ROBERTA CAROLINE RODRIGUES SILVA

ANOTAÇÃO SEMÂNTICA AUTOMÁTICA POR MEIO DE REDES NEURAIS PROFUNDAS PARA CORPORA NA LÍNGUA INGLESA

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae.*

Orientador: Alcione de Paiva Oliveira

Coorientadora: Alexandra Moreira

VIÇOSA – MINAS GERAIS

2019

Ficha catalográfica preparada pela Biblioteca Central da Universidade Federal de Viçosa - Câmpus Viçosa

T
S586a
2019

Silva, Roberta Caroline Rodrigues, 1993-
Anotação semântica automática por meio de redes neurais profundas para corpora na língua inglesa / Roberta Caroline Rodrigues Silva. – Viçosa, MG, 2019.
66 f. : il. (algumas color.) ; 29 cm.

Orientador: Alcione de Paiva Oliveira.
Dissertação (mestrado) - Universidade Federal de Viçosa.
Referências bibliográficas: f. 60-66.

1. Processamento de linguagem natural (Computação). 2. Língua inglesa - Semântica. 3. Redes neurais (Computação). 4. Memória de longo prazo. 5. Ontologia. I. Universidade Federal de Viçosa. Departamento de Informática. Programa de Pós-Graduação em Ciência da Computação. II. Título.

CDD 22. ed. 005.1

ROBERTA CAROLINE RODRIGUES SILVA

ANOTAÇÃO SEMÂNTICA AUTOMÁTICA POR MEIO DE REDES NEURAIS PROFUNDAS PARA CORPORA NA LÍNGUA INGLESA

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae.*

APROVADA: 28 de novembro de 2019.

Assentimento:

Roberta Caroline Rodrigues Silva
Autora

Alcione de Paiva Oliveira
Orientador

Dedico esse trabalho, com muito amor e gratidão, aos meus pais, que nunca mediram esforços para lutar por minha educação. Amo vocês mais que tudo!

AGRADECIMENTOS

Agradeço primeiramente a Deus pelas oportunidades colocadas em minha trajetória de vida e por me inspirar força e perseverança nos momentos de dificuldade. Aos meus pais pela compreensão, apoio e instrução ao me guiarem no caminho certo. Aos professores Alcione e Alexandra, exemplos de profissionais e seres humanos, por terem emprestado a este trabalho seu saber. Também por terem sido orientadores seguros, atentos e pacientes. Minha eterna gratidão pelas leituras, revisões, conselhos e pelo conhecimento compartilhado que me fizeram crescer para além da sala de aula. Aos demais professores do DPI e a todos os meus professores da graduação, essenciais em minha trajetória acadêmica. Obrigada a todos os colegas de sala e companheiros de laboratório por todos os momentos que compartilhamos. Não posso deixar de citar o grupo de Linguística, os funcionários do CCE e demais servidores da UFV pelo excelente serviço prestado à comunidade. O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001. Muito obrigada!

RESUMO

RODRIGUES SILVA, Roberta Caroline, M.Sc., Universidade Federal de Viçosa, novembro de 2019. **Anotação Semântica Automática por meio de Redes Neurais Profundas para Corpora na Língua Inglesa**. Orientador: Alcione de Paiva Oliveira. Coorientadora: Alexandra Moreira.

A anotação semântica permite que pessoas e dispositivos computacionais entendam mais facilmente o significado de uma sentença expressa em linguagem natural. Classificar textos de acordo com seu conteúdo é frequentemente uma das primeiras etapas realizadas por aplicativos voltados para o processamento de linguagem natural. E, apesar de ser um princípio básico, este passo é feito, geralmente, de forma manual, o que faz com que o processo seja lento, custoso e limitado. Para que a anotação seja realizada automaticamente, os métodos devem ser bem definidos por meio de um conjunto de características ou *features*, elaborado por especialistas, a fim de que o sistema possa atribuir probabilidades e fazer inferências. Nesta dissertação é apresentado um modelo de rede recorrente profunda que anota semanticamente textos escritos em inglês, e manipula como rótulo categorias de uma ontologia de nível topo. Os testes mostraram que é possível obter melhores resultados do que os encontrados em modelos que precisam do fornecimento prévio de *features.*

Palavras-chave: PLN. Anotação Semântica. Rede Neural Recorrente. LSTM. Ontologia.

ABSTRACT

RODRIGUES SILVA, Roberta Caroline, M.Sc., Universidade Federal de Viçosa, November, 2019. **Semantic Labeling of English Texts with Ontological Categories Employing Recurrent Networks**. Adviser: Alcione de Paiva Oliveira. Co-adviser: Alexandra Moreira.

Semantic labeling of texts allows people and computing devices to more easily understand the meaning of a natural language sentence as a whole. Semantic annotation is often one of the first steps carried out by applications focused on natural language processing. However, this step is often done manually, which is very expensive and time-consuming. When automatic methods are employed, they require that a set of features, elaborated by specialists, be provided so that the system can assign probabilities in order to make inferences. In this thesis we present a model of the deep recurrent network that semantically annotates texts in English using as labels the top categories of an ontology. The tests showed that it is possible to obtain better results than the models that need the features to be made explicit.

Keywords: NLP. Semantic Annotation. Recurrent Network. LSTM. Ontology.

LISTA DE ILUSTRAÇÕES

Figura 1 Um esquema da tarefa de marcação 22
Figura 2 Modelo básico de um neurônio artificial 29
Figura 3 Funções de ativação 30
Figura 4 Modelo de uma rede feedforward..... 31
Figura 5 Um perceptron multicamadas totalmente conectado com uma camada oculta 32
Figura 6 Modelo de Rede Recorrente Simples..... 36
Figura 7 Bloco de memória LSTM com uma célula..... 38
Figura 8 Um segmento do *corpus* OANC anotado com a ontologia Schema 43
Figura 9 Um segmento do *corpus* Wikiner após limpeza 44
Figura 10 Uma ilustração da arquitetura de rede neural BiLSTM proposta 47
Figura 11 Curvas de precisão geradas pelo modelo com o *corpus* OANC. 53
Figura 12 As curvas de precisão do modelo com o *corpus* Wikiner..... 56

LISTA DE TABELAS

Tabela 1 OANC em números 43
Tabela 2 Wikiner em números 44
Tabela 3 Hiperparâmetros utilizados no modelo 48
Tabela 4 Matriz de confusão *corpus* OANC 54
Tabela 5 Resultados por classe do Teste 2 55
Tabela 6 Comparação dos resultados com o trabalho de Andrade (2018). 55
Tabela 7 Matriz de confusão do *corpus* Wikiner 57
Tabela 8 Resultados por classe Teste 3 57
Tabela 9 Comparação dos resultados com o trabalho de Mendonça Junior et al. (2016) 57

LISTA DE ABRAVIATURAS E SIGLAS

ADAM	Adaptive Moment Estimation
AM	Aprendizado de Máquina
BiLSTM	Bidirectional Long Short-Term Memory
BIO	Beginning-Inside-Outside
CAPES	Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
CPU	Central Processing Unit
CNN	Convolutional Neural Networks
CRF	Conditional Random Field
DOLCE	Descriptive Ontology for Linguistic and Cognitive Engineering
FNNs	Redes Neurais Feedforward
GPU	Unidade de Processamento Gráfico
GRU	Gated Recurrent Units
LSTM	Long Short-Term Memory
MLPs	Multilayer Perceptrons
NER	Named Entity Recognition
OANC	Open American National Corpus
OLiA	Ontologies of Linguistic Annotation
PoS	Part of Speech
PLN	Processamento de Linguagem Natural
ReLU	Unidade Linear Retificada
RNAs	Redes Neurais Artificiais
RNN	Rede Neural Recorrente
Rprop	Retro Propagação Resiliente
SRL	Semantic Role Labeling
SUMO	Suggested Upper Merged Ontology
TanH	Tangente Hiperbólica

