). Primeiramente, o vetor de entrada passa por uma função σ para decidir quais partes de C_t irão para a saída. A própria C_t passa por uma função tanh para "encaixar" os valores entre -1 e 1. A multiplicação dos resultados é a saída h_t da função que será passada para o instante seguinte e também para as células da próxima camada em certos casos.

Figura 7 – Saída da Célula



retirado de http://colah.github.io/posts/2015-08-Understanding-LSTMs/ em 08/05/18

2.4.2 Tipos de LSTM

Existem diferentes tipos da redes LSTM na literatura. No entanto, aqui serão comentadas as seguintes arquiteturas LSTM (BROWNLEE, 2017b):

- **LSTM padrão.** É a arquitetura mais simples de uma LSTM. É constituída por: (*i*) camada de entrada; (*ii*) camada oculta LSTM completamente conectada; (*iii*) camada de saída conectada.

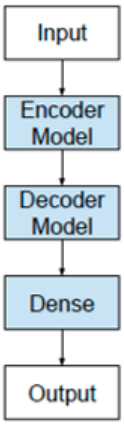
- **LSTM Empilhada.** Essa arquitetura possui múltiplas camadas ocultas LSTM, cada uma com várias células de memória. As múltiplas camadas fazem da rede uma rede profunda, que caracteriza uma arquitetura de DL. Essa maior quantidade de camadas geralmente traz resultados mais satisfatórios para problemas complexos como SA. Isso porque as camadas seguintes recombinam os resultados aprendidos pelas camadas anteriores e criam representações de maior nível de abstração (BROWNLEE, 2017a). Aumentar o número de camadas nos permite reduzir a quantidade de neurônios por camada. Isso torna o treinamento mais rápido.

- **CNN - LSTM.** É uma arquitetura que usa uma CNN como extrator de características dos dados de entrada, combinada à habilidade de memória de uma LSTM usada para predição de sequências. Esse tipo de arquitetura foi desenvolvido para o problema de predição de séries temporais visuais, um problema relacionado à vídeos. Porém, essa arquitetura também tem sido usada em problemas de reconhecimento de fala e NLP. Ela é estruturalmente formada por: (*i*) uma camada de entrada; (*ii*) camadas de um modelo CNN; (*iii*) camadas de um modelo LSTM; (*iv*) uma camada de saída.

- ***Encoder - Decoder* LSTM.** O problema de predição de sequência geralmente envolve prever o próximo valor dado uma sequência ou classificar uma sequência com um determinado rótulo. Esses problemas podem ser classificados como uma saída

para uma entrada (*one-to-one*) ou uma saída para uma sequência (*many-to-one*). Existe um outro tipo de problema mais desafiador, que é o de gerar uma sequência de saída com base numa sequência de entrada (*sequence-to-sequence* - seq2seq). Uma abordagem para a resolução desse tipo de problema é chamada de *Encoder-Decoder* LSTM. Essa arquitetura é constituída de dois módulos. Um para a codificação da sequência de entrada em um vetor de tamanho fixo, e outro para decodificar esse vetor de tamanho fixo em uma sequência de saída. Essa arquitetura foi desenvolvida para NLP e tem demonstrado um excelente desempenho. A estrutura desse tipo de arquitetura: (*i*) camada de entrada; (*ii*) camadas que constituem módulo de *Encoding*; (*iii*) camadas que constituem módulo de *Decoding*; (*iv*) camada de saída.

Figura 8 – LSTM *Encoder-Decoder*

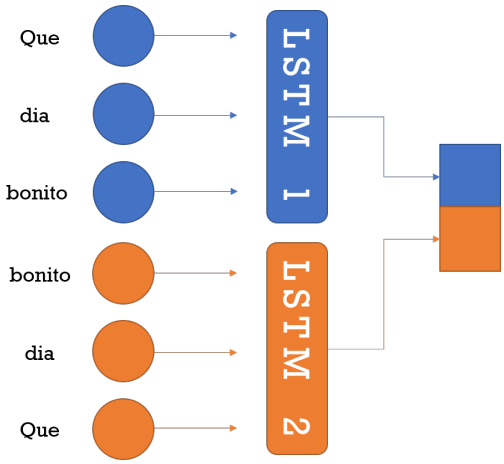


Fonte: Brownlee (2017)

- **LSTM Bidirecional.** Essa abordagem tenta aproveitar o máximo de uma sequência de entrada fazendo ela passar duas vezes por LSTM, só que de maneira inversa. Na prática, essa estratégia duplica a primeira camada recorrente da rede e passa a entrada por elas paralelamente, da segunda vez invertendo a ordem da entrada, como mostrado na figura 9.

  As saídas das 2 camadas são concatenadas célula a célula. Essa abordagem permite que cada saída da LSTM Bidirecional não possua apenas informações das células anteriores, mas também das posteriores.

Figura 9 – LSTM Bidirecional para NLP



Fonte – produção do próprio autor

2.5 *Word Embedding*

Para realizar a análise de sentimentos leva-se em conta os significados de cada palavra contida no texto, e o que essa palavra representa. Porém, uma dúvida que surge é: como representar possíveis significados ou características semânticas das palavras em um contexto computacional?

