UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE COMPUTAÇÃO
CURSO DE CIÊNCIA DA COMPUTAÇÃO

SAMANTA DE SOUSA

ESTUDO DE MODELOS DE WORD EMBEDDING

TRABALHO DE CONCLUSÃO DE CURSO

MEDIANEIRA

2016

SAMANTA DE SOUSA

ESTUDO DE MODELOS DE WORD EMBEDDING

Trabalho de Conclusão de Curso apresentado ao Departamento Acadêmico de Computação da Universidade Tecnológica Federal do Paraná como requisito parcial para obtenção do título de "Bacharel em Computação".

Orientador: Prof. Dr. Arnaldo Candido Junior

Co-orientador: Me. Nathan Siegle Hartmann

MEDIANEIRA

2016

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Diretoria de Graduação e Educação Profissional
Coordenação do Curso de Ciência da Computação

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

TERMO DE APROVAÇÃO

ESTUDO DE MODELOS DE WORD EMBEDDING

Por

SAMANTA DE SOUSA

Este Trabalho de Conclusão de Curso foi apresentado às 13:00h do dia 16 de novembro de 2016 como requisito parcial para a obtenção do título de Bacharel no Curso de Ciência da Computação, da Universidade Tecnológica Federal do Paraná, Câmpus Medianeira. O candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Prof. Dr. Arnaldo Candido Junior
UTFPR - Câmpus Medianeira

Prof. Me. Jorge Aikes Junior
UTFPR - Câmpus Medianeira

Prof. Dr. Evandro Carlos Pessini
UTFPR - Câmpus Medianeira

A folha de aprovação assinada encontra-se na Coordenação do Curso.

RESUMO

SOUSA, Samanta de. ESTUDO DE MODELOS DE WORD EMBEDDING . 53 f. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade Tecnológica Federal do Paraná. Medianeira, 2016.

A área de Inteligência Artificial busca construir mecanismos que simulem a inteligência do ser humano de forma que os mesmos executem tarefas que os auxiliem. Tem-se o campo de estudo de Processamento de Língua Natural uma sub-área de IA que busca compreender e gerar a língua natural, dessa forma o PLN é utilizado pela IA como um meio para aprimorar os mecanismos que utilizam da língua natural na sua execução, como escrita e produção de um texto, tradução, aprendizagem e ensino entre outros. A língua segue um formato não estruturado de difícil processamento pelo computador, como as variações morfológicas e sintáticas além da ambiguidade na língua natural que dificultam o processo de compreensão, dessa forma metodologias da área convertem tais informações de forma que a manipulação das mesmas pelo computador sejam mais fáceis. Dentre as representações de informações existentes a técnica de Word Embedding está em tendência atualmente no campo de PLN, onde as informações são representadas em vetores onde os seus valores são semelhantes quando as palavras são similares, ou seja, é uma representação que codifica as relações de similaridade entre as palavras além de possuir um custo computacional baixo. Desse forma o objetivo do trabalho foi realizar um comparativo entre três modelos de Word Embeddings Cbow, Skip-gram e Glove com a finalidade de identificar qual apresenta melhor desempenho na geração dos vetores de representação das palavras (*embeddings*). Primeiramente foi realizada a construção de um *corpus* utilizando a Wikipédia em sequência foi realizado o pré-processamento dessas informações para serem utilizadas como conjunto de treinamento, os modelos foram treinados utilizando *scripts* que forma criados utilizando as bibliotecas do Python Gensim e Glove, as avaliações dos *embeddings* foram feitas com as arquivos disponíveis por Pennington et al. (2014), onde em cada avaliação/teste feito os parâmetros eram modificados afim de verificar a sua influência no desempenho dos modelos. Algumas configurações específicas para execução do treinamento dos modelos foram identificadas e relatadas no trabalho, os resultados obtidos demonstraram que o Cbow foi o modelo que apresentou melhores desempenhos na maioria dos testes. Foi verificado que a técnica de Word Embeddings codifica razoavelmente bem as informações de similaridade entre as palavras mesmo com os valores dos parâmetros sendo pequenos se comparados com outros trabalhos.

Palavras-chave: word embedding, representação de palavras, Skip-gram

ABSTRACT

SOUSA, Samanta de. STUDY WORD EMBEDDING MODELS. 53 f. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade Tecnológica Federal do Paraná. Medianeira, 2016.

The area of Artificial Intelligence seeks to construct mechanisms that simulate the intelligence of the Human beings so that they perform tasks that help them. There is the field of Natural Language Processing, an AI sub-area that seeks to understand and To generate the natural language, in this way the PLN is used by AI as a means to The mechanisms that use the natural language in its execution, such as writing and production Of a text, translation, learning and teaching among others. The language follows a format Not difficult to process by the computer, such as sd morphological variations and Syntactic as well as the ambiguity in the natural language that hinder the process of comprehension, In this way, area methodologies convert such information so that the manipulation Computer are easier. Among the information representations Existing Word Embedding technique is currently in the PLN field, where The information is represented in vectors where their values are similar when the Words are similar, that is, it is a representation that encodes similarity relations Between the words besides having a low computational cost. In this way the goal of Work was to carry out a comparison between three models of Word Embeddings Cbow, Skip- Gram and Glove with the purpose of identifying which presents better performance in the generation of Vectors of representation of words (embeddings). First, construction was carried out Of a corpus using Wikipedia in sequence, the pre-processing of those corpus Information to be used as a training set, the models were trained Using scripts that are created using the Gensim and Glove Python libraries, the Embedding evaluations were done with the files available from Pennington et al. (2014), where in each evaluation / test the parameters were modified in order to verify the Their influence on the performance of models. Some specific settings for running Of the training of the models were identified and reported in the study, the results obtained Demonstrated that the Cbow was the model that presented better performances in the majority Of the tests. It has been found that the Word Embeddings technique fairly Similarity information between words even with the values of the parameters being Small compared to other jobs.

Keywords: word embedding, word representation, Skip-gram

AGRADECIMENTOS

É difícil agradecer a todas as pessoas que de algum modo, nos momentos serenos ou apreensivos, fizeram parte da minha formação direta ou indiretamente, por isso agradeço primeiramente a todos de coração.

Agradeço a minha mãe pelo amor, incentivo e apoio incondicional nas horas difíceis de desânimo e cansaço que apesar das dificuldades me fortaleceu, ao meu namorado por me apoiar. A todos os professores do curso, que foram tão importantes na minha vida acadêmica e no desenvolvimento deste trabalho e que me proporcionaram o conhecimento não apenas racional, agradeço principalmente ao meu orientador Arnaldo Candido Junior pela dedicação, paciência e pelas muitas horas de reuniões incluindo aquelas fora do horário de trabalho em que o senhor se dispunha para me ajudar, este trabalho não estaria pronto se não fosse pela sua orientação.

Meus agradecimentos aos amigos Marcela, Diego, Patrícia, Miho, Jefferson, Tiago, Guilherme, companheiros de trabalhos e irmãos na amizade que fizeram parte da minha formação e que vão continuar presentes em minha vida, principalmente a Marcela que nos momentos de maior desespero me ajudou, obrigado a todos pela a amizade.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

BW	Bag of Words
CBOW	Continuous Bag of Words
GLOVE	Global Vectors
IA	Inteligência Artificial
MLP	Multilayer Perceptron
PLN	Processamento de Língua Natural
RFF	Redes Feed-Forward
RNA	Rede Neural Artificial
RR	Redes Recorrentes

SUMÁRIO

1 INTRODUÇÃO

Inteligência Artificial (IA) é uma ramo da ciência da computação que se propõe a compreender a capacidade humana de raciocinar, tomar decisões e resolver problemas, além disso busca-se elaborar dispositivos que simulem a capacidade de ser inteligente (ZUBEN, SD). O Processamento de Língua Natural (PLN) é uma sub-área da Inteligência Artificial onde tem-se como objetivo fazer com que os computadores executem aplicações e/ou tarefas úteis que utilizem a língua humana (transcrição da fala, tradução automática, automação de tarefas administrativas como agendas, encontros, viagens) (JURAFSKY; MARTIN, 2015).

Nas frases "Joaquim comprou um Fusca. Ele gosta de carros.", sabe-se que na segunda frase o "ele" se refere a "Joaquim", como um sistema poderia ser capaz de identificar/codificar tal informação. Dúvidas como estas estão relacionadas com a língua natural que é um dos objetos de estudo do PLN. O entendimento desta área envolve tanto conhecimento em computação como em linguística (ANNES, SD). Tal conhecimento é útil para aprimorar sistemas de conhecimento, já que a habilidade de adquirir, recuperar, explorar e apresentar criticamente o conhecimento depende da compreensão e produção da língua natural (GUREVYCH; MUHLHAUSER, SD; WAHLSTER, SD).

O estudo do campo de PLN pode ser dividido em duas categorias, ajudar o ser humano a desenvolver tarefas linguísticas ou capacitar o computador a realizar tarefas linguísticas. No primeiro caso tem-se aplicações que melhorem o desempenho das atividades de escrita e fala do ser humano ou até ferramentas de ajuda à redação e à tradução, já no segundo caso o objetivo seria a construção automática de vocabulários onde as palavras estejam organizadas pelo seu significado[1] ou dicionários podendo até realizar a síntese de notícias em várias línguas pelo telefone (ANNES, SD).

Existem diversas tarefas que de algum modo fazem uso em maior ou menor grau das capacidades linguísticas como escrita e produção de um texto, leitura e folheamento, tradução, aprendizagem e ensino entre outros. Em algumas aplicações a funcionalidades envolvendo a língua natural está integradas de forma que elas tornam o produto mais eficiente, exemplo os corretores sintáticos que geralmente vem associados a um editor de texto (ANNES, SD).

[1] Tesauros.

Como pode ser visto recursos lexicais são os alicerces de qualquer sistema de PLN, sendo necessário obter uma descrição sistemática e completa das línguas, pois algumas das dificuldades enfrentadas por esta área são as variações morfológicas e sintáticas das unidades lexicais e a ambiguidade inerente a língua natural. Dessa a forma a eficiência de tarefas e aplicações que utilizem a língua natural está diretamente ligada aos avanços no estudo de PLN. Quanto mais especializado for a língua que se pretende escrever, mais fácil é desenvolver programas.

É possível verificar que a língua natural segue um formato não estruturado de difícil processamento por computadores. Devido a isso, metodologias da área, geralmente, convertem textos em representações mais fáceis de serem manipuladas pela máquina. Em particular, a representação de vetores One Hot (ou Bag of Words) transforma palavras em vetores esparsos (vetores com grande quantidade de números com valores zero). Esses vetores perdem informações sobre similaridade entre as palavras.

A técnica Word Embeddings é baseada em modelos que buscam suprir essa limitação, gerando vetores não esparsos similares para palavras semanticamente semelhantes. Além das palavras propriamente ditas, um modelo de representação baseado em Word Embedding também pode ser usado para representar vetorialmente informações de outros níveis linguísticos, por exemplo, categorias gramaticais e funções sintáticas. Essas técnicas têm como propósito melhorar o desempenho da execução de tarefas do PLN, como: sistema de Tradução Automática, sistema de Recuperação de Informação e sistema de Auxílio à Escrita entre outros (NUNES, 2008).

O campo de representação distribuída também conhecida como Word Embedding, foi definido por Bengio et al. (2003), mas se popularizou quando a ferramenta Word2Vec com implementações de modelos de Word Embeddings Continuous Bag of Words (CBOW) e Skip-gram foi desenvolvida por Mikolov et al. (2013a). Logo em sequência Pennington et al. (2014) realiza o lançamento do Global Vectors (GLOVE) também um modelo de Word Embeddings. O grande diferencial do modelos propostos por Mikolov, são em relação a complexidade computacional necessária para a execução de tarefas.

O que assemelha esses modelos e os diferencia dos demais são as relações sintáticas e semânticas dentro da língua que são codificadas. Os modelos Cbow e Skip-gram são modelos preditivos ou seja modelos que tentam prever uma palavra de saída dada uma palavra como entrada através de uma função matemática, utilizam redes neurais artificiais para realizar o treinamento dos modelos. O Glove é um modelo de contagem básica que faz uso da matriz de co-ocorrência de palavras é uma metodologia desenvolvida por Pennington et al. (2014) para realizar o treinamento dos *embeddings*. O modelos são voltados para a produção de *embeddings*

que capturem as relações sintáticas e semânticas presente no conjunto de dados.