SUMÁRIO

1 INTRODUÇÃO 11
1.1 Objetivos 13
1.2 Organização da dissertação 14
2 FUNDAMENTAÇÃO TEÓRICA 15
2.1 Base Linguística 15
2.1.1 O Item Lexical, Lexema, Semântica e Polissemia 15
2.1.2 Ontologias 18
2.1.5 Anotação Semântica 21
2.2 Aprendizado de Máquina 23
2.2.1 Aprendizado Supervisionado 25
2.2.2 Reconhecimento de Padrões 26
2.3 Aprendizado Profundo 27
2.4 Redes Neurais Artificiais 27
2.4.1 Processos de Aprendizagem 33
2.4.2 Regularização 34
2.4.3 Redes Neurais Recorrentes 35
2.5 Long Short-Term Memory 36
2.5.1 LSTM Bidirectional 38
2.6 Trabalhos Correlatos 39
3 MODELO PROPOSTO 41
3.1 A ontologia Schema.org 41
3.2 Características dos *Corpora* 42
3.2.1 Open American National Corpus (OANC) 42
3.2.2 Wikiner *corpus* 43
3.4 Hiperparâmetros 48
4 RESULTADOS 51
5 CONCLUSÃO 58
REFERÊNCIAS 60

1 INTRODUÇÃO

O volume de informação armazenada em meios digitais pode ser representado por um número tão extenso que fica difícil de ser imaginado e compreendido pelas pessoas. Um estudo recente levantado pela Cisco mostra que a capacidade instalada de armazenamento dos data centers atingirá 1,3 zettabytes em 2021 (CISCO, 2018). Esse crescente volume de informação a ser armazenado e disponibilizado se apresenta, em sua grande parte, na forma de textos em linguagem natural. Esse volume de textos em linguagem natural é impossível de ser tratado e analisado de forma manual. Com efeito, há uma grande necessidade em explorar cada vez mais o potencial dos métodos de aprendizado de máquina (AM) para analisar e extrair o conhecimento existente nessas bases de dados. Uma das subáreas que podem se beneficiar deste grande volume de dados para atingir seus objetivos é o ramo do Processamento da Linguagem Natural (PLN). O Processamento de Linguagem Natural é uma subárea da Inteligência Artificial, cujo objetivo, segundo Jurafsky e Martin (2008) é o de desenvolver dispositivos computacionais capazes de realizar tarefas úteis que envolvam a linguagem natural. Já para Pustejovsky e Stubbs (2012), o principal propósito do PLN é permitir que a máquina alcance certa compreensão humana de textos nos mais diversos idiomas, entenda-os, extraia inferências e produza novas informações de forma precisa, útil e inteligente.

Dentre as tarefas mais fundamentais para desenvolver sistemas capazes de compreender a linguagem humana destaca-se a tarefa da compreensão das unidades lexicais, ou seja, a compreensão da palavra. A compreensão da palavra envolve diferentes aspectos ou facetas. Existem aspectos sintáticos que ajudam na compreensão de uma palavra como, por exemplo, a classe gramatical a qual pertence. A determinação e anotação da classe gramatical de uma palavra é chamada de etiquetagem PoS (*Part of Speech) tagging.* Ela indica se a palavra é um verbo, adjetivo, preposição, substantivo, etc. O número de etiquetas ou *tags* é variável, sendo um dos mais comuns o proposto por Mitchell et al. (1993) para o projeto *Penn Treebank Tagset*, contendo 45 classes de palavras.

A classe sintática de uma palavra também revela parte de seu significado. Por exemplo, um substantivo indica um ente concreto ou abstrato. Um verbo indica um estado ou uma ação. Mas para compreender uma palavra é preciso aprofundar em seus aspectos semânticos. Novamente, a semântica possui diversas facetas. Por

exemplo, a palavra “carro”, dentre outros aspectos, indica um meio de transporte (aspecto funcional), também indica um artefato produzido pelo homem (sua natureza intrínseca ou ontológica). Cada uma destas facetas ajuda na compreensão do significado subjacente ao item lexical.

No entanto, o ser humano percebe essas diversas facetas de forma integrada sendo ele capaz de detectar qual aspecto semântico está sendo ressaltado em um enunciado emitido em determinado contexto. Para que os itens lexicais sejam melhor compreendidos é necessário adicionar informações sintáticas e semânticas nos dispositivos computacionais. Essas informações adicionais são chamadas de “anotações”. Segundo Pustejovsky e Stubbs (2012), qualquer adição de metadados usados para marcar elementos do conjunto de dados é chamada de anotação sobre a entrada. Essas anotações podem diminuir resultados indesejados, além de permitir a produção de algoritmos mais eficientes para suprimir problemas de ambiguidade e inexatidão de contexto. Dentre os diversos tipos de anotações semânticas, a anotação de papéis semânticos é uma das que mais tem se destacado. Segundo Màrquez et al. (2008), a anotação de papéis semânticos (*Semantic Role Labeling* - SRL) é a identificação e anotação por meio de dispositivos computacionais dos argumentos em um texto. Isso tem sido uma tarefa comum em congressos importantes tais como o *Conference on Computational Natural Language Learning* (CARRERAS, MARQUEZ, 2004). Todavia, existe outro aspecto semântico que caracteriza um item lexical que não tem recebido o mesmo nível de atenção que o SRL. Trata-se da anotação de como um objeto é percebido em sua essência, ou seja, o seu aspecto ontológico.

Uma anotação baseada em ontologia é capaz contextualizar melhor um documento e assim otimizar o processo de interpretação semântico de bases textuais. Por exemplo, se a palavra “vendo” for anotada com a classe *Commerce sell* da ontologia FrameNet (RUPPENHOFER et al., 2016) ela pode ser entendida que trata da transferência permanente de um bem. No entanto, se a mesma palavra for anotada com a classe *Perception_active*, oriunda da mesma ontologia, pode inferir que trata da percepção de algo por uma entidade animada. Sendo assim, a tarefa de anotação semântica baseada em ontologias também deve receber atenção dos pesquisadores devido ao seu potencial de contribuição para o entendimento de um texto.

Tradicionalmente, o processo de anotação é caro e demorado, sendo feito manualmente por especialistas. No entanto, avanços recentes nas técnicas de hardware e aprendizado de máquina, especialmente os de aprendizado profundo,

abriram novas perspectivas para a anotação semântica automática. Quando sistemas de etiquetagem automáticos são empregados, métodos como modelos de máxima entropia são usados. Estes métodos necessitam ser treinados por meio de textos previamente anotados por especialistas e que estes especialistas, também, forneçam as regras (*features*) que devem ser usadas para atribuir a probabilidade de um item pertencer a uma determinada classe. Esta etapa de criação das regras ou *features* é chamada de engenharia de *features*, o que, também, torna o desenvolvimento de um sistema de anotação uma tarefa cara. Contudo, os modelos de redes neurais recorrentes de sequência a sequência (SUTSKEVER et al., 2014) surgiram, recentemente, como uma possibilidade de criação de sistemas de anotação que prescindem da etapa de engenharia de *features*, facilitando o desenvolvimento de sistemas de anotação.

O trabalho de pesquisa descrito nesta dissertação parte das hipóteses de que a anotação semântica baseada em ontologia pode contribuir significativamente para a compreensão de documentos textuais por dispositivos computacionais e que a técnica de aprendizado profundo pode eliminar a etapa de engenharia de *features* e produzir sistemas de anotação mais precisos. A ontologia usada nesta pesquisa foi a Schema.org (GUHA et al., 2016). Os dados que serviram de entrada para o modelo foram retirados do OANC (IDE, 2013). Este *corpus* é composto por uma gama de textos, artigos técnicos, cartas, transcrições governamentais, dados de fala oral, bem como telefonemas e conversações. Tanto a ontologia quanto o *corpus* foram escolhidos porque os mesmos foram utilizados na pesquisa de Andrade (2018), a qual serviu como *baseline* para medir a precisão do modelo proposto.

1.1 Objetivos

O objetivo geral desta pesquisa é produzir um anotador de *corpus* textuais tendo como elementos categorias semânticas. Para isso, especificamente, pretende-se:

Anotar semanticamente, baseando-se em uma ontologia, de forma automática textos do Inglês americano com aplicação de aprendizagem supervisionada por meio de uma rede neural profunda.

Comparar os resultados das técnicas de aprendizado profundo, neste trabalho utilizadas, com os de técnicas mais custosas desenvolvidas previamente com o mesmo *corpus*.

1.2 Organização da dissertação

Esta dissertação é composta por cinco capítulos. Os temas foram abordados de acordo com a pertinência para esta discussão e proposta.