Embedding é um mapeamento de características de alguns objetos na forma de um vetor numérico de n dimensões. Cada uma dessas dimensões representa uma característica que pode ser observada no tipo de objeto descrito pelo *Embedding*, e cada valor representa a intensidade dessa característica.

Word Embedding ou *embedding* de palavras, refere-se geralmente a uma grande base de dados contendo parte das palavras de uma língua cada uma associada a um vetor n-dimensional numérico contendo suas características.

Cada dimensão do vetor tenta representar a intensidade de algum possível aspecto da palavra. Essas características são estimadas de acordo com o uso da palavra em diferentes contextos, considerando sempre as palavras em torno da palavra examinada. Tenta-se captar que sentidos e informações cada palavra pode trazer para uma frase na qual está contida.

Word Embeddings são construídos utilizando algorítimos de AI como Redes Neurais, que buscam agrupar palavras em regiões de um espaço n-dimensional procurando por

similaridades e padrões em sua utilização. A dimensão do *Embedding* é determinada de acordo com as preferências do pesquisador que irá construí-lo, podendo ser de 50, 100, 300 ou até 1000 dimensões. A grande quantidade de dimensões se deve ao entendimento que palavras, assim como qualquer parte de uma língua, são de grande complexidade, subjetivas e difíceis de analisar até pelos próprios falantes, sendo assim extremamente difíceis de representar através de números. Dessa maneira, tenta-se captar a maior quantidade de aspectos possíveis.

2.5.1 Algorítimos de *Word Embedding*

Aqui são comentados alguns tipos de *embeddings* existentes na literatura (BROWNLEE, 2017c).

- **Camada de *Embedding***. A camada de *embedding*, *Embedding Layer*, é um tipo de *word embedding* que é aprendido juntamente com um modelo de rede neural em uma tarefa específica de NLP como *Language Modeling*[1]. A base de dados para essa abordagem deve ser limpa e cada palavra representada com *one-hot encoding*. Uma camada, a camada de *embedding*, é colocada antes do modelo. A dimensão do *embedding* é especificada como parte desse modelo e os vetores são inicializados com valores aleatórios pequenos. Toda a rede é treinada de maneira supervisionada utilizando o algorítimo *Backpropagation*. Essa abordagem resulta em um *word embedding* direcionado a um tipo de base de dados e à tarefa de NLP utilizada.

- **Word2Vec**. É um algoritmo estatístico para a aprendizagem de um *word embedding* de forma eficiente. Desenvolvido em (MIKOLOV et al., 2013) esse método se tornou padrão no desenvolvimento de *word embeddings* pré-treinados, ou seja, *word embeddings* construídos de uma maneira que possam ser usados para qualquer aplicação posteriormente. O trabalho envolveu também a possibilidade de usar estrutura algébrica nas representações das palavras. Um exemplo comum é o seguinte: Considere um *word embedding* contendo as palavras "Rei","Rainha","Príncipe", e "Princesa", seria possível montar a Tabela a seguir, em que Gênero e Proximidade do Trono seriam as dimensões.

  A partir de *embeddings* pode-se estabelecer relação de proximidade entre palavras, além de outras relações. Pode-se perceber a partir da tabela acima que:

$$Rainha - Rei = Princesa - Príncipe$$

[1] *Language Modeling* é a tarefa de mapear as características de uma linguagem de forma a predizer a próxima palavra de uma sentença repetidamente, ou seja, escrever um texto.

Tabela 1 – Exemplo de *Embedding* de 4 palavras e 2 dimensões

Característica	Rei	Rainha	Príncipe	Princesa
Gênero	1	-1	1	-1
Proxm. Trono	1	1	0,6	0,6

Fonte – Prodção do próprio autor

- **GloVe**. GloVe é acrônimo de *Global Vectors for Word Representation* (tradução livre, Vetores Globais para Representação de Palavras), é uma extensão do algoritmo Word2Vec que foi desenvolvido por (PENNINGTON; SOCHER; MANNING, 2014). Métodos clássicos de representações de palavras por espaço vetorial foram desenvolvidos utilizando técnicas de fatorização de matriz, como LSA[2]. Apesar dessas técnicas serem boas em utilizar as estatísticas de palavras nos textos, elas não são tão eficientes como Word2Vec em capturar o significado das palavras ou em demonstrar suas similaridades matematicamente. GloVe é uma abordagem que tenta juntar as duas propostas. Ao invés de usar uma janela para definir o contexto local, GloVe constrói uma matriz explícita de contexto de palavras e coocorrência de palavras, o que geralmente resulta em *word embeddings* melhores.

2.5.2 Uso de *Word Embeddings* pré-treinados

Em grande parte das aplicações de NLP são utilizados *word embeddings* pré-treinados por outros pesquisadores. Isso porque o treinamento de *word embeddings* é uma tarefa que exige uma grande base de dados.

Para a utilização com redes neurais monta-se antes do modelo próprio da rede uma camada de *embedding*, e carrega-se essa camada com os pesos do *embedding* utilizado.

Há duas maneiras de usar um *embedding* pré treinado.