A proposta deste trabalho é realizar um estudo comparativo entre os seguintes modelos de Word Embeddings: Cbow, Skip-gram e Glove. Para tal, será efetuada uma pesquisa sobre esses três modelos verificando seu desempenho, suas diferenças e similaridades, o que será avaliado por meio do modelo de avaliação desenvolvido por Rodrigues et al. (2015). Para isso, será realizado o treinamento dos modelos utilizando um determinado *corpus*[2].

1.1 OBJETIVOS GERAL E ESPECÍFICOS

Este trabalho tem como objetivo geral, por meio de um estudo comparativo entre modelos de Word Embeddings, identificar qual modelo apresenta melhor desempenho em testes que avaliam a precisão dos *embeddings* gerados. Os objetivos específicos são:

- Fazer um estudo sobre as técnicas: o estudo consiste em entender o funcionamento de cada uma, verificar suas áreas principais de aplicação a partir da análise de trabalhos relacionados e levantar implementações de referência em *software*;
- Utilizar a Wikipédia como *corpus*: obter os textos dos artigos em português e fazer um pré-processamento englobando limpeza;
- Aplicar as técnicas no *corpus*: realizar o treinamento deste *corpus* utilizando as técnicas dos modelos Cbow, Skip-gram e Glove;
- Realizar a avaliação: neste processo será utilizado arquivos de modelos de avaliação disponíveis por Rodrigues et al. (2015);
- Comparar os resultados obtidos: analisando o desempenho dos testes realizados de cada modelo estudado.

[2]Um *corpus* é uma base de textos criada para uma tarefa específica (por exemplo, geração de Word Embeddings)

1.2 JUSTIFICATIVA

Os avanços das tarefas na área da IA que envolvem a língua natural estão condicionados as pesquisas e avanços dos estudos de PLN, dessa forma quanto maior for a compreensão e especificação da língua pela campo PLN mais eficientes serão as tarefas.

Word Embedding são representações de palavras muito utilizadas como base representacional nas tarefas de PLN. É atualmente um dos subcampos de estudo com grande progresso. Word Embedding consiste de que o significado das palavras está relacionado com o as palavras ao seu redor, com a frase de Firth (apud JURAFSKY; MARTIN, 2015).

> Você deve conhecer uma palavra pelo contexto que a acompanha!

Há diversos modelos de representação de palavras que foram desenvolvidos em outras áreas, os quais fazem uso das características contextuais, cada modelo possui técnicas de *embeddings* específicas que os diferencia. Sendo assim existe uma variedade de modelos que podem ser utilizados em diversas tarefas de PLN, dessa forma não se sabe ao certo quais dos modelos são vantajosos na execução de determinadas tarefas com a língua natural.

Tendo conhecimento desses fatos da existência de um grande número de modelos que utilizam informações contextuais, será feito uma pesquisa para verificar o desempenho de alguns dos modelos de Word Embedding, identificando qual apresenta melhor resultado na geração de *embeddings*. Tal conclusão será utilizada como apoio para o desenvolvimento de um projeto que envolve o a implementação de uma aplicação que recomende conteúdo, além desse projeto o estudo pode servir como incremento nos trabalhos de representação distribuída.

1.3 ORGANIZAÇÃO DO DOCUMENTO

Este documento será organizado da seguinte forma. O Capítulo 2 introduz duas subáreas da inteligência artificial: o PLN e as redes neurais artificiais. O Capítulo 3 apresenta os três modelos avaliados no trabalho e a metodologia de avaliação desses modelos. A metodologia utilizada se encontra no Capítulo 4, nele são descritas todas as etapas para o desenvolvimento do projeto e listados os recursos de *software* utilizados, bem como a forma de aquisição de cada

um. No Capítulo 5 será discutido os resultados obtidos e, por fim, no Capítulo 6 encontra-se a conclusão da proposta do trabalho de conclusão de curso.

2 INTELIGÊNCIA ARTIFICIAL

Neste capítulo, serão descritos uma introdução a área de estudo de PLN e de Redes Neurais Artificiais.

2.1 PROCESSAMENTO DE LINGUAGEM NATURAL

Línguas Naturais podem ser faladas ou escritas e provêem um meio de interação entre as pessoas. Geralmente, seu propósito é a comunicação. São o objeto de estudo do PLN, no qual o objetivo é fazer com que as máquinas compreendam as informações da língua (CHOPRA et al., 2013). Dentro do PLN, a língua é representada através de diferentes estruturas, e a captura de suas regularidades nessas estruturas é de grande importância (GOLDBERG, 2015).

O PLN possui diversas tarefas que são focos de estudos, como exemplo a tradução automática que é um dos subcampos mais antigos, datada do período da segunda guerra mundial e que tinha como objetivo traduzir as conversações gravadas em russo para o inglês. Naquele tempo, tal tarefa apresentou demasiada complexidade, pois necessitava-se do completo entendimento da língua origem para assim gerar-se a sua tradução corretamente para a língua destino. Assim como a tradução automática as outras tarefas como geradores e sumarizadores de texto também necessitam da compreensão da língua natural para que seus objetivos fossem alcançados (NUNES, 2008).

Dessa forma o estudo dentro do campo de PLN tem como objetivo possibilitar que máquinas executem tarefas úteis, relacionadas à Língua Natural, como: auxiliar a comunicação homem-máquina, fazer processamento de textos, extrair automaticamente informação, sintetizar a fala, entre outras (JURAFSKY; MARTIN, 2015). A representação formal da língua é necessária para que as máquinas possam manipular as informações (COLLOBERT; WESTON, 2008).

2.1.1 História do Processamento de Linguagem Natural

Jurafsky e Martin (2015) apresentam em seu livro os acontecimentos que deram início a área de estudo de PLN. As pesquisas na área começaram a partir da publicação de Alan Turing, cujo título é *Machine and Intelligence* (Máquina e Inteligência) e que trouxe a proposta do conhecido Teste de Turing (CHOPRA et al., 2013). Com a publicação de Chomsky (1956), envolvendo o estudo da gramática, se pôde ter uma estimativa geral da dificuldade do processamento da língua. O trabalho trouxe a Forma Normal de Backus-Naur, utilizada para especificar uma gramática livre de contexto. Estes modelos, juntamente com outros, deram origem a área de linguagens formais (NADKAMI et al., 2011). Um outro trabalho que também impulsionou os estudos de processamento de linguagem e fala foi o desenvolvimento de algoritmos probabilísticos.

Em 1972 o sistema de PLN Shrdlu foi desenvolvido sua utilização se dava por meio de uma interface que possuia um robô virtual num mundo de quadrados, tais quadrados poderiam ser movidos por meio da entrada de comandos na língua inglesa (CHOPRA et al., 2013; LIDDY, 2001). Pesquisas no PLN se intensificaram no começo da década de 1970, quando a utilização de abordagens mais gerais foi iniciada pelos sistemas afim de descrever formalmente as regras da língua (LIDDY, 2001).

2.1.2 Níveis do processamento de língua

Pardo e Nunes (2003) apresentam em seu estudo pesquisas que comprovam que um texto não é somente composto por simples sequências de sentenças e sim faz parte de uma estrutura altamente elaborada. A interpretação e o processo de escrita de textos sofre grande influência dessa estrutura que é formada por componentes linguísticos e informações de natureza extralinguística.

O PLN estuda diversos aspectos da língua natural como sons, palavras e sentenças considerando estruturas e significados, contextos e usos onde se utiliza de recursos computacionais como analisadores sintáticos para lidar com a língua (PARDO; NUNES, 2003). Sendo os aspectos de pesquisas separados em seis categorias onde os níveis de processamento da língua são apresentados conforme complexidade do tratamento computacional:

- Fonologia: neste primeiro nível, estuda-se a interpretação do som das falas e tenta-se identificar as palavras por meio dos sons (LIDDY, 2001);
- Morfologia: nesta etapa, estuda-se a composição das palavras. A morfologia tem como interesse reconhecer como as palavras base são modificadas para formar outras palavras com significado similar (a partir radicais) (LIDDY, 2001; CHOPRA et al., 2013) como exemplo o prefixo na de nadar e o sufixo de nadamos que é mos, examina-se os componentes em que a palavra é desmembrada e como isso afeta seu estado gramatical (substantivo, verbo);
- Sintaxe: nesta fase, realiza-se a análise da estrutura das sentenças da língua. Verifica-se se a estrutura analisada é válida, geralmente através de uma árvore sintática, e analisa-se a relação de dependências entre as palavras (CHOPRA et al., 2013; LIDDY, 2001), após realizar a análise morfológica identificando o estado gramatical de cada palavra que compõe a frase, com uma gramática definida é analisada a frase gerando uma estrutura dos componentes verificando se a frase está de acordo com as regras estabelecidas pela gramática, como exemplo a frase "O cão persegui o gato" onde O é artigo cão e gato são substantivos e perseguir é verbo, caso a sequência "artigo substantivo verbo artigo substantivo" esteja de acordo com as regras da gramática estabelecida a estrutura é validada;
- Semântica: este nível preocupa-se com o possível significado das palavras e das sentenças que é definido por meio das interações entre os significados cada um desses elementos (LIDDY, 2001) para isso é utilizado meios para guardar informações que caraterizem os elementos semânticos que compõem a frase em análise, como exemplo "O gato da minha sogra" onde gato é um animal, mamífero e quadrúpede dessa forma ao realizar a análise semântica com as informações dos elementos da frase o gato é definido como um animal que pertence a sogra;
- Discurso: nesta etapa se trabalha com a compreensão do texto como um todo, identificando o significado de uma determinada sentença e/ou palavra por meio da análise do contexto das sentenças que o antecedem e/ou precedem (CHOPRA et al., 2013) como exemplo "Embora você não goste, trabalhar é importante. O trabalho enobrece o homem" utilizando as análises morfológicas, sintáticas e semânticas apresentadas é obtido informações das sentenças separadamente, sendo mais complexo o tratamento computacional das informações a partir da análise semântica devido a ambiguidade inerente a língua e a natureza das informações. É possível verificar que há mais informação no trecho apresentado do que o obtido pelas análises citadas, como a relação de oposição "Embora você não goste" e "trabalhar é importante" tendo ênfase na

segunda cláusula que é reafirmada na segunda sentença “O trabalho enobrece o homem” justificando o porque de trabalhar é importante (PARDO; NUNES, 2003);

- Pragmática: utiliza informações extra-linguísticas, ou seja, o contexto no qual um texto falado ou escrito é produzido. Também determina significados que não estejam explícitos no texto analisado. Reinterpreta o que foi dito para o que realmente se queria dizer (CHOPRA et al., 2013), nesta etapa busca-se identificar as intenções subjacentes do texto, como a frase citada no exemplo anteriormente uma possível intenção seria persuadir uma pessoa a trabalhar, tais informações de relacionamentos são importantes mas poucos sistemas computacionais são capazes de reconhecê-las (PARDO; NUNES, 2003).

Dentre os níveis de processamento apresentados os modelos de Word Embeddings utilizam a análise morfológica para realizar a geração dos vetores das palavras e manter a similaridade sintática e semântica entre elas.

2.2 REDES NEURAIS ARTIFICIAIS

Cardon e Muller (1994) apresenta que, em 1943, McCullock e Pitts publicaram o primeiro estudo envolvendo Redes Neurais, no qual foi desenvolvida uma pesquisa sobre os neurônios biológicos que tinha como propósito a representação dos mesmos em forma de funções matemáticas. Tal trabalho proporcionou discussões e influenciou futuras pesquisas e desenvolvimentos sobre o assunto.

Uma Rede Neural Artificial (RNA) consiste em unidades de processamento, inspiradas nas redes neurais biológicas. São sistemas com a finalidade de solucionar problemas sem a interferência do homem, ou seja que possuam a habilidade de “aprender” a resolver problemas, por meio da extração de informações dos dados (CARVALHO, SD).

Na Figura 1, é apresentada a composição do bloco básico de uma rede neural, o modelo do neurônio artificial. Um neurônio é formado valores de entrada $x_1, x_2, \ldots, x_n$, que são ponderados pelos pesos $w_1, w_2, \ldots, w_3$, somados e acrescidos *bias* ou limiar de ativação θ. O valor obtido é então submetido a uma função de ativação, cujo resultado representa a saída do neurônio (FINOCCHIO, 2014).