O presente Capítulo, Capítulo 1, faz uma introdução ao tema. O Capítulo 2 apresenta a fundamentação teórica entorno do Processamento de Linguagem Natural e mostra algumas técnicas que são empregadas no processamento de texto, além de descrever trabalhos relacionados com esta dissertação. O Capítulo 3 expõe a metodologia empregada detalhando as técnicas de aprendizado de máquina, a arquitetura utilizada na construção do modelo, a ontologia e alguns hiperparâmetros. O Capítulo 4 apresenta os resultados obtidos com a implementação do modelo proposto. Por fim, o Capítulo 5 mostra as conclusões sobre o trabalho realizado.

2 FUNDAMENTAÇÃO TEÓRICA

Nesse capítulo é apresentada uma revisão bibliográfica com a finalidade de familiarizar o leitor com os conceitos abordados neste trabalho, tais como linguística e anotação de *corpus*, aprendizado de máquina, aprendizado profundo, ontologias e anotação semântica.

2.1 Base Linguística

Uma vez que o PLN tem como objeto de estudo a linguagem natural oral e escrita, visando o desenvolvimento de aplicações que sejam capazes de compreender o significado subjacente a enunciados em língua natural, é de fundamental importância conhecer os fundamentos básicos desta forma de comunicação. Os tópicos a seguir descrevem alguns conceitos da linguística que estão relacionados com a presente pesquisa.

2.1.1 O Item Lexical, Lexema, Semântica e Polissemia

A compreensão da linguagem natural envolve diversas etapas e desafios. Apesar da linguagem natural, segundo a visão da linguística cognitiva, seguir o princípio geral da psicologia Gestalt (KOFFKA, 2013), onde o "todo é maior que a soma de suas partes", analisar um enunciado pelas suas partes colabora com o processo de compreensão de um texto. Em uma análise simplificada, textos são compostos por sentenças, que são compostas por palavras, que por sua vez são compostas por morfemas, sendo estas as menores unidades semânticas (MATTHEWS, 2014). Segundo Lewis et al. (1997), a visão tradicional divide a linguagem em gramática (estrutura) e em vocabulário (palavras), porém a abordagem lexical, que é adotada neste trabalho, propõe que a linguagem é formada por blocos (itens lexicais) que quando combinados produzem um texto contínuo coerente (LEWIS et al., 1997). Neste sentido, itens lexicais podem variar desde morfemas até sequências de várias palavras, como expressões idiomáticas e construções[1] (GOLDBERG, 2003).

[1] Construções são pares forma-significado que não estão restritas pelos níveis de divisões estruturais da gramática tradicional.

Apesar dos morfemas serem as menores unidades com significado, o foco desta pesquisa são as palavras ou lexemas. Um lexema é um conjunto constituído pela forma canônica de uma palavra ou lemma, mas as flexões da palavra comumente abrigadas na mesma entrada de dicionário. Parte-se do princípio que a função dos morfemas (prefixo, sufixo, radical) está bem estabelecida, enquanto que as palavras possuem um maior grau de ambiguidade. Em relação a análise automática das palavras, a determinação de suas características sintáticas e de sua classe gramatical (*PoS ou Part of Speech*) é uma tarefa que é considerada como solucionada (JURAFSKY, 2000). No entanto, estabelecer o significado de um item lexical é ainda uma tarefa desafiadora devido ao seu caráter polissêmico (NORVIG, LAKOFF, 1987). Por exemplo, a palavra "banco" possui diversos significados como ilustrada nas sentenças a seguir:

1. O banco faliu - (substantivo) organização financeira.
2. A casa fica depois do banco - (substantivo) estabelecimento bancário.
3. Não sente nesse banco - (substantivo) item de mobília.
4. Eu banco esse contrato - (verbo) assume os custos.

Essas são apenas algumas possibilidades de significados da palavra "banco". Vale ressaltar que em 1 e 2 tem-se um caso de polissemia ao passo que em 3 e 4 fala-se de homonímia. A polissemia é uma área de estudo complexa dentro da linguística e possui teorias divergentes formuladas pela linguística clássica e a linguística cognitiva (GRIES, 2015; RAVIN e LEACOCK, 2000). Independentemente da teoria subjacente, para se estabelecer o sentido correto de um item lexical é preciso conhecer o contexto onde ocorre, ou seja, as palavras que ocorrem em sua vizinhança. Segundo Firth (1957 apud MONAGHAN, 1999), "se conhece uma palavra por sua companhia". No exemplo mencionado anteriormente, no item 3, é possível inferir que se trata de uma mobília devido a coocorrência do verbo "sentar". Em alguns casos é preciso conhecer também os micro papeis (FILLMORE, 2006) dos participantes do contexto para se compreender o significado de uma palavra. Por exemplo, a diferença de significado do verbo "substituir" nas sentenças "Tite substituiu Neymar" e "Firmino substituiu Neymar" só é possível identificar se os papéis de Tite como treinador e de Neymar e Firmino como jogadores são conhecidos[2].

[2] Adaptação de exemplo apresentado por Maria Margarida Martins Salomão.

Uma discussão aprofundada da polissemia foge do escopo deste trabalho, sendo que, no que se refere esta pesquisa, o importante é saber que uma palavra possui diversos significados e que o estabelecimento do significado correto para um determinado discurso é uma contribuição importante para compreender o enunciado como um todo. Independentemente do número de significados de uma palavra, quando se examina um único significado detalhadamente, é possível perceber que tal significado se trata de conceito complexo com diversas facetas. Primeiro é preciso responder à pergunta: O que é o significado de uma palavra? Autores como Alan Cruse e Dirk Geeraerts (CRUSE, 2011; CRUSE, 1986; GEERAERTS, 2010) dedicaram obras tratando apenas deste tema. O trabalho de Geeraerts apresenta detalhadamente as diversas teorias semânticas do léxico. De forma a validar a pesquisa corrente e definir o aspecto semântico tratado neste trabalho, foi adotada a teoria do léxico gerativo de James Pustejovsky (PUSTEJOVSKY, 1991). Segundo a teoria gerativa, associado ao léxico, existem pelo menos quatro níveis de representação: em estrutura argumental, estrutura de evento, estrutura de herança lexical e estrutura *qualia*. No caso desta pesquisa há interesse apenas na estrutura *qualia* que, segundo Pustejovsky, é responsável por registrar a força relacional de um item lexical. A estrutura *qualia* é composta pelos aspectos FORMAL, CONSTITUTIVO, TÉLICO e AGENTIVO (PUSTEJOVSKY, 1991). O aspecto CONSTITUTIVO estabelece relação entre um objeto e suas partes. O aspecto TÉLICO estabelece seu propósito e função. O aspecto AGENTIVO envolve sua origem. Finalmente, o aspecto FORMAL é o que o distingue em um domínio mais amplo. Por exemplo, de acordo com Pustejovsky, um livro de romance teria a seguinte estrutura em *qualia*:

$$\begin{cases} \textbf{Romance} \\ QUALIA = \begin{cases} CONSTITUTIVO = \boldsymbol{narrativa} \\ FORMAL = \boldsymbol{livro} \\ TÉLICO = \boldsymbol{leitura} \\ AGENTIVO = \textbf{escrito} \end{cases} \end{cases}$$

Dentre os aspectos descritos na estrutura *qualia*, o aspecto FORMAL é o que trata das características essenciais de um conceito. Ele revela acontece o reconhecimento no mundo, ou seja, seu aspecto ontológico. Esse aspecto da semântica é justamente o aspecto que este trabalho busca identificar, uma vez que a natureza formal de um determinado conceito permite eliminar ambiguidades. Assim,

se a palavra “livro” é anotada com a categoria *Creative_Work*[3] em uma sentença, como no caso de “Jorge Amado escreveu um livro que abordava a Capoeira”, “livro” refere-se a uma obra intelectual. Por outro lado, se a palavra “livro” é anotada com a categoria *Product* em uma sentença, como no caso de “Maria emprestou o livro”, “livro” tem sentido de um exemplar do livro.

É importante ressaltar que nesta pesquisa não foi adotada a classificação de Pustejovsky para o aspecto FORMAL. Para tanto, buscou-se uma ontologia com maior aplicabilidade, que é a ontologia Schema.org, descrita com maiores detalhes na seção 3.1.

2.1.2 Ontologias

Segundo Moreira et al. (2004a) o termo ontologia é derivado do grego que significa *ontos*=ser e *logos*=palavra. Sua origem pode ser atribuída a Aristóteles na busca de enquadrar o que existia no mundo em dez categorias. Portanto, o objetivo deste ramo da filosofia é definir quais categorias devem ser usadas para classificar o que existe no mundo (GUARINO, 1998). A ontologia pode ser definida como o ramo da metafísica que estuda as categorias de coisas existentes no mundo, ou seja, é uma descrição de conceitos das entidades existentes e relacionamentos entre estas entidades (MOREIRA et al., 2004b). Fica então a cargo da ontologia o estudo das diversas entidades que existem, a descrição de tipos e estruturas dessas coisas, suas propriedades, eventos, relacionamentos e processos em toda a extensão do mundo real (GUARINO, 1998).