- **De forma Estática**. Os pesos da camada de *embedding* são fixos e permanecem os mesmos durante toda etapa de treinamento da rede.
- **De forma Atualizada**. A camada de *embedding* é preenchida com os pesos do dicionário usado, porém durante o treinamento da rede neural também os pesos da camada de *embedding* são atualizados, produzindo no final um *embedding* próprio para a aplicação.

[2] LSA é acrônimo de *Latent Semantic Analysis* (tradução livre, análise semântica latente) uma técnica para criar uma representação em vetor de um documento.

2.6 *Attention Mechanism*

Attention Mechanisms, ou mecanismos de atenção, tradução livre, consistem de métodos recentemente desenvolvidos para melhorar o rendimento de redes neurais.

Attention Mechanisms em redes neurais foram inspirados no mecanismo de atenção visual de seres humanos(BRITZ, 2016). A visão humana possui um mecanismo que foca em uma região específica da imagem, enxergando essa região em "alta resolução", enquanto enxerga as regiões no entorno em "baixa resolução". O ponto focal é ajustado a cada momento, dependendo do que nós achamos que é mais importante enxergar em "alta resolução".

No caso de tarefas de NLP, o mecanismo das redes neurais ajusta o foco dependendo da palavra analisada. Para cada palavra a ser analisada, as palavras no entorno possuem uma importância diferente.

Antes do *attention mechanism* as redes neurais para tarefas de NLP deveriam levar em conta toda a sentença para a tomada de decisões. Porém, para muitas tarefas apenas algumas partes da sentença são úteis para um determinado momento. Por exemplo, considere a seguinte tradução de português para inglês:

"O gato gordo comeu o rato." $\longrightarrow$ *"The fat cat ate the rat."*

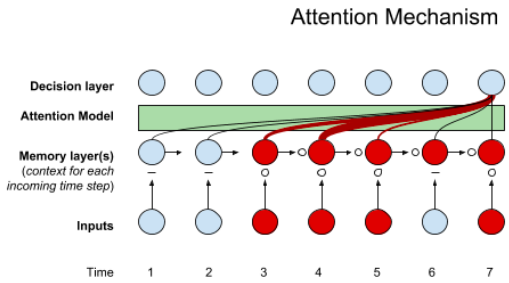
No momento em que a rede neural irá gerar a palavra *"fat"*, a importância da frase original se encontra na palavra *"gordo"* e o resto da sentença não possui muita importância. O *attention mechanism* permite focar em partes diferentes dos dados originais em momentos diferentes.

2.6.1 Implementação

Em uma RNN a saída aproveitada é geralmente a saída da última célula, pois considera-se que essa célula contém a informação de todas as células anteriores e portanto está mais apta a tomar decisões. Porém na prática um problemas com RNNs é que se a sentença for muito longa a rede tende a esquecer informações das células anteriores antes de chegar na última. Pra isso se utilizam células de memória mais complexas, como a LSTM. Apesar de apresentar uma melhora, a LSTM não elimina completamente o problema. Informações importantes ainda podem estar contidas na saída de células intermediárias. A ideia do

Attention Mechanism é levar em conta todas as saídas da RNN, como mostrado na Figura 11.

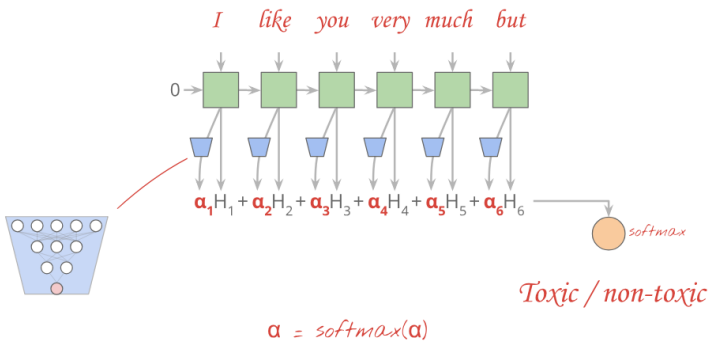
Figura 10 – Rede com *Attention*



Fonte – https://skymind.ai/wiki/attention-mechanism-memory-network em 18/11/18

No *attention mechanism* cada saída da camada RNN é multiplicada por um fator α. Os resultados dessas multiplicações então são somados gerando uma variável chamada *contexto*. Dessa forma, a variável *contexto* considera cada uma das saídas da RNN, cada uma com sua devida importância. Esse cálculo é mostrado na Figura 11 que contém um exemplo de *attention mechanism* sendo utilizado para classificação de comentários tóxicos.

Figura 11 – Rede com *Attention* para classificação de comentários tóxicos



Fonte – https://github.com/conversationai/conversationai-models/blob/master/attention-tutorial/ em 18/11/18

Os αs são calculados utilizando uma pequena rede neural, na Figura 11 sendo representada pelo trapézio azul. Para o cálculo são utilizadas as saídas da camada anterior e a entrada da camada seguinte.