Em uma rede neural artificial, os neurônios se organizam em uma estrutura de camadas, de forma que cada camada se comunica apenas com suas camadas imediatamente adjacentes (CARVALHO, SD).

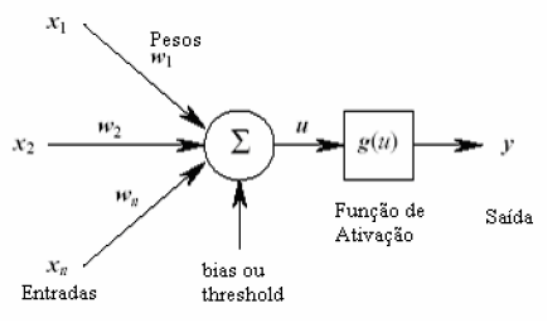

Figura 1 – Modelo de neurônio proposto por McCulloch e Pitts

Fonte: (FINOCCHIO, 2014)

Redes neurais são poderosos modelos de aprendizado e serão apresentadas duas arquiteturas de redes neurais artificiais *feed-forward* e recorrentes (GOLDBERG, 2015).

2.2.1 Redes *feed-forward*

É uma rede neural artificial estruturada em camadas, compostas cada uma por conjunto de neurônios artificiais distinto. As informações recebidas pelo meio externo apresentam somente uma direção dentro da rede *feed−forward* (RFF), que é da camada de entrada para a de saída, como pode ser visto na Figura 2. A rede *feed-forward* mais simples é composta pelos sinais de entrada e duas camadas: a de neurônios de entrada e a de neurônios de saída. Não há um tamanho fixo para os parâmetros das redes *feed−forward* (KRENKER et al., SD).

Dentre os principais tipos de redes *feed− f orward*, destaca-se o *multilayer perceptron* (MLP). Redes MLP são formadas por múltiplas camadas sendo uma camada de entrada, uma ou mais camadas escondidas e uma camada de saída. Cada camada possui nós e cada nó está totalmente interligado por pesos com todos os nós da camada subsequente (TISSOT et al., SD).

2.2.2 Redes Recorrentes

Redes Recorrentes (RR) são redes semelhantes às *feed−forward*, o que as diferencia é a retroalimentação dos neurônios. Na Figura 3, pode-se verificar que as informações possuem

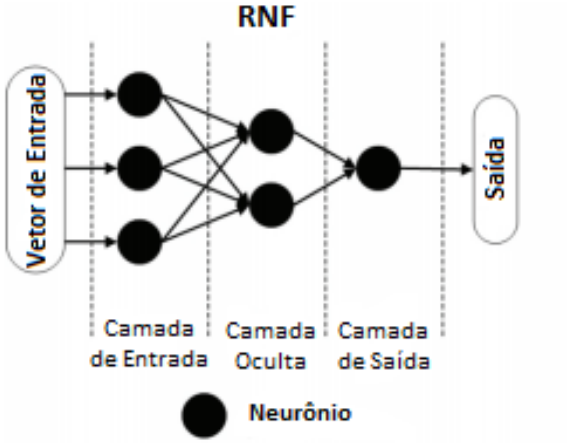

Figura 2 – Arquitetura da Rede Neural Artificial Feed-Foward
Fonte: Adaptado de (KRENKER et al., SD)

diversas direções, dessa forma a saída de um neurônio de uma camada posterior, pode ser a entrada para outro, em uma cada anterior (KRENKER et al., SD).

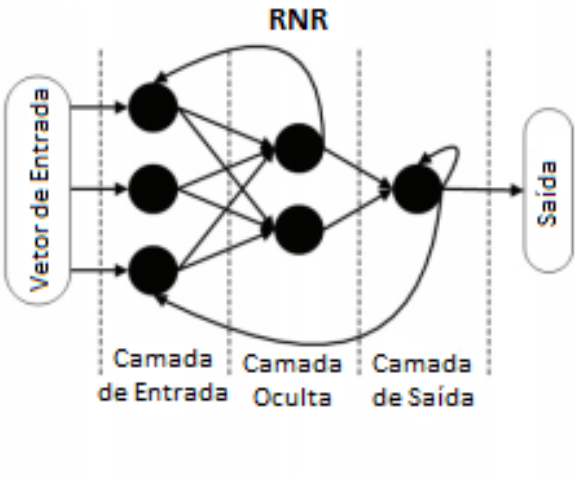

Figura 3 – Arquitetura da Rede Neural Artificial Recorrente
Fonte: Adaptado de (KRENKER et al., SD)

Backpropagation é o método de aprendizagem utilizado o qual é realizado por meio de ajuste dos pesos da rede dessa forma as diferenças entre os valores da saída obtidos e os valores da saída desejada são minimizados. De início os pesos são gerados aleatoriamente e durante o processo de aprendizagem, um vetor de entrada é apresentado para a rede e propagado para determinar os valores da saída. Em seguida, o vetor de saída produzido pela rede é comparado com o vetor de saída desejado, resultando num sinal de erro, que é retro-propagado através da rede para permitir o ajuste dos pesos. Este processo é repetido até que a rede responda, para

cada vetor de entrada, com um vetor de saída com valores suficientemente próximos dos valores desejados (OLAH, 2014).

2.3 REPRESENTAÇÃO DE PALAVRAS

Muitas tarefas de PLN possuem como entrada de dados as palavras. Devido à dificuldade em seu processamento, é comum o uso de uma representação numérica para essas palavras, a qual facilita a manipulação de informação pela máquina.

A escolha da representação de palavras influencia o desempenho de uma dada aplicação. Parte do processo para solucionar um problema no PLN consiste em encontrar qual representação de dados é mais adequada para a tarefa. Dessa forma a escolha da representação de palavras é importante.

2.3.1 Vetores One-hot

A representação One-Hot consiste em tratar as palavras como símbolos atômicos, ou seja uma representação discreta, na qual cada palavra é um atributo diferente. As palavras são diferentes umas da outras e não apresentam nenhuma relação de similaridade como exemplo a palavra “cachorro” e dissimilar da palavra “pensando” tanto quanto da palavra “gato”.

Sendo animal uma das características das palavras gato e cachorro e pensando não tendo uma correlação tão direta com essas palavras gato e cachorro apresentam maior grau de similaridade em relação a palavra pensando, tais informações de semelhança não são possíveis de serem codificadas na técnica de representação One-hot (CHAUBARD et al., 2015).

Esses vetores possuem o tamanho do vocabulário e como pode ser visto na Figura 4, todas as posições do vetor de uma determinada palavra são representadas com zeros, com exceção da posição de índice *i* no vetor que possui valor um, já que esta posição *i* está associada com uma determinada palavra com a mesma posição no vocabulário.

Dessa forma os vetores One-hot representam as palavras que compõem o vocabulário onde cada vetor possui somente uma posição com valor um que é o da palavra em questão no

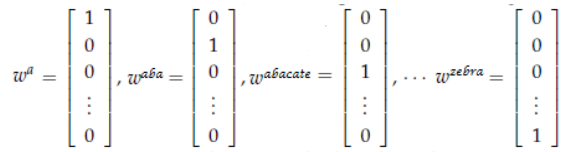

Figura 4 – Representação *One-hot*

Fonte: Adaptado de (CHAUBARD et al., 2015)l

vocabulário e todas as outras posições possuem valor zero.

Um dos grandes problemas desta representação está relacionado com a alta dimensionalidade do modelo que está diretamente ligada ao tamanho do vocabulário, o que pode elevar custo computacional (GOLDBERG, 2015).

Goldberg (2015) apresenta algumas situações nas quais podem haver ganhos com a utilização desse método como exemplo casos com poucas palavras, e nos quais tais palavras apresentem baixa correlação semântica [1] entre si, ou ainda, sob circunstâncias onde o espaço de características é pequeno e os dados de treinamento são abundantes ou não se pretende compartilhar informações estatísticas entre as palavras distintas.

2.3.2 Word Embeddings

Sistemas de PLN usam dados rotulados para aprender um modelo, mas geralmente há apenas uma quantidade limitada de texto rotulado disponível para essas tarefas. Assim, *embeddings* de palavras, que podem ser aprendidas com grandes quantidades de dados não marcados, fornecem um conjunto altamente discriminativo de recursos que permitem o aprendizado supervisionado ter desempenho melhor (DHILLON et al., 2015).

Os *embeddings* produzem características em vetores de baixa dimensionalidade, ao contrário da abordagem tradicional (DHILLON et al., 2015). Word Embedding ou representação distribuída de palavras são vetores de distribuição ou *embeddings* que possibilitam o alcance de melhor desempenho nas tarefas do Processamento de Linguagem Natural, pois não tratam as palavras como únicas, em vez disso, refletem a similaridade e dissimilaridade entre elas (MIKOLOV et al., 2013c; LEVY; GOLDBERG, 2014). Onde a troca de uma palavra por um sinônimo não deve interferir na validade da sentença exemplo nas sentenças “as meninas cantam bem” e “as moças cantam bem” a sentença deve ser válida

[1] Palavras com mesmo significado exemplo “cão” e “cachorro”

(OLAH, 2014).

Com os vetores das palavras semelhantes estando próximos é possível generalizar não somente uma frase mais sim uma classe dessas frases, permitindo dessa forma a mudança de uma palavra por uma outra em uma classe similar, exemplo “a parade é azul” por “a parede é vermelha” (OLAH, 2014).

Esse método tem como base aprender uma representação para cada palavra, chamada de *embeddings*, as palavras são mapeadas para vetores exemplo “vec(cão) = (0.0, 1.6, -1)”. O Word Embedding aprende muitas regularidade e padrões linguísticos. Seja “vec(palavra)” a representação de uma dada palavra. A equação 1 apresenta um caso de relação que pode ser aprendida (MIKOLOV et al., 2013c).

$$vec(Madri) - vec(Espanha) + vec(França) = vec(Paris) \tag{1}$$

Levy e Goldberg (2014) aponta que trabalhos propõem a representação de palavras em vetores não esparsos vem surgindo recentemente, tais trabalhos são inspirados pelo modelo de língua de rede neural de (BENGIO et al., 2003). Essas representações são apresentadas como *neural embedding* ou *word embedding*, e vem demonstrando bom desempenho em algumas tarefas do PLN.

As redes neurais artificias aprendem maneiras de representar dados automaticamente. A representação dos dados é uma etapa essencial para o sucesso em muitos problemas de aprendizagem de máquina. Os *embeddings* são apenas um exemplo particularmente marcante de aprender uma representação (OLAH, 2014).

3 MODELOS DE WORD EMBEDDING

Neste capítulo, serão descritos dois tipos de modelos de semântica distribucional[1]: o Skip-Gram e o Cbow (MIKOLOV et al., 2013a). Eles se encontram na categoria de modelos preditivos[2], também chamados de modelos neurais de linguagem, que tentam predizer uma palavra ou mais, por meio de uma rede neural simplificada, dado como entrada uma palavra ou um conjunto de palavras. Também será descrito um terceiro modelo, o Glove, da categoria de modelos baseados em contagem. O Glove conta o quão frequentemente uma palavra co-ocorre com as outras em um contexto, seja esse contexto uma janela de n-palavras, uma sentença, um parágrafo ou documento como um todo.

3.1 SKIP-GRAM

É um modelo que aprende vetores de representação de palavra utilizando redes neurais, descritas na Seção 2.2. O cálculo empregado codifica padrões linguísticos nos vetores gerados. Como exemplo, a analogia “rei está para rainha assim como homem está para mulher” é codificado em um espaço vetorial no qual a equação vec(“rei”) - vec(“rainha”) = vec(“homem”) - vec(“mulher”) (PENNINGTON et al., 2014).

O modelo Skim-gram busca, por meio de geração de um vetor de representação de palavras, predizer as palavras vizinhas que se encontram ao redor de uma determinada palavra. O modelo Skip-gram foi introduzido por Mikolov et al. (2013a), Soutner e Muller (2014). Mikolov et al. (2013b) realça em seu estudo a baixa complexidade computacional que seus modelos possuem, e como isso aprimora o tempo de treinamento em um *corpus* grande.