Em computação o termo ontologia está mais relacionado à estruturação dos conceitos do que no entendimento da natureza das coisas que existem (MOREIRA et al., 2004b). As ontologias são utilizadas para classificar os objetos de algum domínio com base em critérios preestabelecidos. Assim, dada uma sentença, por exemplo, é função da ontologia abstrair as entidades nela contida e classificá-las segundo algum domínio, área de conhecimento ou mesmo uma parte do mundo. Sob essa faceta, o principal objetivo da construção de ontologias e classificação de objetos segundo tal é permitir compartilhamento e reutilização de conhecimento para técnicas computacionais (MAEDCHE, STAAB, 2001).

[3] Aplicação das categorias de nível topo da ontologia Schema.org.

Nesse sentido, as ontologias quando utilizadas na ciência da computação precisam estar munidas de uma especificação formal para os conceitos por meio de linguagens específicas (GUARINO, 1998). Ao serem aplicadas juntamente com técnicas computacionais as ontologias representam o conjunto de regras das entidades e seus respectivos relacionamentos, estas conexões são elaboradas por especialistas, e só assim usuários podem criar consultas com os conceitos descritos (MOREIRA et al., 2004b).

Já as ontologias de nível topo tratam de categorias que capturaram, por meio de categorias gerais, a forma como o mundo é estruturado (GUARINO, 1997). São consideradas gerais porque descrevem conceitos mais abrangentes que não dependem de um problema específico. Sendo assim, elas são adequadas para fornecer informações que revelam a natureza essencial dos conceitos.

Existem diversas propostas para ontologias de nível topo. Como a *Descriptive Ontology for Linguistic and Cognitive Engineering* (DOLCE) (GANGEMI et al., 2002), a *Suggested Upper Merged Ontology* (SUMO) (NILES, PEASE, 2001) e a Cyc (LENAT et al., 1990). Para este trabalho foi selecionada a ontologia Schema.org, sendo que a sua descrição e justificativa para escolha são apresentadas na seção 3.1.

2.1.3 Linguística de Corpus

O ramo do PLN responsável pelas pesquisas utilizando bases textuais, denominadas *corpora*, é conhecido como linguística de *corpus*. De acordo com Leech (1997), a linguística de *corpus* é vista como um estudo de eventos linguísticos que se dá através de coleções de textos que podem ser tratados por dispositivos computacionais. Tais coleções são conhecidas como *corpora* eletrônicos. Os *corpora* podem ser usados em várias áreas de pesquisa devido à possibilidade de aplicação em diferentes problemas, que vão desde estudos da norma culta pela análise sintática à contextos mais abrangentes como a aprendizagem de idioma (LEECH, 1997).

Esquemas e ferramentas de anotação adquiriram uma dimensão importante na linguística de *corpus*, e vale ressaltar que a linguística de *corpus* tem como função principal analisar e estudar os dados com um tratamento empirista da linguagem, com base em um sistema probabilístico (SARDINHA, 2004). A ideia por trás do contexto que diz que a linguagem é vista como um sistema probabilístico tem sentido, uma vez

que, a língua pode ser vista mais como uma questão de probabilidade que de possibilidade.

Materiais que são considerados bases textuais, tais como livros, jornais, revistas, e outros textos impressos, não são considerados *corpus* pela linguística de *corpus*, em razão de não se encontram em formato computadorizado (ALUÍSIO, ALMEIDA, 2006). O conteúdo de um *corpus* então deve representar o aspecto linguístico do fenômeno estudado para se obter melhor desempenho das técnicas aplicadas. Nesse sentido, Biber (1993, p. 243) destaca que a representatividade de um *corpus* está relacionada a variedade de amostras disponíveis, e sugere que um *corpus* representativo quanto a população, idioma, texto e volume da amostra deve permitir a análise de várias distribuições.

Desse modo, o *corpus* deve possuir amostras completas da população alvo possibilitando evidenciar e quantificar padrões. Tal padronização se caracteriza por colocações ou estruturas que se repetem significativamente. Além disso, uma característica marcante quanto a representação da linguagem refere-se a extensão dos *corpora*, pois quanto maior o recurso mais inferências podem ser obtidas, o que representa melhores chances de acurácia nos resultados.

2.1.4 Anotação de *Corpus*

No PLN, o conceito de anotação é amplamente difundido e também conhecido como anotação linguística ou anotação de linguagem natural. Segundo Leech (1997), anotação refere-se à prática de adicionar informações linguísticas a um *corpus*. Um *corpus* quando anotado transforma-se em um repositório de informação linguística, que, por ser feito de maneira explicita, favorece a recuperação e análise da informação rapidamente.

Para Sinclair (2004), a anotação de *corpus* é uma atividade perigosa que de alguma forma afeta negativamente a integridade do texto. Segundo o autor, existe uma grande chance de pesquisadores observarem apenas os dados do *corpus* através das *tags* e, por isso, qualquer informação que não é percebida através das *tags* é perdida.

Neste contexto, Pustejovsky e Stubbs (2012) mostram que o processamento de uma linguagem natural é muito mais complexo que simplesmente fornecer uma grande quantidade de informações ao computador e esperar que ele aprenda. Os

autores relatam que “qualquer *tag* de metadados usada para marcar elementos de um conjunto de dados é chamada de anotação”. Assim a linguística deve caminhar junto com a linguística computacional a fim de capturar as propriedades computacionais das estruturas linguísticas.

As três versões supracitadas em torno da anotação, apesar de terem sido levantadas por diferentes autores em diferentes épocas, convergem que o processo de anotação é importante tanto para linguistas quanto para cientistas da computação e demais pesquisadores da área. E, tal processo tange um elo crítico no desenvolvimento de tecnologias inteligentes no quesito linguagem humana.

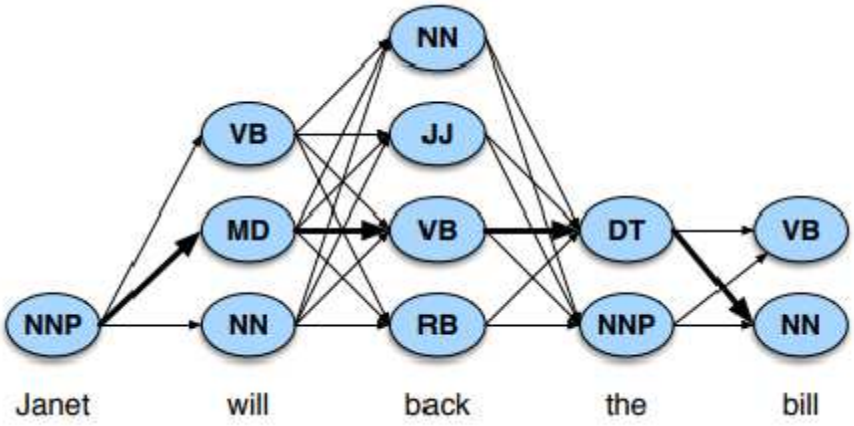
Com efeito, um *corpus* pode conter anotações de diferentes tipos ou várias versões de um único tipo de anotação. A anotação pode ser um processo manual (realizada por pessoas, anotadores, manualmente sem assistência de um programa de computador), automático (utiliza uma ferramenta de anotação que faz a etiquetagem automaticamente), ou semiautomático (utiliza ferramenta de anotação automática, porém permite que um anotador verifique e corrija possíveis erros de predição) (LEECH, 1997). A tarefa a qual se destina o *corpus* é que define o tipo da anotação a ser usada.

2.1.5 Anotação Semântica

Certas aplicações precisam lidar com interpretação de frases bem formadas sendo necessário conhecer o significado dessas construções. De acordo com Vieira e Lima (2001), “em um tratamento automático, a análise semântica consiste em associar a uma sequência de marcadores linguísticos, uma representação interna, entendida como a representação do significado dessa frase”. Tal sequência de marcadores geralmente é derivada da análise sintática.