Por exemplo, em uma rede neural utilizada para a tradução na seção anterior, a entrada da rede seria a frase no idioma original, português, e os valores das palavras entrariam numa camada RNN. As saídas $h(t)$ da camada seriam relativas a cada palavra, sendo t a posição de cada palavra na frase.

Para a frase seguinte, a primeira "palavra" a ser escrita seria um *token* $< s >$ indicando o início de uma frase. Chamaremos a sentença traduzida de $s(t')$, sendo t' a posição da palavra a ser escrita na nova frase. Para o cálculo dos αs são levados em conta todos os $h(t)$ já calculados pela RNN, e o $s(t')$ atual, no caso sendo $s(0)$ o *token* de início de frase $< s >$.

A tradução é feita então palavra por palavra na nova frase, sempre recalculando os αs a medida que cada palavra nova é escrita.

Por se tratar de um mecanismo de atenção, os pesos de α devem somar 1, assim é utilizada a função *softmax* no cálculo dos αs.

No caso de um problema de classificação, o *attention mechanism* permite prestar atenção nas partes com maior importância na frase, e aplicar um peso maior nelas na tomada de decisões.

2.7 Resumo

Análise de Sentimentos é um tema relevante hoje devido principalmente a grande disseminação de textos na internet e que seu desenvolvimento é dependente da Inteligência Artificial. *Embedding* é uma ferramenta extremamente útil para tarefas de NLP pois permite representar palavras por meio de vetores numéricos que contenham suas principais características. As redes LSTM são capazes de aprender informações sequenciais mesmo que espaçadas, pois possuem uma "memória" de longo-prazo, o que as torna uma ótima escolha para o trabalho com Análise de Sentimentos. As redes *attention* melhoram a performance para trabalhos com dados sequenciais pois permitem considerar o que é mais relevante nos dados. A partir dessas informações foi desenvolvida a proposta de solução descrita no próximo capítulo.

3 PROPOSTA

3.1 Introdução

Nesse capítulo é apresentada a solução implementada para a tarefa de análise de sentimentos. Tal proposta está baseada em duas etapas, especificamente:

- **Pré-processamento**. Nessa etapa o texto é tratado e preparado para servir de entrada para a rede neural.
- **Classificador de sentimentos**. Se trata da rede neural em si, as arquiteturas utilizadas nas diferentes camadas.

Cada etapa será explicada em detalhe nas próximas seções.

3.2 Pré-processamento

Esta etapa objetiva o tratamento dos textos na base de dados para deixa-la mais apropriada para as futuras etapas do projeto.

A base de dados utilizada é constituída por vários *reviews* de livros separados em frases. Cada frase possui uma polaridade atrelada a ela, positiva negativa ou neutra. Na Tabela 2 abaixo encontram-se exemplos de frases pertencentes a cada uma das polaridades.

Tabela 2 – Exemplos de frases da base de dados e suas polaridades

Frase	Polaridade
Um livro muito bom que retrata a cruel realidade dos garotos de rua da Bahia da década de 30.	positivo
Eu de verdade não sabia o que esperar do livro.	neutro
Porém algumas coisas foram desnecessárias como vampiro brilhar no sol.	negativo

Fonte – Produção do próprio autor

Na primeira etapa do pré-processamento foram separadas as frases e suas polaridades. Após a separação foram removidas das frases todas as *stopwords* encontradas. [1] Além dos *stopwords* foram removidos também palavras iniciadas com "http://", que indicavam endereços de internet, não úteis à análise, e uma gama de emoticons.

Vale ressaltar que como a base de dados foi retirada de um site de resenhas de livros, ela consiste em sua maior parte de textos na norma culta e não contém um grande número ou variedade de elementos informais como *links* e *emoticons*, o que facilitou o processo de limpeza.

Após limpas, as frases foram listadas todas as palavras diferentes contidas na base de dados e a cada uma foi associada um índice, formando um dicionário de palavras.

Nas frases limpas as palavras foram substituídas pelos seus respectivos índices, sendo a base de dados transformada em uma matriz de inteiros. Essa transformação foi feita para posteriormente ser utilizado os *Embeddings*.

3.3 Classificador de sentimentos

O classificador de sentimentos foi elaborado seguindo a proposta de (HONNIBAL, 2016) baseada em quatro etapas: *embed*, *encode*, *attend* e *predict*. Nessa metodologia a informação passa por quatro etapas bem definidas.

Na etapa de *Embedding* as palavras são mapeadas em vetores contendo suas características de maneira que possam ser entendidas pela máquina, como demonstrado na Figura 12.

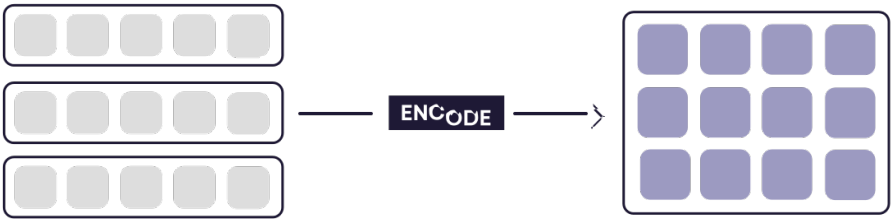
Figura 12 – Etapa *Embed*

retirado de https://explosion.ai/blog/deep-learning-formula-nlp em 22/11/18

Na etapa de *encoding* os dados da sequência são codificados em uma matriz de sequência, exemplificada na Figura 13, para que cada palavra seja representada no contexto do resto da sentença. Para essa etapa geralmente são usadas RNNs.