Conforme a Figura 5 o modelo Skip-gram possui duas matrizes a $W_{|V|xN}$ que conecta a camada de entrada a camada oculta,sendo $|V|$ o tamanho do vocabulário e N o tamanho

[1] São modelos que baseiam se pressuposto que o significado de uma palavra está relacionado com a sua utilização no cotidiano ou de sua ditribuição no texto

[2] Modelos que são treinados utilizando redes neurais artificiais

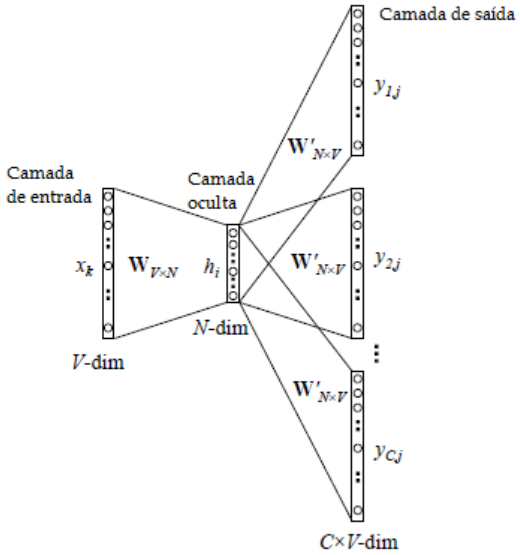

Figura 5 – Arquitetura do modelo Skip-gram

Fonte: Adaptado de (RONG, 2015)

de dimensões, e a matriz $W'_{Nx|V|}$, que liga a camada oculta a camada de saída. Para o desenvolvimento do método, primeiro tem se a entrada da palavra em representação One-Hot, que será utilizada para prever o contexto, em seguida multiplica-se a matriz W pelo vetor da palavra w_t, obtendo um vetor u_{1xN} (este vetor é o *embedding* usado na entrada da rede). A camada oculta denominada h recebe o vetor gerado pela multiplicação da matriz W com o vetor da palavra w_t $(h = u_{1xN})$.

Para cada uma das palavras contextos, multiplica-se o vetor h pela matriz W' (que também armazena embeddings, mas usados na camada de saída), como resultado do cálculo de cada uma das palavras contexto, v = hW', aplica-se a função softmax nos vetores v gerados, dessa forma obtém-se para cada palavra contexto um vetor $1x|V|$, com pontuações para cada uma das palavras do vocabulário. A seguir, o vetor one-hot da saída pode ser recuperado se desejável (JURAFSKY; MARTIN, 2015; CHAUBARD et al., 2015).

3.2 CBOW

O modelo Cbow tem como objetivo prever ou representar uma certa palavra baseando-se no contexto em que ela está contida, exemplo: "o" "gato" "na" "mesa" é tido como contexto

de entrada, e a palavra “pulou” seria provavelmente o alvo a ser alcançado.

O modelo apresenta maior simplicidade em relação aos modelos anteriores, o seu diferencial em relação ao modelo Bag of Words (BW) é a técnica de representação de palavras que utiliza representação distribuída (MIKOLOV et al., 2013a; MIKOLOV et al., 2013b).

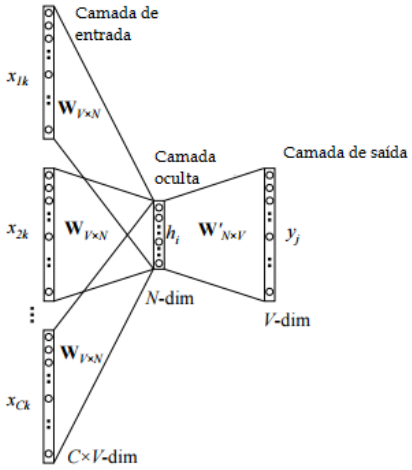

Figura 6 – Arquitetura do modelo CBOW

Fonte: Adaptado de (RONG, 2015)

O Cbow trabalha com a predição de palavras assim como o Skip-gram, a diferença entre eles e que o Cbow prediz uma palavra, conforme a janela de palavras contexto ao redor, enquanto o Skip-gram prediz o contexto a partir da palavra. O modelo tenta predizer uma palavra w_t de uma janela de treinamento.

Na Figura 6 está a arquitetura do método CBOW que é composta pela camada de entrada, camada oculta e camada de saída. A camada de entrada é conectada a camada oculta por uma matriz de contexto $W_{|V|xN}$, onde o número de colunas N é um valor arbitrário do tamanho de características, e suas linhas $|V|$ representam os vetores das palavras do vocabulário. Na camada de entrada, as palavras da janela que serão utilizadas para predizer w_j, são representadas em vetores One-Hot, os vetores são multiplicados pela matriz W, em seguida é realizada a média dos vetores *embeddings* resultantes da multiplicação, resultando em um vetor somente, obtendo dessa forma a camada oculta.

O vetor da camada oculta h é multiplicado pela matriz $W'_{Nx|V|}$ para se obter o vetor de saída z, que após a aplicação da função softmax, gera um vetor y $1x|V|$, que representa uma pontuação para cada palavra do $|V|$ (JURAFSKY; MARTIN, 2015; CHAUBARD et al., 2015).

3.3 GLOVE

Tanto o Cbow quanto o Skip-gram são modelos preditivos de semântica distribucional que utilizam a função softmax para predizer palavras. O que os diferencia do Glove é que este último é um modelo baseado em contagens de palavras (*count-based*) (BARONI et al., 2014).

No modelo Glove Pennington et al. (2014), a matriz de co-ocorrência entre palavras é utilizada para gerar *embeddings*, ou seja, o quão frequente um termo específico aparece em relação ao outro em um *corpus*. Na notação empregada, X denota uma matriz de co-ocorrência entre palavras, onde a entrada $X_{i,j}$ exibe o número de vezes que a palavra j apareceu no contexto da palavra i. E a probabilidade da palavra j ocorrer no contexto da palavra i é definida pela equação 2:

$$P_{i,j} = X_{ij} / \sum_{k=0}^{n} X_{i,k} \tag{2}$$

Sendo n o número de palavras que co-ocorrem com a palavra i. A Figura 7 mostra a probabilidade para as palavras gelo e fumaça (ice e smoke) utilizando um conjunto de palavras de contexto k, e o resultado do calculo de probabilidade entre as palavras gelo e fumaça com o contexto em questão, apresenta a relação entre elas e o contexto.

Probability and Ratio	k = *solid*	k = *gas*	k = *water*	k = *fashion*
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

Figura 7 – Exemplo de taxas de probabilidade de de uma combinação de palavras
Fonte: (PENNINGTON et al., 2014)

Como sólido é uma palavra utilizada com maior frequência no contexto de gelo, dessa forma a razão P(sólido/gelo) / P(sólido/fumaça), é alta. Já quando tem-se uma palavra relacionada com fumaça, exemplo gás, a razão deve apresentar um resultado baixo. Para palavras que estão relacionadas com as duas, como água, e palavras que não estão relacionadas com nenhuma das duas, como moda, a razão resultante do cálculo deve ser próxima de 1. Dessa forma podendo distinguir palavras que possuam relevância do ponto de vista de contexto das que são irrelevantes.

Glove é um modelo que tem como objetivo utilizar com eficiência as estatísticas obtidas de uma matriz de co-ocorrência de palavras-palavras. Esse combina os benefícios dos modelos preditivos com relação às tarefas que envolvem palavras semelhantes, e o benefício de

poder examinar informações estatísticas globais, proveniente do método de fatoração de matriz. Por meio do estudo da taxa de probabilidade de co-ocorrência de determinadas palavras com várias outras, é possível verificar o relacionamento entre elas.

Pennington et al. (2014) afirmam que a primeira fonte de informação disponível para o aprendizado de representação de palavras, por métodos não supervisionados, se encontra nas estatísticas de um *corpus*.

$$F(w_i^T \tilde{w}_k) = P_{ik} = \frac{X_{ik}}{X_i} \tag{3}$$

Na Equação 3 é apresentada uma função, que realiza ajustes nos *embeddings* das palavras, que em primeira instância são gerados aleatoriamente. Ajustes são feitos de forma que o produto escalar de seus vetores, seja semelhante ao seus valores na tabela de co-ocorrência.

É possível verificar que cada palavra possui dois *embeddings* $\tilde{w_k}$ e $\tilde{w_i}$. A função F não é simétrica, pois tem-se que $F(w_i, \tilde{w_k})$ é diferente de $F(\tilde{w_k}, w_i)$, já que as probabilidades de uma determinada palavra w_i ocorrer no contexto de outra palavra w_k não é a mesma probabilidade quando se inverte as palavras. Como exemplo a palavra “tinto” possui como contexto “vinho” quase 100% de tempo, já ao contrário não verdadeiro. Dessa forma, F não pode ser utilizada como uma função de semelhança entre *embeddings*.

Para sanar esse problema, é proposta a função F', que adiciona dois bias (um para w_i e outro para w_k) que devem ser aprendidos pelo modelo de modo a garantir simetria. A equação 4 oferece simetria. Adicionalmente, F' reflete os logaritmos dos valores da Figura 8, já que é mais eficiente e preciso, computacionalmente, trabalhar com probabilidades na forma de logaritmos.

$$F'(w_i, w_k) = w_i^T \tilde{w}_j + b_i + \tilde{b_j} \tag{4}$$

Existe um problema de otimização quadrática, onde os *embeddings* sofrem ajustes para minimizar a soma. A equação J é a utilizada pelos autores Pennington et al. (2014) para realizar a geração dos *embeddings*:

$$J = \sum_{i,j=1}^{V} N(X_{ij})(F'(w_i, w_j) - log(X_{ij}))^2. \tag{5}$$

$$N(x) = \begin{cases} (x/x_{max})^a & \text{se } x < x_{max} \\ \\ 1 & \text{caso contrário.} \end{cases} \tag{6}$$

Na Equação 6 a função N, realiza a contagem da matriz de coocorrência e é utilizada em palavras raras ou pouco frequentes. Realiza um pequeno ajuste, pois todas as ocorrências são pesadas igualmente, dessa forma co-ocorrências com baixa frequência são consideradas ruídos que carregam poucas informações.

É realizado um somatório que percorre todo o vocabulário, entre, a primeira parte e a segunda parte, que se encontra uma função que estima um valor de relação entre as palavras, utilizando os *embeddings* das palavras que são definidos pela P(i,j).

3.4 CÁLCULO DOS MODELOS PREDITIVOS

Os três modelos que foram utilizados no trabalho foram apresentados e descritos anteriormente. Sa Seção 3.3 foi demonstrado o funcionamento do modelo Glove e a equação utilizada para geração dos *embeddings*, já nas Seções 3.1 e 3.2 foi descrito o funcionamento dos modelos Cbow e Skip-gram e não o processo para gerar os *embeddings* devido a uma complexidade computacional que a função Softmax que é a utilizada pelos modelos apresenta. Dessa forma esta seção tem como objetivo apresentar como é gerado os *embeddings* dos modelos preditivos.

Esses modelos tem como finalidade calcular a probabilidade de uma determinada palavra dado o contexto em que ela está inserida. Utilizando a equação softmax em uma rede neural é possível obter a a probabilidade de uma sentença completa por meio dos cálculos de probabilidade de cada palavra dado as palavras contextos da mesma (RUDER, 2016).

Os modelos preditivos Skip-gram e Cbow utilizam a equação 7 para estimar a probabilidade de uma palavra alvo w_t ocorrer dado que um contexto h observado.

$$P(w_t|h) = softmax(score(w_t, h)) \tag{7}$$

Como pode ser visto em Harp et al. (2016), tem-se como entrada na equação, os

embeddings da palavra w_t e do contexto h. Quando h contém mais de uma palavra, é realizada um soma entre esses vetores resultando em uma pontuação, que é a probabilidade de ocorrência dessa palavra em relação ao contexto. Em primeira instância, esse valor é errado e não condiz com o esperado, dessa forma são realizadas ajustes nos *embeddings* de forma que a Equação 2 possa estimar razoavelmente as probabilidades.

A função *score*, ou pontuação, tem como finalidade verificar a semelhança entre dois vetores. Qualquer medida de similaridade pode ser utilizada, mas, geralmente, é usado o produto escalar[3].