A analise sintática avalia a gramática da linguagem utilizada e gera uma sentença da estrutura analisada (VIEIRA, LIMA, 2001). Em comparação com linguagens naturais, linguagens formais apresentam uma semântica bem definida. Portanto, existe uma grande influência da lógica nos estudos da semântica computacional de linguagem natural. De acordo com a estrutura sintática de uma frase, é possível estabelecer uma representação lógica correspondente, onde o verbo indica uma relação entre os argumentos expressos por sujeito e o complemento verbal (objeto direto ou indireto).

Figura 1 – Um esquema da tarefa de marcação.



Fonte: Jurafsky (2000)

A figura 1 apresenta um esquema de possíveis tags para cada palavra com uma sequencia final de tags corretas indicada pelas setas em negrito. Na frase “Janet will back the bill”, tem-se um nome próprio (Janet > NNP), seguido por um modal (will > MD), um verbo (back > VB), um determinante (the > DT) e um substantivo (bill > NN). As *tags* destacadas em azul na figura 1 foram retiradas do projeto Penn Treebank.

Tendo consciência dessa complexidade, a anotação semântica é uma anotação que tenta explorar o sentido dos dados que estão sendo marcados. Cabe a anotação semântica buscar elementos do texto e classificá-los de acordo com o seu significado no fragmento em que está inserido (MITKOV, 2004). Esse tipo de anotação é muito importante devido à variação e ambiguidade presentes na linguagem (KIRYAKOV et al., 2004). As palavras podem ter diferentes sentidos dependendo do contexto ao qual estão inseridos, então é função dos anotadores, expressar e anotar o sentido da palavra selecionada (HANDSCHUH, STAAB, 2003). A anotação semântica é de extrema importância e demanda bastante conhecimento prévio sobre a língua dos anotadores.

A anotação semântica pode trazer vários benefícios quanto a sua utilização em documentos não estruturados. Os documentos anotados semanticamente possuem um significado que transcende além do conteúdo explicitado através das informações contidas (MITKOV, 2004). Também, a anotação semântica permite que os dados sejam interpretáveis por aplicações de tal forma que as máquinas possam também interpretar o significado contido no conteúdo (REEVE, HAN, 2005). A recuperação de

informação é facilitada pelo processo de anotação semântica uma vez que se torna fácil acessar e entender a estrutura do documento analisado (KIRYAKOV et al., 2004).

A anotação semântica geralmente se baseia no conceito de ontologias para anotar e dar significado aos fragmentos do texto selecionado. As ontologias são responsáveis por guiar o processo de anotação transformando-se em classes em que as palavras serão marcadas (KIRYAKOV et al., 2004). As ontologias trazem consigo um conhecimento adicional advindo das relações existentes entre os níveis superiores e inferiores da classe ontológica (REEVE, HAN, 2005).

Existem três formas de realizar a anotação semântica em um *corpus* ou em documentos de textos: semiautomáticas, automáticas e híbridas (HANDSCHUH, STAAB, 2003). A anotação semântica semiautomática é constituída pela associação de palavras do texto a classes, instâncias e propriedades da ontologia utilizando-se de julgamento humano. O modelo automático de anotação semântica faz uso de técnicas de processamento de linguagem natural, aprendizado de máquina e extração de informação para associar as palavras selecionadas do documento à ontologia escolhida. E por fim, a anotação híbrida trata da combinação de modelos automáticos e semiautomáticos (HANDSCHUH, STAAB, 2003).

2.2 Aprendizado de Máquina

Aprendizado de Máquina (AM) refere-se ao método que analisa dados e automatiza o desenvolvimento de modelos analíticos usando algoritmos capazes de aprender interativamente ou melhorar seu desempenho com base na experiência (PUSTEJOVSKY, STUBBS, 2012). A composição de técnicas computacionais em AM é possível porque dados não aleatórios geralmente possuem algum padrão, e tais padrões permitem que a máquina extraia generalizações. Assim, podem ser localizadas no texto características sobre a massa de dados cujo modelo foi treinado.

Durante a fase de aprendizagem, os parâmetros numéricos que caracterizam o modelo implícito de um determinado algoritmo são calculados pela otimização de uma medida numérica, geralmente através de um processo iterativo (MANNING, 1999). Aplicar a estatística e a aprendizagem de máquina em linguística de *corpus* envolve o desenvolvimento e/ou uso de algoritmos que permitam um programa inferir padrões sobre dados de treinamento, que viabilizem previsões sobre novos dados (SARDINHA, 2004).

O reconhecimento desses padrões pode ser obtido com o uso de técnicas de AM. Russell e Norving (1995 apud CARVALHO, 2012) consideram que um dos fatores mais importantes na determinação da natureza do problema de aprendizado é o tipo de retorno disponível para aprendizado no qual o algoritmo se depara. O aprendizado de máquina regularmente é divido em três categorias:

Supervisionado: A aprendizagem supervisionada acontece quando o modelo está sendo treinado em um conjunto de dados rotulado. O conjunto de dados rotulado é aquele que possui parâmetros de entrada e saída. Neste tipo de aprendizagem, ambos conjuntos de dados, de treinamento e validação, são rotulados (ALPAYDIN, 2009).

Não supervisionado: O aprendizado não supervisionado é o treinamento da máquina que usa informações que não são classificadas nem rotuladas, permitindo que o algoritmo aja com base nessas informações sem orientação. Com esse tipo de aprendizagem, a tarefa da máquina consiste em agrupar informações não classificadas de acordo com semelhanças, padrões e diferenças sem qualquer treinamento prévio de dados (ALPAYDIN, 2009).

Reforço: Trata-se de tomar medidas adequadas para maximizar a recompensa em uma situação particular com o objetivo de encontrar o melhor comportamento ou caminho possível em determinado ambiente. No aprendizado por reforço um agente aprende, com sua experiência, como se comportar em um ambiente executando ações e vendo os resultados sem depender de algum conjunto de dados rotulados (SUTTON, BARTO, 2018).

Para que o aprendizado de máquina possa ocorrer é preciso ter disponível bases de dados com tipos e tamanhos apropriados. O formato irá variar conforme o tipo de aprendizado que será empregado. Se o aprendizado for supervisionado será necessário uma base com dados previamente classificados, enquanto que no caso de um método não supervisionado tal requisito não será exigido. Quanto ao tamanho da base de dados, este pode variar desde alguns milhares de dados, adequado para métodos generativos, tais como *Naïve Bayes*, até alguns milhões ou bilhões de dados, sendo um tamanho apropriado para métodos de aprendizado profundo. Em se tratando de métodos de aprendizado de máquina voltados para o processamento de linguagem natural, o conjunto de dados de treinamento será compostos de instâncias de enunciados e textos, denominados de *corpus* produzidos por humanos, tanto usando a linguagem escrita como a oral. Muitas vezes a base de textos é anotada

com *tags* que destacam recursos específicos e relevantes para a tarefa de aprendizagem.

2.2.1 Aprendizado Supervisionado

Existe uma grande variedade de tarefas que podem ser resolvidas com o AM. Duas tarefas bastante comuns são a regressão e a classificação, ambas relacionadas à predição. Os problemas de regressão e classificação pertencem à categoria supervisionada do AM (ALPAYDIN, 2009). No AM supervisionado, conforme descrito anteriormente, um modelo ou uma função é aprendida a partir dos dados do treinamento para prever dados futuros. A regressão prevê um valor a partir de um conjunto contínuo, enquanto a classificação prediz o pertencimento de uma entrada a determinada classe (KOTSIANTIS et al., 2007).

A natureza e o grau de supervisão fornecidos pelos vetores da saída desejada (*target*) variam muito entre as tarefas de aprendizagem supervisionada. Por exemplo, treinar um algoritmo supervisionado para rotular corretamente cada pixel correspondente a uma pessoa em uma imagem requer um objetivo muito mais informativo do que simplesmente treiná-lo para reconhecer se uma pessoa está ou não presente em um ambiente. Para distinguir esses extremos, muitos estudiosos se referem aos dados como fracos e fortemente rotulados (HORNIK et al., 1989).

De acordo com Gareth et al. (2013) uma tarefa de aprendizado supervisionado padrão consiste então em um conjunto de treinamento S de pares mapeados para categorias desejadas (x, z), em que x é um elemento do espaço de entrada X e z é um elemento do espaço de saída Z, e um conjunto disjunto de teste S, ambos elaborados a partir da mesma distribuição $D_{X \times Z}$. Em alguns casos, um conjunto extra de validação é extraído do conjunto de treinamento para validar o desempenho do algoritmo de aprendizado.