[1] *Stopwords* são palavras de uma língua que não acrescentam sentido próprio a uma frase e, ao invés disso, auxiliam em como outras palavras serão interpretadas. Exemplos clássicos de *stopwords* da língua portuguesa são artigos e conjunções.

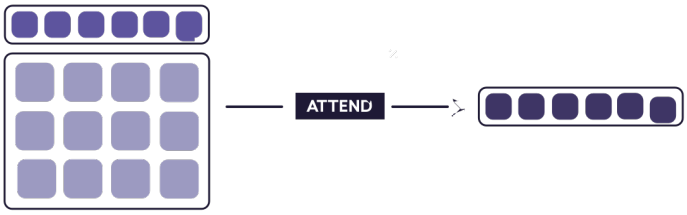
Figura 13 – Etapa *Encode*



retirado de https://explosion.ai/blog/deep-learning-formula-nlp em 22/11/18

A etapa de *attention* permite fazer um "resumo" de todas as informações da sentença, como mostrado na Figura 14. Ela possui como entrada a matriz de sequência gerada na etapa anterior e gera como saída um vetor contendo um resumo das informações da matriz, porém com um mecanismo que permite levar em conta as partes mais relevantes da matriz de entrada.

Figura 14 – Etapa *Attend*



retirado de https://explosion.ai/blog/deep-learning-formula-nlp em 22/11/18

Na etapa *predict* a rede neural aprende como deve classificar a entrada, que agora é constituída de um único vetor, em um conjunto de opções. Para isso é utilizada a arquitetura clássica *fully connected*. Especificamente, a saída é constituída de uma decisão, uma classe, como exposto na Figura 15.

Figura 15 – Etapa *Predict*

retirado de https://explosion.ai/blog/deep-learning-formula-nlp em 22/11/18

A seguir será explicada a implementação de cada etapa da metodologia *embed-encode-attend-predict*.

- **Etapa de *Embedding***. A primeira camada da rede criada foi a camada de *embedding*, em que cada palavra é substituída pelo seu vetor correspondente. Essa sequência de vetores serão as entradas da próxima camada da rede neural. Aqui foi utilizado os *embeddings* do tipo GloVe (PENNINGTON; SOCHER; MANNING, 2014) para língua portuguesa feitos por (HARTMANN et al., 2017), disponíveis em várias dimensões: 50, 100, 300, 600 e 1000.

- **Etapa de *Encode***. Aqui foi utilizada a arquitetura *Bidirectional-LSTM* para que o fluxo de informações nas camadas LSTM fosse aproveitado nos dois sentidos. As informações extraídas pela *Bidirectional-LSTM* serão passadas para uma camada de *attention* antes das camadas de classificação.

- **Etapa de *Attention***. O mecanismo de *attention* permite que a rede neural preste atenção nas partes mais importantes da frase antes de tomar decisões nas camadas seguintes. Por esse motivo foi inserido as camadas de *attention* entre a LSTM e as camadas de classificação.

- **Etapa de *Predict***. Para a classificação foram utilizadas as camadas de *GlobalMax-Pool* e uma camada *fully connected* com 3 saídas e a função de ativação *sigmoid*.

3.4 Resumo

Após descrever a implementação da etapa de pré-processamento e do classificador de sentimentos é possível compreender a importância de cada uma das etapas principais como também a importância de cada mecanismo utilizado na metodologia *embed-encode-attend-predict*. Após implementada a proposta descrita nesse capítulo, foram feitos experimentos com a base de dados. No próximo capítulo são apresentados os resultados desses experimentos como também a comparação com os resultados de trabalhos similares.

4 RESULTADOS

4.1 Introdução

Neste capítulo serão apresentados os resultados obtidos da implementação da solução proposta para o problema de Análise de Sentimentos para textos da língua portuguesa. De tal modo, o capitulo inicia descrevendo a base de dados utilizada para treinamento e teste das abordagens propostas. Também são apresentados todos os resultados obtidos em diferentes etapas do processo, mostrando a evolução obtida a partir da implementação de algumas técnicas apresentadas anteriormente. Por fim, é feita uma análise destes resultados e uma comparação com os resultados obtidos por outros trabalhos semelhantes que utilizaram a mesma base de dados com o intuito de validar as abordagens propostas.

4.2 Alocação de Recursos

4.2.1 Base de dados *corpus* ReLi

Para a realização do treinamento e validação da rede neural é necessária uma base de dados. Grande parte dos trabalhos realizados utilizam como base de dados *posts* de redes sociais como *Twitter* ou *Facebook*. Porém, por se tratar de um ambiente informal os textos são permeados de irregularidades e erros gramaticais. Essas características tornam a base de dados poluída, o que aumenta a complexidade do seu tratamento. Por esse motivo foi decidido o uso de uma base de dados composta de textos chamados "estruturados", em que há uma garantia maior de que as palavras estejam corretamente escritas.