A função softmax é usada como uma normalização vetorial[4] de um vetor construído com o *score* entre h e todas as palavras do vocabulário, incluindo w_t . Seu propósito é realçar componentes com valores elevados, pode ser interpretada como probabilidade, e deseja-se que o maior valor seja w_t. Assim, a função da Equação 8 pode ser reescrita como:

$$P(w_t|h) = \frac{e^{s(w_t,h)}}{\sum_{w_i \in W} e^{s(w_i,h)}} \tag{8}$$

Onde W é o conjunto de palavras do vocabulário. Para melhor compreensão da equação, por exemplo, seja a palavra w_t, "cachorro" e o contexto h, "latiu" (neste exemplo é utilizado o contexto tamanho um, mas é possível utilizar contextos com tamanhos maiores) e um vocabulário que vai da palavra "abacate" até "zebra", sendo os vetores dessas palavras gerados aleatoriamente[5], a probabilidade seria calculada da seguinte forma:

$$P(cachorro|latiu) = \frac{e^{s(cachorro,latiu)}}{e^{s(abacate,latiu)} + \ldots + e^{s(cachorro,latiu)} + \ldots + e^{s(zebra,latiu)}} \tag{9}$$

Onde após a função *score* realizar o produto escalar entre os vetores das palavras, sendo na parte superior da equação o vetor "cachorro" comparado com o vetor "latiu" e na parte inferior são os vetores de todas as palavras do vocabulário comparadas com o de "latiu". Para um contexto maior, se a palavra alvo fosse "colchão" e o contexto "o gato sentou no", w_t seria $vec(colchao)$ e h seria $vec(o) + vec(gato) + vec(sentou) + vec(no)$. Aplica-se a função softmax no vetor resultante construído com scores, supondo que o vetor de scores seja (1, 2, 3, 4, 1, 2, 3), aplicando a função softmax, temo-se como resultado (0.024, 0.064, 0.175, 0.475, 0.024, 0.064, 0.175). O processo busca maximizar as palavras preditas w_t, o que é alcançado por meio dos ajustes dos vetores de *embeddings*.

[3]Função binária entre dois vetores que fornece um número real, ex. dados os vetores $u = (u_1, u_2)$ e $v = (v_1, v_2)$, o produto escalar é definido pelo valor obtido por: $u.v = u_1.v_1 + u_2.v_2$.

[4]Normaliza valores de saída entre 0 e 1.

[5]Nos modelos de Mikolov et al. (2013a) não são utilizados vetores totalmente aleatórios no início do processo; ao invés disso, embeddings iniciais são calculados a partir de uma abordagem simples baseadas em trigramas.

3.5 OTIMIZAÇÃO DA EFICIÊNCIA COMPUTACIONAL

Nos modelos Skip-gram e Cbow, o cálculo de probabilidade utilizado, conhecido como softmax, apresenta um custo computacional caro, por estar diretamente ligado ao tamanho do vocabulário, que pode chegar na casa dos milhões, sendo impraticável nestas situações. Para resolver este problema algumas abordagens podem ser utilizadas para otimizar o cálculo.

3.5.1 *Softmax* Hierárquico

O modelo *Softmax* apresenta demasiada complexidade computacional, devido a isso, uma alternativa é utilizar a técnica *Softmax* Hierárquico (MORIN; BENGIO, 2005 apud GOLDBERG; LEVY; LOPES; MIKOLOV et al., 2014, 2015, 2013b) para diminuir a complexidade. Dentre diversos tipos de árvores que poderiam ser utilizadas no Softmax Hierárquico, a implementação word2vec (apresentada na seção 3.6) utiliza a árvore de Huffman para um treinamento mais rápido dos modelos. Na árvore binária todas as palavras do vocabulário são nós folhas, e para cada nó folha existe somente um caminho da raiz para o nó em questão; e este caminho é utilizado para estimar a probabilidade da palavra representada, pelo nó folha (RONG, 2015). Sendo a probabilidade de uma palavra, dividida em uma sequência de probabilidades.

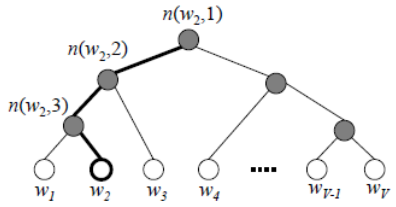

Figura 8 – Árvore binária representando as palavras do vocabulário
Fonte: (RONG, 2015)

O cálculo da probabilidade de uma palavra w_2 dado um contexto, é alcançado por meio da aplicação da função softmax em cada nó interno até chegar na palavra.

Dessa forma, a complexidade computacional do cálculo da probabilidade de uma

palavra na árvore é reduzida, por meio da diminuição de unidades de saída que necessitam ser avaliadas Figura 8, ou seja, em vez de a formula ser aplicada para todas as palavras do vocabulário em cada instância de treinamento, são atualizadas somente os nós caminho para a palavra em questão. Seja N o número de palavras no vocabulário, qualquer palavra pode ser especificada por uma sequência de decisões binárias em O(log N).

3.5.2 Amostras Negativas

É um método que busca diminuir a complexidade do cálculo de probabilidade dos modelos, por meio da redução de vetores que devem ser atualizados a cada iteração de treinamento (RONG, 2015). Essa nova abordagem proposta por Mikolov et al. (2013c) chamada Amostras Negativas demonstrou eficiência na criação de Word Embeddings.

Amostras Negativas são uma simplificação do modelo *Noise Contrastive Estimation* (NCE) que mantém a qualidade de representação dos vetores. O NCE expõe que um bom modelo deve saber diferenciar dados de ruídos.

No modelo são geradas amostras positivas e negativas, pares de palavras e contextos. As positivas são extraídas do *corpus*, enquanto que as negativas são definidas utilizando uma distribuição ruído que sorteia palavras e contextos aleatórios, com maior probabilidade para palavras frequentes do vocabulário. A tarefa do modelo é desenvolver uma função objetivo que maximize a probabilidade de um par de palavra e contexto estar no *corpus*, se realmente estiverem, e maximizar a probabilidade de um par não estar no *corpus*, se eles forem sorteados aleatoriamente (CHAUBARD et al., 2015).

3.6 BASE DE AVALIAÇÃO DOS WORD EMBEDDINGS

Nesta seção é apresentado o estudo utilizado como base para realizar as avaliações dos *embeddings* gerados pelos modelos Cbow, Skip-gram e Glove.

Mikolov et al. (2013a) desenvolveu um conjunto de teste para avaliar Word Embeddings, o qual apresenta com detalhes em seu trabalho. Esse conjunto é formado por uma coleção de analogias, contendo quatro palavras análogas em sequência, como: Berlim

Alemanha Lisboa Portugal, que são interpretadas como 'Berlim está para Alemanha, como Lisboa está para Portugal'.

Rodrigues et al. (2015) traduzem para o Português e disponibilizam esses dados, que serão empregados neste trabalho. Durante o processo, os autores fizeram algumas mudanças. O conjunto de teste utilizado contém cinco tipos de analogias semânticas e nove tipos de analogias sintáticas. As analogias semânticas são: países e capitais comuns, todos os países e capitais, moedas oficiais, capitais dos estados, relações familiares. As sintáticas: adjetivo para advérbio, antônimos, comparativo, superlativo, particípio presente, adjetivos de nacionalidade, pretérito do passado, plural dos substantivos e plural dos verbos.

Durante o processo de tradução do arquivo, algumas circunstâncias foram encontradas, nas quais algumas palavras do inglês não poderiam ser traduzidas precisamente, dessa forma afetando na analogia em questão, resultando em um conjunto de testes com menos analogias que o original. As analogias que sofreram alterações foram relações familiares, adjetivos para advérbios, antônimo, comparativo e superlativo. Exemplos podem ser encontrados em Mikolov et al. (2013a).

Além disso, Rodrigues et al. (2015) também testaram a indução de vários modelos baseados no Skip-gram. Cada modelo foi treinado com parâmetros diferentes. A execução do trabalho é composta pelos seguintes processos: adquirir o *corpus*, montar conjunto de treinamento, aplicar script de treinamento e fazer a avaliação.

Foram testados dezenove *corpora*[6], totalizando 121.706.288 sentenças. Alguns dos *corpora* foram obtidos por organizações que disponibilizam tal conteúdo, outros utilizaram um *web crawler* para obtenção do conteúdo textual. Os que utilizaram *web crawler*[7] para captação de texto passaram por um processo para poderem ser utilizados, onde foi realizada a extração dos dados e em seguida o pré-processamento, que realizou a limpeza nos textos, retirando tags ou outro conteúdo que o autor julgar indesejado. Por fim, é feita a tokenização, onde foi utilizado o LX-Tokenizer um programa desenvolvido na Universidade de Lisboa.

O script de treinamento e avaliação foi desenvolvido utilizando a biblioteca Gensim do Python, que contém a ferramenta word2vec[8], onde está implementado os modelos Cbow e Skip-Gram e diversos métodos utilizados nos modelos, dentre eles um que treina o modelo e outro que avalia a precisão dos *embeddings* gerados no treinamento do modelo.

No trabalho de Rodrigues et al. (2015) o objetivo do estudo é encontrar o Word Embedding com maior precisão, resultado que é alcançado pelos autores, por meio de cinco tipos de testes, os quais eles descrevem especificamente, que quando realizados, ou seja,

[6]Plural de *corpus*

[7]É um programa que navega na rede de Internet de maneira automática, muito utilizado como mecanismos de buscas com o propósito de criar uma cópia com todas as páginas visitadas

[8]http://code.google.com/p/word2vec/

quando aplicados no modelos já treinados, fornecem a informação de qual corpora e com quais parâmetros definidos, pode-se chegar ao resultado com maior taxa de precisão. Já no trabalho aqui desenvolvido a finalidade do estudo é avaliar qual modelo de Word Embedding apesenta melhor desempenho, utilizando para treinamento um conjunto de dados composto por textos da Wikipédia de tamanho de 1GB, e para avaliação faz se uso de conjunto de testes de (RODRIGUES et al., 2015).

Todos os testes são realizados com e sem restrições de palavras (Função implementado na ferramenta Word2vec, onde a frequência das quatro palavras que compõem as analogias, devem ser acima de 30.000 no vocabulário). Onde na maioria dos testes a utilização do *corpus* da Wikipédia apresentou maior valor na precisão dos *embeddings*, a avaliação dos *corpus* sendo eles incrementados um ao outro em sequência foi melhor que a avaliação dos *corpus* separadamente.

Dentre os parâmetros existentes no método, os utilizados pelos autores foram o tamanho da dimensão, tamanho da janela de palavras de contexto, taxa de aprendizado e os métodos de otimização disponíveis Amostras Negativas e Softmax Hierárquico. O valor dos parâmetros verificados que apresentaram maior precisão foram: dimensões com tamanho 300 e 400, método Amostras Negativas com tamanho quinze, janela de contexto igual a dez e taxa de aprendizado 0,025.

4 MATERIAL E MÉTODOS

Nesse capítulo será descrita a metodologia utilizada para o desenvolvimento deste projeto. Serão descritas as etapas do projeto e os principais fundamentos e tecnologias a serem empregados. O desenvolvimento do trabalho será composto pelos seguintes passos:

1. Utilizar a Wikipédia como *corpus*;
2. Pré-processamento dos dados de treinamento;
3. Treinamento e avaliação dos modelos;
4. Realizar comparativo de desempenho dos modelos.

4.1 DESCRIÇÃO DAS ETAPAS

Será descrito todo o processo para a geração de Word Embeddings, primeiramente a criação do *corpus* que é utilizado como conjunto de treinamento dos modelos o pré-processamento necessário para que o *coprus* não contenham informações que interfiram nos *embeddings* das palavras o treinamento e a avaliações dos *embeddings* gerados.

Etapa 1: foi desenvolvido um *script* de *download* de conteúdo textual tendo como base de pesquisa a Wikipédia, ou seja, em cada página visitada da Wikipédia o seu conteúdo textual era armazenado para compor o *corpus* e os links encontrados nessas páginas eram guardados para que em futuras pesquisas os textos desses *links* também fossem utilizadas para montar o *corpus*. O download do conteúdo textual foi realizado e o *corpus* foi montado, com tamanho de 1GB;

Etapa 2: foi feita a limpeza no *corpus* retirando informações desnecessárias, exemplo *tags*, *links* que direcionam para conteúdos dentro da página, por meio da criação de um *script*. A remoção de stop-words (palavras que não agregam valor semântico quando associados com outras palavras, ex: artigos e preposições) também foram retiradas do *corpus* pelo script e palavras com frequência inferior a 5 não foram utilizadas no treinamento dos modelos;

Etapa 3: Foi desenvolvido o algoritmo de treinamento dos modelos Cbow e Skip-gram

utilizando a biblioteca do Python Gensim, já o *script* utilizado para o treinamento do Glove é econtrado https://github.com/maciejkula/glove-python. As avaliações foram realizadas no modelos após o seu treinamento e geração de *embeddings* sendo o *script* para avaliações desenvolvido por Rodrigues et al. (2015) e encontrado em https://github.com/nlx-group/lx-dsemvectors;

Etapa 4: Foram realizadas as avaliações sintáticas e semânticas nos *embeddings* gerados pelo treinamento utilizando o *script* de Rodrigues et al. (2015), as avaliações de semelhança que apresentam os *embeddings* gerados com maior grau de similaridade foram realizadas por meio de funções das bibliotecas Gensim e Glove. Após isso foram realizadas as tabelas e gráficos com as informações de desempenho dos modelos, com os parâmetros definidos e utilizados no treinamento. Na Seção 5 é apresentado o comparativo das avaliações.