O objetivo da tarefa é usar o conjunto de treinamento para minimizar alguma medida de erro, E, específica extraída do conjunto de testes (FRIEDMAN et al., 2001). Por exemplo, em uma tarefa de regressão, uma medida de erro que pode ser aplicada é a soma dos quadrados entre as saídas do algoritmo e as saídas desejadas. Em algoritmos de redes neurais, a abordagem comumente utilizada para a minimização de erros é o ajuste dos parâmetros do algoritmo para otimizar uma função de custo, O, no conjunto de treinamento, onde O está relacionado, mas não necessariamente é

idêntico a *E*. A capacidade de um algoritmo de transferir o desempenho do conjunto de treinamento para o conjunto de teste é chamada de generalização (BISHOP, 2006). Como descrito por Friedman et al. (2001), “a avaliação deste desempenho é extremamente importante na prática, uma vez que orienta a escolha do método ou modelo de aprendizagem, e fornece uma medida da qualidade do modelo escolhido”.

2.2.2 Reconhecimento de Padrões

O reconhecimento de padrões, que tem por objetivo a classificação de objetos em categorias ou classes, tem sido extensivamente estudada na literatura de AM (BISHOP, 2006; DUDA et al., 2012), e vários algoritmos de classificação de padrões, como Perceptron Multicamadas (RUMELHART et al., 1985; BISHOP, 1995) e Máquinas de Vetores de Suporte (VAPNIK, 2013), são usados regularmente em diferentes tipos tarefas.

Como o modelo proposto nesta dissertação aborda exclusivamente a rotulagem de palavras de forma supervisionada, assume-se a presença de um conjunto de treinamento *S* de pares de entrada-saída desejada (*x, z*), onde cada entrada *x* é um vetor com valor real de um comprimento fixo, *M* e cada destino *z* representa uma única classe desenhada de um conjunto de classes *K*.

Um pressuposto implícito à maioria dos trabalhos sobre classificação de padrões supervisionados é que os pares de entrada-saída desejada são independentes e identicamente distribuídos. Então, considera-se que *h* é um classificador padrão que mapeia vetores de entrada para rótulos de uma classe, e *S′* um conjunto de teste de pares de entrada-saída desejada que não estão contidos em *S*. Para tarefas de classificação de padrões em que todas as classificações erradas são igualmente ruins, o objetivo é usar *S* para treinar um classificador de forma a minimizar a taxa de erro de classificação E^{classe} em *S,* como demostra a equação (1).

$$E^{classe}(h, S') = \frac{1}{|S'|} \sum_{(x,z)\in S'} \begin{cases} 0, & se\ h(x) = z \\ 1, & caso\ contrario \end{cases} \tag{1}$$

No caso mais geral, em que erros diferentes têm pesos diferentes, pode-se definir uma matriz de perdas L cujos elementos são as perdas incluídas pela atribuição

de um padrão com a classe verdadeira C_i à classe C_j. Nesta situação, procura-se minimizar a perda da classificação E_{perda} em S′, como pode ser observado na equação (2).

$$E^{perda}(h, L, S') = \frac{1}{|S'|} \sum_{(x,z)\in S'} L_{h(x)z'} \quad (2)$$

2.3 Aprendizado Profundo

Por mais que apresentem bons resultados quanto a extração de informação em alta velocidade, as técnicas convencionais de aprendizagem de máquina são limitadas em sua capacidade de processar dados naturais em forma bruta. Durante décadas, a construção de um sistema de reconhecimento de padrões exigiu um trabalho minucioso e considerável especialização de domínio para projetar um extrator de recursos que transformasse dados brutos em uma representação interna adequada ou em vetor de recursos do qual o subsistema de aprendizado, geralmente um classificador, pudesse detectar ou classificar padrões na entrada.

Os métodos de aprendizagem profunda são considerados um tipo específico de AM com os quais o computador aprende por experiência através de uma hierarquia de conceitos (GOODFELLOW et al., 2016). Tal hierarquia permite que um grande problema seja dissolvido em problemas menores, o que facilita o processo de aprendizagem, isto é, modelos computacionais compostos por múltiplas camadas de processamento aprendem representações de dados em múltiplos níveis um pouco mais abstratos. Em vista disso, os métodos de aprendizagem profunda melhoraram eficientemente o estado da arte em PLN.

2.4 Redes Neurais Artificiais

De modo genérico, redes neurais artificiais (RNAs) são modelos simplificados do sistema nervoso biológico (RAJASEKARAN, PAI, 2003). Singh e Chauhan (2009) definem as RNA como um modelo matemático que é baseado em redes neurais biológicas e, portanto, é uma emulação de um sistema neural biológico. Pode-se dizer que uma rede neural é um sistema de processamento de dados, constituído de um

grande número de elementos de processamento simples e altamente interconectados (neurônios artificiais), em uma arquitetura inspirada na estrutura do cérebro. As redes neurais fornecem uma ferramenta de aprendizagem robusta adequada para uso em problemas de linguagem natural. Sua capacidade diminui, em certa medida, os problemas de descrição e de dispersão de dados (MEHROTRA et al., 1997).

Em comparação com os algoritmos convencionais, as redes neurais podem resolver problemas que são bastante complexos, em um nível substancialmente mais fácil em termos de complexidade do algoritmo. Portanto, a principal razão para usar RNA é devido a sua estrutura simples e natureza auto organizada que lhes permite resolver uma gama de problemas sem qualquer interferência adicional do programador. Tal como no cérebro humano, numa RNA a unidade de base de processamento é o neurônio.

Existem conexões ponderadas entre esses neurônios que podem ser adaptadas durante o processo de aprendizado da rede cuja função de ativação define o valor de saída de cada nó dependendo de seus valores de entrada (MEHROTRA et al., 1997). Toda rede neural é constituída por diferentes camadas. A camada de entrada recebe informações de fontes externas, como valores de atributos da entrada de dados, a camada de saída produz a saída da rede, e as camadas ocultas conectam a entrada e a saída entre si. O valor de entrada de cada nó em cada camada é calculado pela soma de todos os nós de entrada multiplicados pelo respectivo peso da interconexão entre os nós (ERB, 1993).

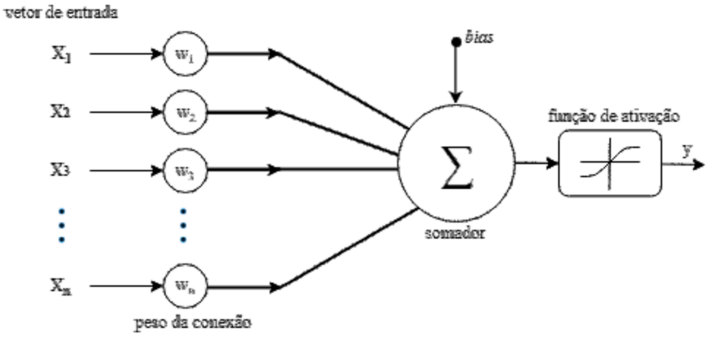
O modelo matemático tradicional de um neurônio artificial, resumido na equação (3), proposto por Mcculloch e Pitts (1943), pode ser melhor exemplificado de acordo com a figura 2. Há uma camada que recebe sinais de entrada, pesos referentes aos estímulos recebidos, uma função de soma, e finalmente uma função de ativação que definirá qual será a saída do neurônio.

$$y = f\left\{\sum_{i=1}^{n} x_i \times w_i\right\} \tag{3}$$

1. As entradas (x_1, x_2, ..., x_n) podem ser as saídas de outros neurônios, entradas externas, um *bias* ou mesmo uma combinação desses elementos.

2. Pesos (w_1, w_2, ..., w_n) são valores para ponderar os sinais de cada entrada da rede. Esses valores são aprendidos durante o treinamento.
3. Somatório de todas estas entradas, multiplicadas por seus respectivos pesos.
    1. *Bias* ou Tendência é uma constante que ajuda o modelo de maneira que ele possa se adequar melhor aos dados fornecidos (BISHOP, 2006). Uma unidade *bias* basicamente é um neurônio extra adicionado a cada camada pré-saída que armazena um valor **x**. Essas unidades não estão conectadas a nenhuma camada anterior e, nesse sentido, não representam uma atividade verdadeira. Esta variável é incluída ao somatório da função de ativação, com o intuito de aumentar o grau de liberdade desta função e, consequentemente, a capacidade de aproximação da rede. O valor do *bias* é ajustado da mesma forma que os pesos.
4. A função de ativação define o modo como a saída é manipulada com base na entrada de uma rede neural (BISHOP, 2006). Ela basicamente decide se um neurônio deve ser ativado ou não. Ou seja, se a informação que o neurônio está recebendo é relevante para a informação fornecida ou deve ser ignorada.
5. Saída (y) é o resultado final do processamento.