A base de dados escolhida foi a do *corpus* Reli (Resenha de Livros), uma base formada por resenhas de livros retiradas do site Skoob (SKOOB, 2018), uma rede social brasileira para leitores. O *corpus* Reli foi criada no âmbito do projeto Anotadores Semânticos baseados em Aprendizado Ativo do Laboratório de Engenharia de Algoritmos e Redes Neurais - LEARN da PUC-Rio (FREITAS, 2014). Por se tratar de um ambiente criado com foco na literatura e dos textos serem do tipo resenha, a linguagem encontrada é menos informal, possui menos erros gramaticais e estrutura mais coerente com a norma culta. Essas características facilitam o processo de análise de sentimentos.

O *corpus* contém 1.600 resenhas, de 14 livros diferentes, sendo no total 12.248 frases e 259.978 palavras. Dessa frases já estão classificadas quanto a polaridade: 2870 positivas, 586 negativas e 8792 com opiniões neutras. Os autores dos livros avaliados são Jorge Amado, Stephenie Meyer, George Orwell, Thalita Rebouças, Jerome David Salinger, José Saramago e Sidney Sheldon.

4.2.2 Recursos Computacionais

Recursos de *Software*. Nesse projeto serão desenvolvidos programas para as etapas de pré-processamento de textos, para a implementação da rede em si, e também para o cálculo das métricas de avaliação dos resultados. Para isso são necessários alguns recursos de *software*, como determinadas linguagens de programação e bibliotecas especializadas em DL e NLP.

- **Python.** O projeto foi inteiramente desenvolvido usando a linguagem de programação Python, por ser uma linguagem com estruturas de dados que facilitam o trabalho com estruturas linguísticas e mesmo com redes neurais (BIRD EWAN KLEIN, 2009). Python é uma linguagem interpretada orientada a objeto de alto-nível. Ela foi construída para permitir ao usuário formular soluções complexas com poucas linhas de código e para ser mais legível, quando comparadas com outras linguagens existentes (ROSSUM, 1993). Devido a sua rápida curva de aprendizagem, a linguagem tem se tornado muito popular no meio científico. Como sua aquisição é gratuita ela já se tornou um componente padrão de alguns sistemas operacionais, como o Ubuntu.
- **Keras** é uma *Application Programming Interface* - API (tradução, interface de programação de aplicações) para o desenvolvimento de redes neurais escrita em Python. Ela foi desenvolvida com o objetivo de permitir rápida experimentação. Essa API suporta RNN, CNN entre outras arquiteturas.
- ***Embedding*** são grandes dicionários que contêm as características das palavras de uma língua, sendo essas características representadas por vetores numéricos. É uma ferramenta essencial para NLP pois permite a representação de palavras de forma que a máquina possa entender. Foi utilizado nesse trabalho o *embedding* GloVe feito por (HARTMANN et al., 2017) e disponível na página na internet do Núcleo Interinstitucional de Linguística Computacional - NILC. Esse *embeding* inclui 933716 palavras e está disponível em várias dimensões: 50, 100, 300, 600, 1000. Nesse trabalho foram utilizadas as versões de 600 e 1000 para os testes por apresentarem um número maior de características das palavras.

Recursos de *Hardware*. A máquina a ser utilizada nos experimentos possui a seguinte configuração: (i) sistema operacional Linux, distribuição Ubuntu 16.04 LTS Desktop; (ii) processador Intel Core i5-4440 Haswell, Cache 6MB, 3.1GHz; (iii) memória $8GB$ $1600Mhz$ DDR3; (iv) HD desktop HDD $1TB$47200RPM64MB Cache SATA $6Gb/s$; (v) placa de Vídeo Nvidia Geforce GTX 1060 com $3GB$ de memória dedicada.

4.3 Metodologia de Testes

4.3.1 *10-fold Cross-Validation*

Para os treinos e testes foi utilizado o método *k-fold Cross-Validation*. Nesse método a base de dados é dividida em k conjuntos de tamanho igual. O processo treino-teste é repetido k vezes. Cada vez um conjunto é escolhido como conjunto de teste e os outros $k-1$ são usados para treino. Dessa forma toda a base de dados é usada tanto para treino quanto para teste. As métricas finais são as médias simples das métricas de cada etapa. Nesse trabalho foi utilizado $k = 10$, ou seja, o experimento foi repetido 10 vezes.

4.3.2 Métricas

Para a avaliação dos resultados foram utilizadas as métricas Acurácia Simples e Acurácia Combinada.

A acurácia simples é calculada da seguinte forma:

$$Acurácia\ Simples = \frac{Acertos}{Total} \tag{1}$$

Já a acurácia combinada leva em consideração apenas as amostras com polaridades positiva e negativa. Ela é calculada da seguinte forma:

$$Acurácia\,Combinada = \frac{Acertos\ Positivos + Acertos\ Negativos}{Total\ de\ Sentenças\ Positivas + Total\ de\ Sentenças\ Negativas} \tag{2}$$

4.4 Experimentos

Inicialmente foram feitos experimentos apenas com as camadas *Embedding* e LSTM. Posteriormente a camada *Attention* foi introduzida, e com isso foi possível melhorar os resultados. Foram alteradas a quantidade de células M na camada LSTM para comparação de performance e foram também testados *embeddings* de 2 dimensões diferentes, 600 e 1000. Os resultados desses experimentos estão expostos na Tabela 3.