4.2 TECNOLOGIAS E FERRAMENTAS

- Python[1]: escolhido devida a popularidade no meio acadêmico, principalmente em áreas como Processamento de Linguagem Natural. Python oferece diversas bibliotecas para se trabalhar com PLN, como: NLTK, GenSim entre outros.
- Gensim[2]: biblioteca com modelos Cbow e Skip-gram implementados
- NLTK[3]: biblioteca com funções Tokenizar.
- Glove[4]: biblioteca com o modelo Glove implementado.

4.3 EXPERIMENTOS

Nesta Seção serão descritos cinco tipos de experimentos que foram realizados nos modelos de Word Embeddings treinados, com o objetivo de identificar parâmetros que pudessem levar a geração de Word Embeddings com maior precisão. Assim como no trabalho

[1] http://python.org.br/
[2] https://radimrehurek.com/gensim/tutorial.html
[3] http://www.nltk.org/
[4] https://pypi.python.org/pypi/glove/1.0.0

de Rodrigues et al. (2015) os resultados das avaliações serão apresentadas de duas formas: a primeira é em relação a utilização do parâmetro de restrição, onde as analogias contidas no conjunto de treinamento devem ter a frequência das quatro palavras que formam a analogia, dentro das 30000 mais frequentes palavras e na segunda não será aplicada a restrição. Com exceção do modelo Glove que não possui um meio para restringir a frequência de palavras que fazem parte das analogias.

- Primeiro teste: Os parâmetros dos modelos Cbow e Skip-gram estão com os valores padrões definido na implementação dos modelos na biblioteca Gensim, com exceção do tamanho da dimensão que no padrão é 100 e no trabalho foi definido com o valor 300, em relação ao tamanho do arquivo. No Glove os parâmetros modificados são a dimensão e o número de épocas de treinamento. Serão apresentadas as precisões dos modelos em relação ao tamanho do conjunto de treinamento;
- Segundo teste: Neste teste o foco será nos modelos preditivos que possuem métodos de otimização computacional apresentados no trabalho, sendo o Amostras Negativas treinado com quantidades diferentes, com o intuito de verificar qual método apresenta melhor resultado. Como o Glove não utiliza tais métodos para execução, ele não irá compor este teste;
- Terceiro teste: este método tem como base a avaliação das épocas de treinamento dos modelos em relação ao tamanho do arquivo ou dimensão, verificando se a quantidade de iterações de treinamento influência no resultado;
- Quarto teste: avaliar a influência da dimensão na precisão dos *embeddings*;
- Quinto teste: teste de similaridade utilizando funções implementadas na ferramenta Word2Vec e Glove, dada uma palavra como entrada retorna os vetores das palavras com maior grau de semelhança.

4.4 PARÂMETROS DOS MODELOS

Nesta Seção serão introduzidos os parâmetros de treinamento do modelos de Word Embeddings, os quais tem grande influência nos resultados finais de desempenho. Os modelos Cbow e Skip-Gram possuem o mesmo método para chamada de treinamento, onde os parâmetros utilizados na avaliação são:

1. *Size*: o tamanho de dimensões utilizados no treinamento;

2. *Window*: define o tamanho da janela de contexto de palavras;
3. Sg: por padrão no modelo é zero, significa que o modelo Cbow será utilizado no para treinamento, caso o valor de Sg seja um, o modelo aplicado será Skip-Gram;
4. Hs: define qual método de otimização computacional será utilizado;
5. *Negative*: é o valor da quantidade de amostras utilizados no método de otimização Amostras Negativas;
6. *iter*: é a quantidade de épocas de treinamento do modelo.

No modelo Glove foram utilizados dois parâmetros para realizar o treinamento sendo eles:

1. Dimensões: representa as quantidades de características utilizadas para induzir os *embeddings*, tamanho da dimensionalidade dos vetores;
2. Épocas: é a quantidade de vezes que o modelo será treinado.

5 EXPERIMENTOS E DISCUSSÕES

Neste capítulo serão descritos os resultados obtidos dos experimentos realizadas onde os valores exibidos nas tabelas são primeiro os experimentos feitos sem restrição e depois com restrição de palavras com exceção do modelo Glove. Para medir a qualidade dos Word Embeddings foi verificada a precisão nas relações sintáticas (antônimos, comparativo, adjetivo para advérbio, plural de verbos e substantivos entre outros) e semânticas (países e capitais comuns, capitais dos estados, relações familiares entre outros) dos vetores gerados.

Foram realizadas oitos experimentos tendo como base os testes definidos no Seção 4.

5.1 AVALIAÇÕES

Na Tabela 1 é apresentado o primeiro experimento (primeiro teste) onde foi realizada a avaliação dos modelos com base no tamanho do conjunto do treinamento. Na grande maioria dos valores apresentados é possível verificar a interferência no resultado final tanto quanto nos resultados sintáticos e semânticos, de que a precisão decai conforme o tamanho do conjunto de treinamento diminui. Os resultados mais satisfatórios são apresentados pelo modelo Cbow sendo o resultado sem e com restrição de palavras (Seção 4.3).

É possível verificar também que na Tabela 1 o modelo Glove apresentou tamanha disparidade em seus resultados comparados com os outros dois modelos Cbow e Skip-gram, o que pode estar diretamente ligado ao tamanho do conjunto de treinamento e quantidade de épocas de treinamento, pois em Pennington et al. (2014) é observado que a menor quantidade de *tokens* de um *corpus* possui tamanho de 1.5B de *tokens* e épocas maiores que 25, sendo que no *corpus* utilizado neste trabalho tem 5M de *tokens* e época 5.

No segundo experimento (Tabela 2) são avaliados a influência dos métodos de otimizações no resultado conforme especificado no segundo teste na Seção 4.3. Foi observado

Tabela 1 – Experimento 1 (teste 1): Precisão (sem e com restrição de palavras) dos modelos separadas por testes sintáticos, semânticos e em seu total

Modelos	Size	Sintática	Semântica	Total
Cbow	1GB	21.7%, 32.2%	17.2%, 31%	20.4%, 32.1%
Skip-Gram	1GB	6.4%, 19.9%	24.5%, 49.7%	11.9%, 25.1%
Glove	1GB	0,043%	0,038%	0,042%
Cbow	500MB	18.4%, 28.3%	11.8%, 26.9%	16.5%, 28.1%
Skip-Gram	500MB	7.6%, 18.3%	17.9%, 43.1%	10,7%, 22,8%
Glove	500MB	0,028%	0,018%	0,025%

que o modelo Cbow gerou *embbedings* com maior qualidade em quase todos os testes, seja o resultado sem ou com restrição de palavras. Verifica-se que a mudança na quantidade de amostras com valores de cinco, dez e quinze não influenciaram no desempenho dos modelos.

Tabela 2 – Experimento 2 (teste 2): Precisão (sem e com restrição de palavras) dos modelos em relação ao método de otimização Amostras Negativas

Modelos	Quantidade de amostras	Precisão
Cbow	5	20.4%, 32.1%
Skip-Gram	5	11.9%, 25.1%
Cbow	10	19.9%, 32.5%
Skip-Gram	10	12.2%, 25.5%
Cbow	15	20.2%, 32.3%
Skip-Gram	15	12.8%, 24.3%

Na Tabela 3 é apresentado o resultado das precisões do experimento 2 (Tabela 2) separadamente em desempenhos sintáticos e semânticos. Os valores da Tabela 3 revelam que independente da mudança nos valores das amostras assim como na Tabela 2 a quantidade de amostras não tem grande influência nas precisões, ou seja, os resultados apresentados não sofrem grande variações com relação as amostras. É observado também que o Cbow tem melhor desempenho nos testes sintáticos, enquanto o Skip-Gram nos testes semânticos.

O experimento na Tabela 4 foi realizado utilizando o teste 2 como base (definido na Seção 4.3) o qual tem como finalidade verificar a influência do método de otimização na precisão dos modelos. Sendo na Tabela 4 apresentadas as precisões sintáticas e semânticas dos modelos em relação ao método Softmax Hiérarquico, é observado que em comparação com os resultados obtidos pelos experimentos com o método Amostras Negativas o modelo Cbow apresenta valores superiores em testes sintáticos, já o Skip-gram é superior em testes semânticos e a utilização de restrição de palavras sempre apresenta desempenho com valor maior. Não foi possível identificar qual o melhor método de otimização pois os resultados de precisões obtidos

Tabela 3 – Experimento 3 (teste 2): Precisão (sem e com restrição de palavras) dos modelos em relação a métodos de otimização Amostras Negativas

Modelos	Quantidade de amostras	Sintática	Semântica
Cbow	5	21.7%, 32.2%	17.2%, 31.0%
Skip-Gram	5	6.4%, 19.9%	24.5%, 49.7%
Cbow	10	21.1%, 32.5%	16.8%, 32.0%
Skip-Gram	10	6.8%, 20.8%	24.5%, 46.7%
Cbow	15	21.2%, 31.8%	17.9%, 34.0%
Skip-Gram	15	6.7%, 18.7%	26.7%, 49.4%

apresentaram valores semelhantes.

No experimento 5 (Tabela 5) e demonstrada a relação dos parâmetros de dimensão com o de épocas de treinamento, observa-se que neste teste o Cbow também apresenta os melhores resultados. A quantidade de épocas treinadas não apresentou grandes mudanças na precisão dos *embeddings* em uma dimensão de tamanho 100.

Na Tabela 6 é apresentada a precisão do modelo Glove entre tamanho de dimensões por época. É constatado que o tamanho da dimensão influência no resultado final é observado também que o treinamento do modelo Glove deve ser realizado em um *corpus* com maior quantidade de textos, pois é por meio das informações contidas no *corpus* que seus *embeddings* são gerados.

Na Tabela 7 os resultados obtidos apresentam que o tamanho do conjunto de treinamento tem maior influência nas precisões do que o tamanho da dimensão é que as épocas de treinamento também influenciam nos desempenhos dos modelos.

Para a verificação sobre o impacto na avaliação dos modelos conforme as mudanças de parâmetros, alguns dos testes realizados serão representados graficamente, facilitando a visualização dos resultados.

Na Figura 9 é possível observar claramente o desempenho superior do modelo Cbow, em ambos tamanhos de conjunto de treinamento, os valores utilizados no gráfico foram dos resultados os modelos treinados com restrição de frequência.

A Figura 10 reflete o desempenho dos modelos sintaticamente e semanticamente, onde

Tabela 4 – Experimento 4 (teste 2): Precisão (sem e com restrição de palavras) dos modelos em relação ao método de otimização Softmax Hierárquico

Modelos	Sintática	Semântica	Total
Cbow	13.5%, 21.9%	8.6%, 17.6%	12.1%, 21.2%
Skip-Gram	6.2%, 19.29%	25.5%, 50.3%	12.1%, 24.9%

Tabela 5 – Experimento 5 (teste 3): Precisão (sem e com restrição de palavras) dos modelos em relação a épocas de treinamento

Modelos	Dim	Época=5	Época=10
Cbow	100	16.4%, 29.2%	16.6%, 28.2%
Skip-Gram	100	14.2%, 28.1%	15.1%, 28.2%
Glove	100	0.039%	0.054%

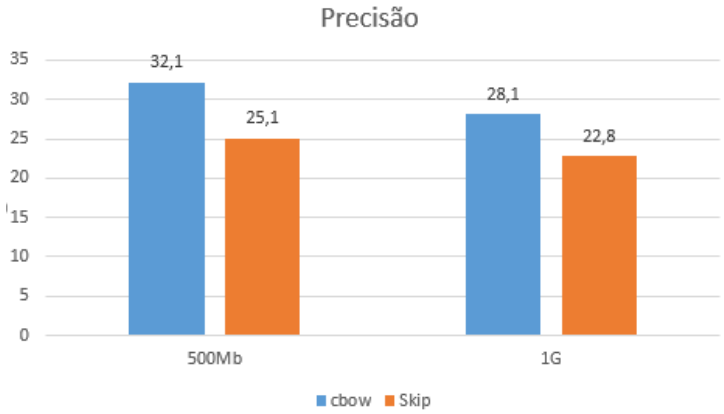

Figura 9 – Representação da relação de precisão em porcentagem por tamanho de conjunto de treinamento dos modelos Cbow e Skip-Gram

Fonte: Autoria própria

o Cbow gerou melhores resultados quando relacionado a semântica das palavras, enquanto o Skip-Gram é um modelo que apresentou melhor desempenho em tarefas sintáticas.