Figura 2 – Modelo básico de um neurônio artificial.



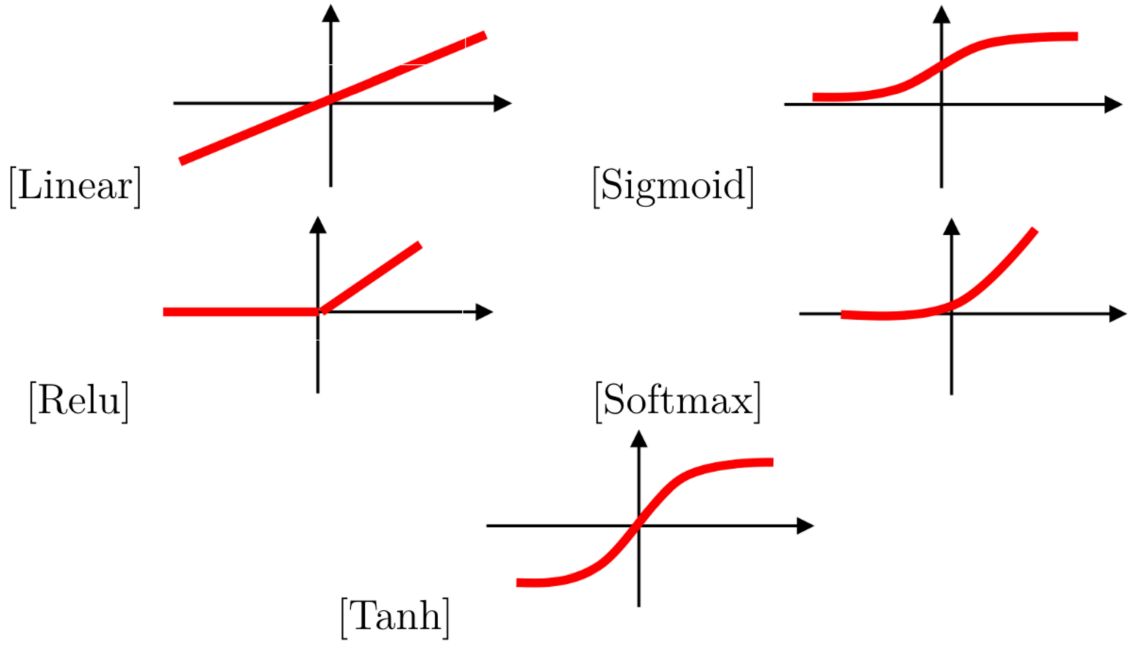
Fonte: Adaptado de Haykin (2007).

As entradas ***x*** ao serem apresentadas ao neurônio são multiplicadas pelos pesos correspondentes, gerando entradas ponderadas, em que, x_1 multiplica w_1, e assim sucessivamente. Essa etapa descreve uma das bases matemáticas do funcionamento de uma rede neural artificial, a multiplicação de matrizes. Após multiplicar, o neurônio então soma todos os produtos e produz um único resultado.

Um neurônio dispara quando a soma dos impulsos que ele recebe ultrapassa o seu limiar (*threshold*) (MEHROTRA et al., 1997). A ativação do neurônio é então obtida através da aplicação de uma função de ativação, que ativa ou não a saída, dependendo do valor da soma ponderada das suas entradas.

Definir qual função de ativação será utilizada em uma rede neural é relevante devido as particularidades que deverão ser observadas nos dados de entrada. A figura 3 mostra os gráficos gerados pelos tipos mais populares: Linear, Sigmoide, Tangente Hiperbólica (TanH), Unidade Linear Retificada (ReLU) e Softmax.

Figura 3 – Funções de ativação.



Fonte: Oliveira, 2018.

Mehrotra et al. (1997) afirmam que alguns métodos de treinamento variam a taxa na qual uma rede é modificada, e a quantidade de alterações feitas em cada etapa é muito pequena na maioria delas para garantir que a rede não se afaste muito do estado evoluído. A grande vantagem no uso de redes neurais artificiais para solução de problemas complexos provém principalmente de sua capacidade de aprendizagem através de exemplos e generalização da resposta (SANTOS et al., 2005). Com isso, as RNAs são utilizadas em problemas onde uma solução analítica ou numérica não pode ser obtida.

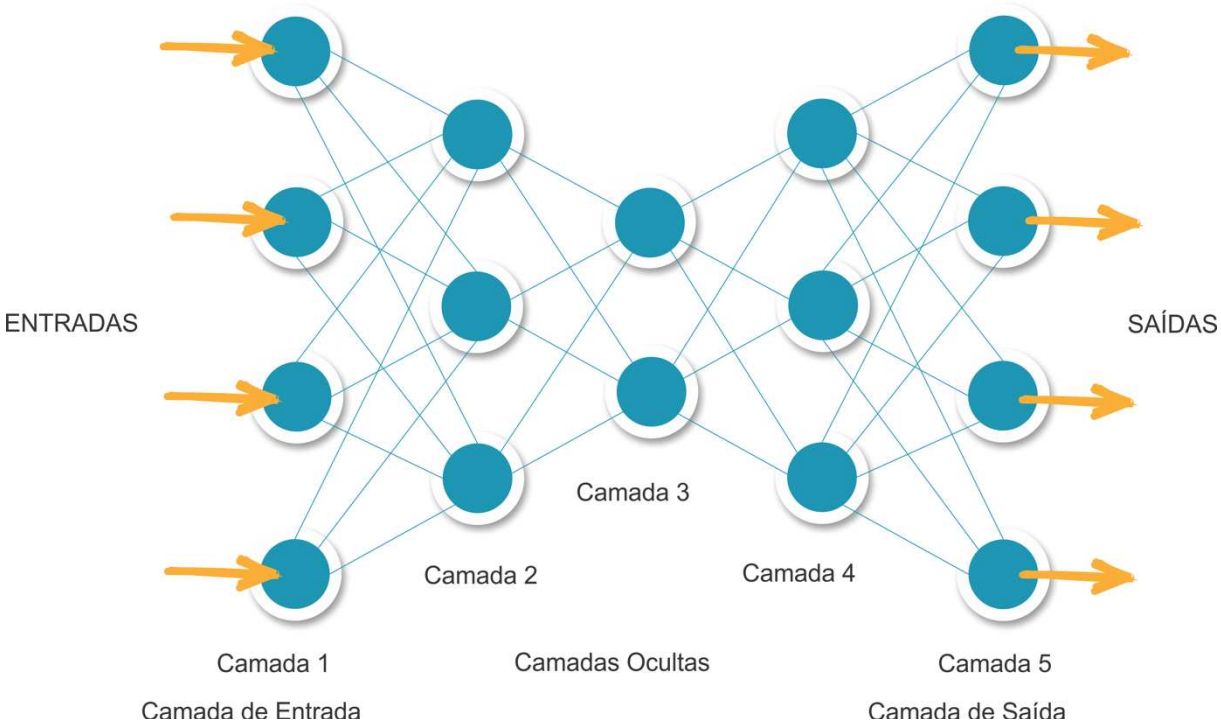
Muitas variedades de RNAs surgiram ao longo dos anos com propriedades muito variadas. Uma distinção importante entre as RNAs destaca e diferencia aquelas

cujas conexões formam ciclos e aquelas cujas conexões são acíclicas. As RNAs com ciclos são chamadas de redes neurais de *feedback*, recursivas ou recorrentes, ao passo que as RNAs sem ciclos são conhecidas como redes neurais *feedforward* (FNNs) (BISHOP, 2006).

As redes neurais *feedforward*, em particular *multilayer perceptrons* (MLPs), permitem trabalhar com entradas de tamanho fixo, ou com entradas de comprimento variável nas quais pode-se desconsiderar a ordem dos elementos conforme representação na figura 4 (MEHROTRA et al., 1997). Ao alimentar a rede com um conjunto de componentes de entrada, a rede aprende a combiná-los de forma significativa.