Tabela 3 – Resultados para a Análise de Sentimentos da base de dados *corpus* ReLi.

Metodologia	$AccSimples(\%)$	$AccCombinada(\%)$
Sem *Attention* 1000D $M = 10$	74,26	49,15
Com *Attention* 600D $M = 10$	76,35	46,28
Com *Attention* 1000D $M = 10$	77,12	48,77
Com *Attention* 600D $M = 20$	**77,97**	**49,36**
Com *Attention* 1000D $M = 20$	77,83	48,06

Fonte – Produção do próprio autor

Pela Tabela 3 podem ser observados alguns pontos :

- O experimento que obteve o melhor resultado foi usando *attention* com um *embedding* de 600 dimensões e com $M = 20$.
- Pode-se observar um aumento na acurácia após a inclusão da etapa de *attention*. Isso comprova a utilidade do mecanismo para tarefas de NLP.
- O aumento da quantidade de células por camada na LSTM também trouxe um aumento na rendimento do sistema, porém não muito significativo. O aumento de células de 10 para 20 traz um custo computacional maior, constituindo assim um *trade-off*, que fica a cargo do projetista decidir.
- O maior aumento no custo computacional vem do aumento de dimensões do *embedding*. O algorítimo se torna mais lento e a mudança traz pouco benefício ao resultado.

4.4.1 Comparação de resultados

Os resultados obtidos foram comparados com o trabalho de (BRUM, 2015), que se propôs a realizar a mesma tarefa de classificação sobre o *córpus* ReLi.

O trabalho de (BRUM, 2015) utiliza as mesmas métricas deste trabalho, facilitando a comparação dos resultados. Os desempenhos estão expostos na Tabela 4.

Tabela 4 – Comparação entre performances dos trabalhos.

Trabalho	$AccSimples(\%)$	$AccCombinada(\%)$
Proposta	$77,97$	$49,36$
Brum,2015	$69,08$	$51,18$

Fonte – Produção do próprio autor

Em (BRUM, 2015) é usado um modelo de Rede Neural Recursiva desenvolvida pela Universidade de Stanford para a classificação da polaridade das sentenças do *córpus*.Também é usada a metodologia 10-*fold cross-validation*. Já na abordagem deste trabalho consiste de uma etapa de pré-processamento e uma rede neural baseada na metodologia *embed-encode-attend-predict* de (HONNIBAL, 2016).

Pode-se perceber que a proposta deste trabalho produziu um resultado melhor na classificação geral com uma diferença de aproximadamente 9% na métrica Acurácia, porém na métrica Acurácia Combinada ficou aproximadamente 2% abaixo do resultado de (BRUM, 2015).

4.5 Resumo

Neste capítulo foram apresentados os experimentos feitos e as métricas utilizadas para a validação da solução proposta. Também foram apresentados os recursos necessários para a realização do trabalho, como recursos computacionais e a base de dados escolhida.

Por fim foram expostos os resultados dos experimentos e uma comparação entre os resultados deste trabalho e de um trabalho similar, assim como análises sobre o resultado.

No próximo capítulo se encontram escritas as conclusões tiradas deste trabalho e também os projetos futuros derivados deste projeto.

5 CONCLUSÕES E PROJETOS FUTUROS

5.1 Conclusões

O objetivo principal desse trabalho foi desenvolver uma solução para a Análise de Sentimentos na Língua Portuguesa utilizando redes neurais profundas. Durante o trabalho foi implementado um sistema de classificação de sentenças em três polaridades: positiva, neutra e negativa.

O trabalho permitiu: (*i*) comprovar os benefícios de usar o modelo *embed-encode-attend-predict* para a tarefa de Análise de Sentimentos; (*ii*) perceber a dificuldade de se lidar com linguagens naturais, se tratando de um campo moldado por uma junção de fatores, poucos lógicos ou físicos, como diversidade cultural, além de todas as outras características encontradas nas sociedades que formam e transformam as línguas.

A escolha de uma base de dados escrita majoritariamente em linguagem culta serviu seu papel de facilitar o pré-processamento dos textos, pois percebeu-se que um número pequeno de palavras da base de dados não puderam ser encontradas nos *embeddings*. Porém essa escolha também trouxe alguns empecilhos. Devido à característica do *site* do qual foi coletada a base de dados e da intenção dos autores ao escrever os textos, que era fazer críticas a livros, as frases apresentaram pouca ênfase nas emoções, sendo em sua maior parte neutras. Mesmo as frases classificadas como positivas e negativas não apresentavam características fortes de suas polaridades, sendo mais difíceis de classificar até por humanos.

5.2 Contribuições e trabalhos futuros

Este trabalho trouxe como contribuição a documentação do desenvolvimento de um modelo de rede neural profunda para Análise de Sentimentos para a língua portuguesa, detalhada no Capítulo 3. Ele apresenta também os resultados do modelo construído e das abordagens utilizadas, explicitados no Capítulo 4.