Na Figura 11 os modelos foram treinados com o algoritmo Amostras Negativas é exemplificado a variação da precisão dos modelos Cbow e Skip-Gram em relação a quantidade de amostras. Observando é possível verificar que a quantidade de amostras utilizadas para treinamento não tem grande influência no resultado final.

Na Figura 12 é demonstrada a influência da quantidade de épocas na precisão do modelo Glove, onde conforme maior a época de treinamento maior o resultado apresentado.

Tabela 6 – Experimento 6 (teste 4): Precisão do modelo Glove em relação a épocas de treinamento

Modelos	Dim	Época=5	Precisão
Glove	100	5	0,039%
Glove	300	5	0,042%

5.2 TESTES DE SIMILARIDADE

Nesta seção serão apresentados os *embeddings* aprendidos pelos modelos por meio de funções que apresentaram os vetores com maior semelhança ou os vetores opostos dependendo da função utilizada e dos dados inseridos. Estes testes apresentam os *embeddings* aprendidos durante o treinamento e como eles é possível identificar se o modelo está aprendendo corretamente as correlações entre as palavras. Conforme maior for os valores de desempenho dos modelos significa que melhores *embeddings* estão sendo gerados (*embeddings* que codificam corretamente as relações de similaridade das palavras) e dessa forma mais eficientes serão as aplicações que utilizarem a técnica de Word Embedding.

As similaridades das palavras podem ser calculadas por meio dos métodos implementados nas bibliotecas Gensim e Glove, na qual os modelos aprendem dimensões diferentes de similaridade, como exemplo utilizaremos a equação: Mulher + Rei - Homem = x, que quando inserida no algoritmo tem como resultado Tabela 8.

Na Tabela 8 foi apresentado alguns dos vetores gerados pela equação Mulher + Rei - Homem = x. Na qual o vetor esperado como resultado seria da palavra rainha. Foi obtido deste teste um resultado bom onde a maioria dos *embeddings* retornados possuem relação de maior ou menor grau com a palavra de saída esperada isso significa que o modelo está aprendendo bem pois os resultados apresentam relações de similaridade.

Nesta Tabela 9 foi utilizado um método disponível para os modelos Cbow e Skip-gram onde se identifica palavras opostas, exemplo se inserirmos as palavras positivas bom e triste e negativa mau (Bom + Triste - Mau = x), espera-se como resultado a palavra feliz. Os resultados são os seguintes vetores (Tabela 9).

Neste experimento foi realizada uma avaliação específica de teste sintático onde busca-se verificar se o modelo aprendeu relações de oposição (antônimo) das palavras. É possível verificar que o modelo aprendeu tais relações sendo o primeiro *embedding* retornado a palavra feliz o que era esperado e os outros valores retornados apresentaram certo grau de similaridade com a palavra feliz.

Tabela 7 – Experimento 7 (teste 3): Precisão do modelo Glove em relação ao tamanho de conjunto de treinamento e o tamanho da dimensão

Modelos	Size	Dimensão	Épocas = 5	Épocas = 10	Épocas = 15
Glove	1GB	100	0,039%	0,054%	0,057%
Glove	500MB	300	0,025%	0,031%	0,038%

Tabela 8 – Vetores resultantes da entrada Mulher + Rei - Homem = x, apresentando a palavra e seu vetor *embedding*. Modelo utilizado Cbow com dimensão 300

Termo	Similaridade
esposa	0.5177568197250366
rainha	0.509258508682251
filha	0.47115662693977356
isabel	0.45340287685394287
matilde	0.45105552673339844
sobrinha	0.4383015036582947
eudoxia	0.4362089931964874
arquiduquesa	0.4351155161857605
princesa	0.4346425235271454

Tabela 9 – Vetores resultantes da entrada Bom + Triste - Mau = x, Modelo utilizado Cbow com dimensão 300

Termo	Similaridade
feliz	0.4139024019241333
perplexo	0.39704298973083496
emocionante	0.38122716546058655
emocionado	0.3701743483543396
bonita	0.367981493473053
contente	0.3637571632862091
comovente	0.35777604579925537
lindo	0.3491721451282501
quieto	0.3462328314781189
lamento	0.34402188658714294

Nesta Tabela 10 é apresentado o mesmo teste realizado no Cbow. O Resultado foi mais preciso no método Skip-Gram, apresentado a palavra esperada por primeiro. Foi observado nos *embeddings* gerados pelo modelo Skip-gram que algumas informações estavam contendo dados desnecessários que interferem no aprendizado do modelo, dessa forma realçando a importância da etapa de pré-processamento do conjunto de treinamento que influência nas precisões dos *embeddings*.

Neste segundo teste do Skip-gram (Tabela 11), os resultados apresentados não foram os esperados. Falhando em relacionar palavras opostas e também apresentou *embeddings* com informações que não contribuem para o aprendizado do modelo.

Esses, são dois testes implementados pelo Word2vec, sendo assim, utilizados pelos métodos Cbow e Skip-Gram nos testes de similaridade. O modelo Glove possui um teste que verifica os vetores mais similares dado uma palavra com entrada, como exemplo a palavra rainha, rei, homem e mulher utilizadas.

Tabela 10 – Vetores resultantes da entrada Mulher + Rei - Homem = x, Modelo utilizado SKip-Gram com dimensão 300

Termo	Similaridade
rainha	0.522188663482666
pylas	0.4944584369659424
saxe-coburgo-saalfeld	0.48616689443588257
ensour	0.47987279295921326
adosinda	0.47823023796081543
odiava-o	0.47748303413391113
etelfleda	0.4774409830570221
brunilda	0.47381290793418884
clodosvinda	0.46935635805130005
montferrat	0.46747395396232605

Tabela 11 – Vetores resultantes da entrada Bom + Triste - Mau = x, Modelo utilizado Skip-Gram com dimensão 300

Termo	Similaridade
emocionado	0.41842854022979736
barbini	0.3991577923297882
emocionante	0.38122716546058655
seguiremos	0.39764365553855896
brioso	0.3922393321990967
entristecido	0.3829878568649292
dor.cumprimentos	0.3803950548171997
convida-la	0.3797486424446106
cadastrou	0.3791902959346771
ferrando	0.37809279561042786

A Tabela 12 apresenta os vetores gerados dada a palavra rainha como entrada, seus *embeddings* resultantes não apresentam um grau direto de semelhança e contém palavras fora do contexto.

Na Tabela 13 foi utilizada a palavra rei, e dos resultados obtidos nenhumas das palavras foi correta e também não apresentaram relacionamento contextual um com os outros e assim como o teste de similaridade realizado no modelo Skip-gram os *embeddings* retornados deste teste continham dados desnecessários que não contribuíram para o aprendizado e interferiram na geração dos *embeddings*.

Nesta Tabela 14 assim como nos testes de similaridade anteriores do modelo Glove os resultados não foram satisfatórios onde as palavras retornadas não condizem em nenhum grau de relacionamento contextual com a palavra inserida homem.

Os vetores resultantes apresentados na Tabela 15 dada a palavra mulher como entrada

deveriam ser semelhantes a palavra em questão em vez disso retornaram *embeddings* sem relação com o contexto.

5.3 RESULTADOS

Após a execução dos treinamentos e avaliações dos modelos foi possível verificar os desempenhos dos modelos. Observando os resultados dos treinamentos dos modelos, o Cbow foi o que apresentou melhor resultado. O aumento do tamanho do conjunto de treinamento de 500Mb para 1G refletiu no resultado final, tendo os valores das precisões aumentados quando realizada a mudança de valores, sendo este um parâmetro que influência no resultado final. A utilização do método Amostras Negativas nos treinamentos não apresentou grande aumento dos valores de desempenho, em relação aos treinamentos com o método Softmax Hierárquico, dessa forma não sendo possível identificar o melhor método em razão dos resultados apresentaram valores semelhantes de precisão.

Foi possível verificar com base em todos os testes realizados do Skip-gram e Cbow qual modelo apresentou melhor desempenho em relação ao tipo de teste (sintático e semântico), onde na maioria dos casos, nos testes sintáticos o melhor modelo foi o Cbow, é nos testes semânticos o modelo Skip-Gram possui resultado melhor.

Os parâmetros dimensão de 100 e 300 e épocas de tamanho e de cinco e dez, quando utilizados como parâmetros avaliadores do treinamento do modelo Glove, não apresentaram grandes mudanças no resultado final da avaliação.

Nos gráficos apresentados nas Figuras 9, 10, 11 e 12 é demonstrado as relações entre as mudanças geradas conforme os valores dos parâmetros utilizados, ou seja, é apresentada a relação precisão por parâmetros.

Com base nos parâmetros alterados, foi observado que os mesmos quando modificados

Tabela 12 – Vetores com maior grau de semelhança com a palavra rainha

Termo	Similaridade
cristina1	0.88293658862381175
isabel	0.86844569063079358
portugalbisnetosjaime	0.84038233680154162
dinamarcainfanta	0.83783755129781001

Tabela 13 – Vetores com maior grau de semelhança com a palavra rei

Termo	Similaridade
ensour	0.89929617968307951
srongbrtsansgampo	0.88120081583034615
portugal33	0.88021087324680247),
lusignano	0.86142520252131005

influenciaram no resultado de desempenho apresentado, por mais que esses valores fossem pequenos. Sendo assim é constatado que a mudança na precisão dos *embeddings* existe e para a possibilidade de se verificar valores maiores no desempenho dos modelos, definições paramétricas com valores superiores ao deste trabalho devem ser utilizadas no processo de treinamento.

Como observado na execução do modelo Glove, onde os valores de parâmetros utilizados como tamanho do conjunto de treinamento, sendo este o parâmetro que mais influenciou no resultado inferior do modelo, dimensões e épocas de treinamento, interferem no resultado final.

Foi verificado no trabalho que para se obter resultados significativos das avaliações dos testes do Glove, seria necessário um *corpus* com tamanho de *tokens* de 1.5B e épocas de treinamento superiores ao valor 25 para se obter desempenhos significativos, como pode ser verificado em Pennington et al. (2014), os parâmetros definidos para utilização neste trabalho não foram favoráveis ao desempenho do modelo Glove.

Os testes de similaridade realizados nos modelos Cbow e Skip-Gram apresentaram resultados satisfatórios, no primeiro e no segundo teste, independente de o modelo Cbow ter sido mais preciso nos resultados, e o Skip-Gram ter apresentado alguns vetores não relacionados ao objetivo da função, os resultados foram bons.

Para avaliar a similaridade do modelo Glove foi utilizado um método que dada uma palavra com entrada retorna as palavras com maior grau de similaridade. Foram realizados quatro teste onde os resultados não foram satisfatórios sendo o primeiro retornando apenas um vetor com grau de semelhança com a palavra inserida, no segundo nenhum dos vetores

Tabela 14 – Vetores com maior grau de semelhança com a palavra homem

Termo	Similaridade
uiri	0.90938833270671537
multiversatil	0.86524604290180851
tulefat	0.83496890900634135
espacializado	0.8308406583894058

apresentou relação clara com a palavra rei. Nos dois últimos testes os vetores retornados não apresentaram relação de similaridade com a palavra inserida. Foi verificado que os vetores retornados nos testes estavam contendo informações indevidas, ou seja, no treinamento do Glove os *embeddings* estavam sendo gerados com informações sem significado afetando diretamente na precisão do teste. Os relacionamentos de semelhanças dos vetores podem ser verificados nas Tabela 12, Tabela 13, Tabela 14 e Tabela 15.

Durante os treinamentos dos modelos algumas configurações foram observadas. Para realizar o treinamento do modelos preditivos Cbow e Skip-gram em alguns dos testes, houve a necessidade de a memória RAM do computador ser maior que 4GB. A utilização de uma máquina com 4GB executava alguns dos testes mas seu tempo de execução é demorado.

O treinamento do modelo Glove apresentou também problemas com memória de 4GB. E para sua execução é necessário utilizar Python 64bits, ele não executa no Python 32bits. Utilizando uma memória de 12GB o modelo mais simples treinado levou em torno de 6 horas para ser treinado.

O processo de avaliação não é tão demorado quanto ao de treinamento, a execução é em torno de minutos. Os modelos gerados, que no treinamento precisavam de mais memória também precisam de memória superior a 4GB para serem avaliados.

Tabela 15 – Vetores com maior grau de semelhança com a palavra mulher

Termo	Similaridade
nulipara	0.8578752658588602
puerpera	0.85463855383119003
rhnegativa	0.8312910729145282
primigesta	0.81504698456283653

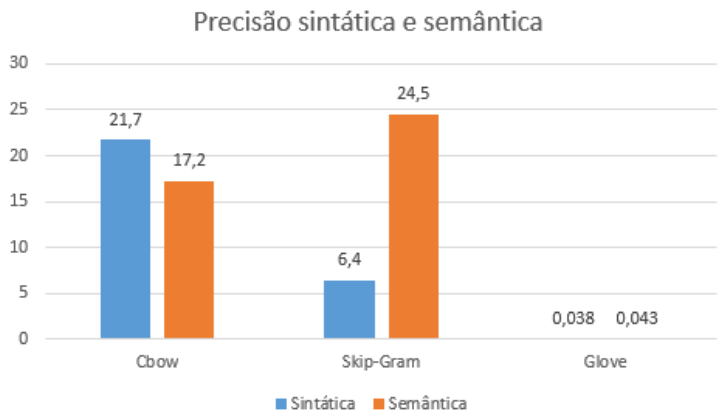

Figura 10 – Representação da relação de precisão em porcentagem por testes semânticos e Sintáticos dos modelos

Fonte: Autoria própria

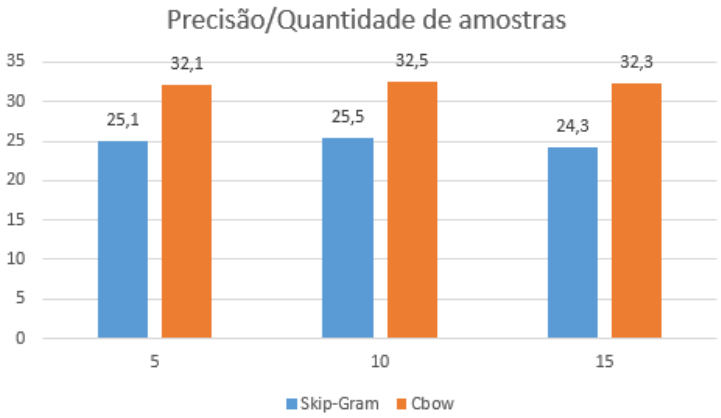

Figura 11 – Representação da relação de precisão em porcentagem por tamanho arquivo do modelos Cbow e Skip-Gram

Fonte: Autoria própria

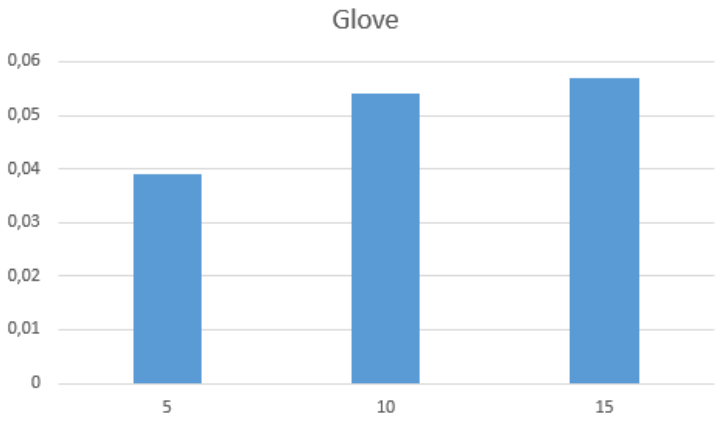

Figura 12 – Relação de precisão em porcentagem do método Glove pelo quantidade de épocas utilizadas, modelos treinados com dimensão 100

Fonte: Autoria própria

6 CONCLUSÕES

Neste trabalho foi descrito todo o processo para a criação, execução e avaliação de Word Embeddings. Foi apresentado o potencial dos Word Embeddings o qual pode aprender o significado das palavras. Foi verificado que os modelos preditivos apresentam resultados bons, mesmo com tamanho razoavelmente pequeno de conjunto de treinamento e tamanho de dimensão menor do que os definidos por Mikolov et al. (2013a) que apresentavam bons resultados se comparados com os de Pennington et al. (2014), sendo esses parâmetros uns dos mais utilizados para avaliar a precisão de *embeddings*. Já do modelo de contagem básica não se pode ter uma conclusão sobre seu desempenho, devido a quantidade de conjunto do treinamento, que acabou sendo inferior ao necessário para que este modelo produzisse resultados que pudessem ser comparados com os outros modelos Cbow e SKip-gram.

Nas tarefas envolvendo a similaridade dos vetores, por meio dos testes realizados, foi identificado que o conjunto de treinamento continha elementos desnecessários e prejudiciais para o aprendizado dos modelos. Pois na execução dos testes alguns dos vetores resultantes eram formados por conjuntos de letras com números ou com algumas pontuações, algo que não deveria ocorrer, sendo na etapa de pré-processamento o momento para se trabalhar no *corpus* tendo como finalidade a produção de um conjunto de treinamento, realizando dessa forma ações no *corpus* para deixa-lo em condições para treinamento. Os resultados obtidos do estudo podem ser aproveitados no desenvolvimento de trabalhos futuros.

Em relação aos resultados dos testes de similaridade, os mesmos apresentaram desempenho mediano, pois em alguns dos testes os valores eram os esperados, já em outros os valores retornados não correspondiam com o desejado, senda a ocorrência desses testes desenvolvidos para visualizar os valores de precisões tabelados.

Nos experimentos e testes realizados no trabalho o modelo que apresentou maior precisão no aprendizado dos *embeddings* e nos testes de similaridade foi o Cbow, e observando as tabelas com os resultados dos testes, é verificado que o parâmetro de restrição das frequências das palavras, independente de outros parâmetros ou método de otimização computacional, quando utilizado aumenta significativamente a precisão (função utilizada para quantificar a qualidade dos *embeddings*).

O resultados obtidos junto ao trabalho demonstram a importância da sub-área de Representação Distribuída, com a captação de conhecimento sintático e semântico, seja por redes neurais ou matriz de co-ocorrência, para ser utilizadas em aplicações do dia-a-dia.

6.1 TRABALHOS FUTUROS

Como possíveis trabalhos futuros, podemos apontar:

- Realizar a avaliação dos modelos apresentados separadamente, dando ênfase no parâmetros de avaliação.
- Fazer um trabalho das equações necessárias no aprendizado de Word Embeddings, em todo o seu processo.
- Realizar um estudo do modelo Glove levando em conta as dificuldades computacionais do modelo.
- Desenvolver uma ferramenta para realizar o pré-processamento de *corpus*.
- Aplicar os *embeddings* gerados pelos modelos em ferramentas.

REFERÊNCIAS

ANNES, R. Aplicações de sistemas multiagentes e aprendizagem automática no processamento da linguagem natural. SD.

BARONI, M.; DINU, G.; KRUSZEWSKI. Dont' count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. 2014.

BENGIO, Y. et al. A neural probabilistic language model. 2003.

CARDON, A.; MULLER, D. N. Introdução as redes neurais artificiais. 1994.

CARVALHO, A. P. d. L. F. d. type, **Redes Neurais Artificiais**. SD. Disponível na Internet. `http://conteudo.icmc.usp.br/pessoas/andre/body.htm`. Acesso em 23/11/2016.

CHAUBARD, F.; MUNDRA, R.; SOCHER, R. **CS 224D:Deep Learning for NLP**. 2015. Disponível na Internet. `http://cs224d.stanford.edu/lecture_notes/notes1.pdf`. Acesso em 16/08/2016.

CHOPRA, A.; PRASHAR, A.; SAIN, C. **Natural Language Processing**. 2013. Disponível na Internet. `http://www.ijteee.org/final-print/nov2013/Natural-Language-Processing.pdf`. Acesso em 27/07/2016.

COLLOBERT, R.; WESTON, J. **A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning**. [S.l.]: NEC Labs America, 2008.

DHILLON, P. S.; FOSTER, D. P.; UNGAR, L. H. Eigenwords: Spectral word embeddings. 2015.

FINOCCHIO, P. M. M. A. F. Noções de redes neurais artificiais. 2014.

FIRTH, J. R. A synopsis of linguistic theory 1930–1955. 1957.

GOLDBERG, Y. A primer on neural network models for nature language processing. 2015.

GOLDBERG, Y.; LEVY, O. Word2vec explained: Deriving mikolov et al.'s negative-sampling word-embedding method. 2014.

GUREVYCH, I.; MUHLHAUSER, M. **Natural Language Processing for Ambient Intelligence**. 10. ed. [S.l.]: Axcel Books, SD.

HARP, A.; DAVIS, A.; AGARWAL, A. **TensorFlow**. 2016. Disponível na Internet. `https://www.tensorflow.org/versions/r0.11/tutorials/word2vec/index.html`. Acesso em 11/10/2016.

JURAFSKY, D.; MARTIN, J. **Speech and Language Processing**. [S.l.: s.n.], 2015.

KRENKER, A.; BESTER, J.; KOS, A. Introduction to the artificial neural networks. SD.

LEVY, O.; GOLDBERG, Y. Neural word embedding as implicit matrix factorization. 2014.

LIDDY, E. D. Natural language processing. 2001.

LOPES, E. D. Utilização do modelo skip-gram para representação distribuída de palavras no projeto media cloud brasil. 2015.

MIKOLOV, T. et al. Efficient estimation of word represesntations in vector space. 2013.

MIKOLOV, T.; LE, Q. V.; SUTSKEVER, I. Exploiting similarities among languages fo machine translation. 2013.

MIKOLOV, T. et al. Distributed representations of words and phrases and their compositionality. 2013.

MORIN, F.; BENGIO, Y. Hierarchical probabilistic neural network language model. 2005.

NADKAMI, P. M.; OHNO-MACHADO, L.; CHAPMAN, W. W. **Natural language processing: an introduction**. 2011. Disponível na Internet. `http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3168328/`. Acesso em 27/07/2016.

NUNES, M. d. G. V. **O processamento de Línguas Naturais: para quê e para quem?** [S.l.: s.n.], 2008.

OLAH, C. **Deep Learning, NLP, and Representations**. 2014. Disponível na Internet. `http://colah.github.io/posts/2014-07-NLP-RNNs-Representations/`. Acesso em 24/11/2016.

PARDO, T. A. S.; NUNES, M. d. G. V. Análise de discurso: Teorias discursivas e aplicações em processamento de línguas naturais. 2003.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. Glove: Global vectors for word representation. 2014.

RODRIGUES, J. et al. Lx-dsem vectors: Distributional semantics models for portuguese. 2015.

RONG, X. **word2vec Parameter Learning Explained**. [S.l.: s.n.], 2015.

RUDER, S. **On word embeddings - Part 1**. 2016. Disponível na Internet. `http://sebastianruder.com/word-embeddings-1/`. Acesso em 22/11/2016.

SOUTNER, D.; MULLER, L. Natural language processing. 2014.

TISSOT, H. C.; CAMARGO, L. C.; POZO, A. T. R. Treinamento de redes neurais feedforward: comparativo dos algoritmos backpropagation e differential evolution. SD.

WAHLSTER, W. **The Role of Natural Language in Advanced Knowledge-Based Systems**. 4. ed. [S.l.]: Campus, SD.

ZUBEN, F. J. V. **Introdução à Inteligência Artificial**. 1. ed. [S.l.]: Unicamp, SD.