Tendo em mão o modelo desenvolvido surgem possibilidades de aprimorações. A rede neural pode ser modificada para tentar melhorar os resultados utilizando outros mecanismos e um outro *layout* de camadas. Outros *embeddings* podem ser testados para tentar melhorar

a performance da rede.

Um outro trabalho futuro pode ser o teste do modelo desenvolvido em outras bases de dados, para entender o efeito que as características de textos de diferentes fontes e tipos produzem na tarefa de análise de sentimento, bem como os desafios específicos de cada um.

REFERÊNCIAS

AL, R. S. et. Recursive deep models for semantic compositionality over a sentiment treebank. 2013. Citado na página 14.

ARAUJO, F. S. Análise de Sentimento usando a Representação Distribuída de Parágrafos para o Português. Monografia (TCC) — Universidade Federal do Pampa, 2015. Citado na página 14.

BIRD EWAN KLEIN, E. L. S. Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit. [S.l.]: O'Reilly Media, Inc., 2009. Citado 3 vezes nas páginas 12, 13 e 34.

BRITZ, D. Attention and Memory in Deep Learning and NLP. 2016. <http://www.wildml.com/2016/01/attention-and-memory-in-deep-learning-and-nlp/>. Acessado em 22/11/2018. Citado na página 26.

BROWNLEE, J. Long Short-Term Memory Networks With Python. [S.l.: s.n.], 2017. Citado na página 21.

BROWNLEE, J. Long Short-term Memory Networks with Python: Develop Sequence Prediction Models with Deep Learning. Jason Brownlee, 2017. Disponível em: <https://books.google.com.br/books?id=ONpdswEACAAJ>. Citado na página 21.

BROWNLEE, J. What Are Word Embeddings for Text? 2017. <https://machinelearningmastery.com/what-are-word-embeddings/>. Acessado em 20/10/2018. Citado na página 24.

BRUM, H. B. Análise de sentimentos para o português usando redes neurais recursivas. Monografia (TCC) — Universidade Federal do Pampa, 2015. Citado 5 vezes nas páginas 5, 6, 14, 36 e 37.

DENG, L. A tutorial survey of architectures, algorithms, and applications for deep learning. APSIPA Transactions on Signal and Information Processing, v. 3, p. 1–29, 2014. Citado na página 13.

FREITAS, C. Corpus ReLi. 2014. <https://www.linguateca.pt/Repositorio/ReLi/>. Acessado em 09/05/2018. Citado na página 33.

GUPTA, S. TDS. 2018. <https://towardsdatascience.com/sentiment-analysis-concept-analysis-and-applications-6c94d6f58c17>. Acessado em 11/05/2018. Citado na página 12.

HARTMANN, N.; FONSECA, E.; SHULBY, C.; TREVISO, M.; RODRIGUES, J.; ALUISIO, S. Portuguese word embeddings: Evaluating on word analogies and natural language tasks. arXiv preprint arXiv:1708.06025, 2017. Citado 2 vezes nas páginas 32 e 34.

HONNIBAL, M. Embed, encode, attend, predict: The new deep learning formula for state-of-the-art NLP models. 2016. <https://explosion.ai/blog/deep-learning-formula-nlp>. Acessado em 22/11/2018. Citado 2 vezes nas páginas 30 e 37.

MIKOLOV, T.; KARAFIÁT, M.; BURGET, L.; ČERNOCKỲ, J.; KHUDANPUR, S. Recurrent neural network based language model. In: Eleventh Annual Conference of the International Speech Communication Association. [S.l.: s.n.], 2010. Citado na página 14.

MIKOLOV, T.; SUTSKEVER, I.; CHEN, K.; CORRADO, G. S.; DEAN, J. Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. [S.l.: s.n.], 2013. p. 3111–3119. Citado 2 vezes nas páginas 14 e 24.

OMNICORE. 2018. <https://www.omnicoreagency.com/twitter-statistics/>. Acessado em 11/05/2018. Citado na página 12.

PANG, L. L. B. Thumbs up? sentiment classification using machine learningtechniques. In: Conference on Empirical Methods in Natural Language Processing (EMNLP), Philadelphia, July 2002. [S.l.: s.n.], 2002. Citado na página 16.

PENNINGTON, J.; SOCHER, R.; MANNING, C. Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). [S.l.: s.n.], 2014. p. 1532–1543. Citado 2 vezes nas páginas 25 e 32.

ROSSUM, G. van. An introduction to python for unix/c programmers. 1993. Citado na página 34.

SKOOB. 2018. <https://www.skoob.com.br/>. Acessado em 09/05/2018. Citado na página 33.

WANG MINLIE HUANG, L. Z. X. Z. Y. Attention-based lstm for aspect-level sentiment classification. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language , 2016Processing. [S.l.: s.n.], 2016. Citado na página 14.

WU, Y.; RAZAVI, R. An introduction to deep learning: Examining the advantages of hierarchical learning. Technical report, SAP affiliate company, 2015. Citado na página 13.