MLPs podem ser usados sempre que um modelo linear for usado anteriormente. A não linearidade da rede, bem como a capacidade de integrar facilmente *embedding words* pré-treinadas muitas vezes levam a uma melhor acurácia na classificação.

Figura 4 – Modelo de uma rede *feedforward.*



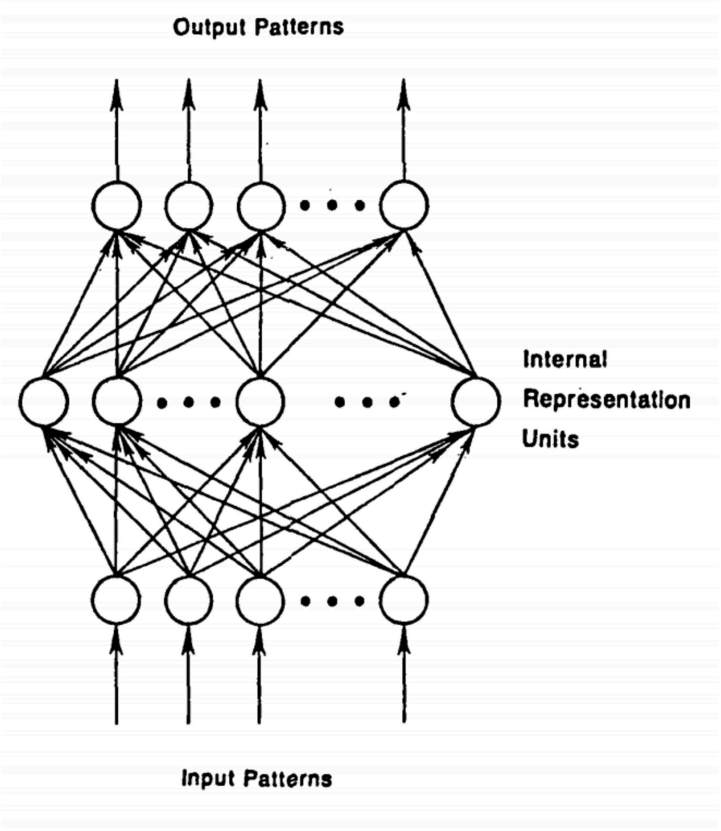
Fonte: Adaptado de Mehrotra et al. (1997)

As redes *feedforward* são arquiteturas especializadas que se destacam na extração de padrões locais nos dados. As *feedforward* são alimentadas com entradas de tamanho aleatório e são capazes de extrair padrões locais significativos que são

sensíveis à ordem das palavras, independentemente de onde elas aparecem na entrada (BISHOP, 2006). Esse tipo de arquitetura trabalha na identificação de frases indicativas ou expressões idiomáticas de tamanho fixo, em frases longas ou documentos. As unidades em um *perceptron* multicamadas são organizadas em camadas, com conexões alimentando-se de uma camada para outra, o ideário de Rumelhart et al. (1985) pode ser observado na figura 5.

Os padrões de entrada são apresentados à camada de entrada e as ativações de unidade resultantes são propagadas pelas camadas ocultas até a camada de saída (MEHROTRA et al., 1997). Esse processo é conhecido como o passo para frente da rede (*forward pass*). As unidades nas camadas ocultas possuem funções de ativação (tipicamente não lineares) que transformam as ativações já somadas que chegam à unidade. Como a saída de um MLP depende apenas da entrada atual, e não de entradas passadas ou futuras, as MLPs são mais adequadas para a classificação de padrões do que para a rotulagem de sequências.

Figura 5 – Um perceptron multicamadas totalmente conectado com uma camada oculta.



Fonte: Rumelhart et al. (1985).

Um MLP pode ser pensado como uma função que mapeia vetores de entrada para saída. Como o comportamento da função é parametrizado pelos pesos da conexão, um único MLP é capaz de instanciar muitas funções diferentes. De fato, foi provado (HORNIK et al., 1989) que um MLP com uma única camada oculta contendo um número suficiente de unidades não lineares pode aproximar qualquer função contínua em um domínio de entrada compacto a uma precisão arbitrária. Por essa razão, as MLPs são consideradas aproximadoras universais.

2.4.1 Processos de Aprendizagem

O processo de aprendizado em uma rede neural está intimamente relacionado com a maneira como pessoas aprendem. Por exemplo, ao realizar uma ação um instrutor/treinador aceita ou a corrige a fim de melhorar o desempenho em determinada tarefa. Da mesma forma, as redes neurais requerem um treinador para descrever o que deveria ter sido produzido como resposta à entrada. Com base na diferença entre o valor real e o valor que foi gerado pela rede, um valor de erro é calculado e enviado de volta pelo sistema. Para cada camada da rede, o valor do erro é analisado e usado para ajustar o limite e os pesos para a próxima entrada. Dessa forma, o erro continua se tornando menor a cada execução, à medida que a rede aprende a analisar valores.

De acordo com Barreto (1999), um algoritmo que tem foco no aprendizado apresenta um conjunto bem definido de diretrizes para resolver um problema. São vários os algoritmos de aprendizagem e muitos deles são exclusivos para certos modelos de rede. Uma característica determinante entre esses tipos está na forma como o peso é modificado além, é claro, do modo como a rede neural se relaciona com o ambiente.

Como caracteriza Haykin (2007), “uma apresentação completa do conjunto de treinamento inteiro equivale a 1 ciclo ou época”. Segundo Mehrotra et al. (1997), existem duas abordagens em torno do processo de correção dos pesos durante a fase de aprendizado:

1. Modo Padrão (Incremental): os pesos são alterados após cada apresentação de amostra do conjunto de treinamento.
2. Modo *Batch:* os pesos são atualizados somente após todas as amostras terem sido apresentadas à rede. Desse modo, as mudanças de peso

sugeridas por diferentes amostras de treinamento são acumuladas em uma única alteração que ocorre no final de cada época.

Ambos os métodos são utilizados e em cada caso, o treinamento é executado até que um erro razoavelmente baixo seja alcançado, ou até que o número máximo de iterações alocadas para treinamento seja excedido. Na literatura, *iteração* geralmente refere-se a uma época.

2.4.2 Regularização

Embora as funções de custo para treinamento de rede sejam, necessariamente, definidas no conjunto de treinamento, o objetivo final é obter o melhor desempenho em dados ainda não vistos no conjunto de teste. O problema cujo desempenho do conjunto de treinamento é transferido para o conjunto de testes chamado de generalização é de fundamental importância para o aprendizado de máquina (VAPNIK, 2013; BISHOP, 2006).

Existem casos em que a RNA pode se especializar, após o processo de treinamento, de modo excessivo em relação aos padrões do conjunto de treinamento, gerando um problema de aprendizado conhecido como super aprendizado ou *overfitting* (BISHOP, 2006). Geralmente este problema pode ser evitado por meio do método *early stopping*. Este método divide o conjunto de padrões em um novo conjunto de treinamento e validação, após cada varredura do conjunto de treinamento a rede é avaliada com o conjunto de validação (BISHOP, 2006). O algoritmo para quando o desempenho com o teste de validação deixa de melhorar.

Early stopping é talvez o método mais simples e universalmente aplicável em prol de uma generalização aprimorada. No entanto, uma desvantagem dessa técnica é que parte do conjunto de treinamento precisa ser sacrificado para o conjunto de validação, o que pode reduzir desempenho, especialmente se o conjunto de treinamento for pequeno. Outro problema é que não há como determinar a priori qual deve ser o tamanho do conjunto de validação.

Além do *early stopping*, o *dropout* é outro método de regularização criado por Srivastava et al. (2014) na tentativa de reduzir o *overfitting*. O *dropout* zera os pesos de alguns neurônios com uma probabilidade *p*. Desse modo, uma taxa de *dropout* de 0,1 significa que cada neurônio da rede possui 10% de chance de ter seu peso zerado na iteração atual. A aplicação do *dropout* acontece durante o treinamento da rede, e

a técnica proporciona aos neurônios o aprendizado de diferentes características, tornando-os capazes de generalizar melhor.

O risco excessivo de *overfitting* pode ser minimizado também através de técnicas como a validação cruzada (*cross-validation*), que particiona aleatoriamente os dados de exemplo em conjuntos de treinamento e teste para validar internamente as previsões do modelo (BISHOP, 2006). Este processo de particionamento, treinamento e validação de dados é repetido várias vezes, e os resultados da validação são calculados em uma média em todas as rodadas.

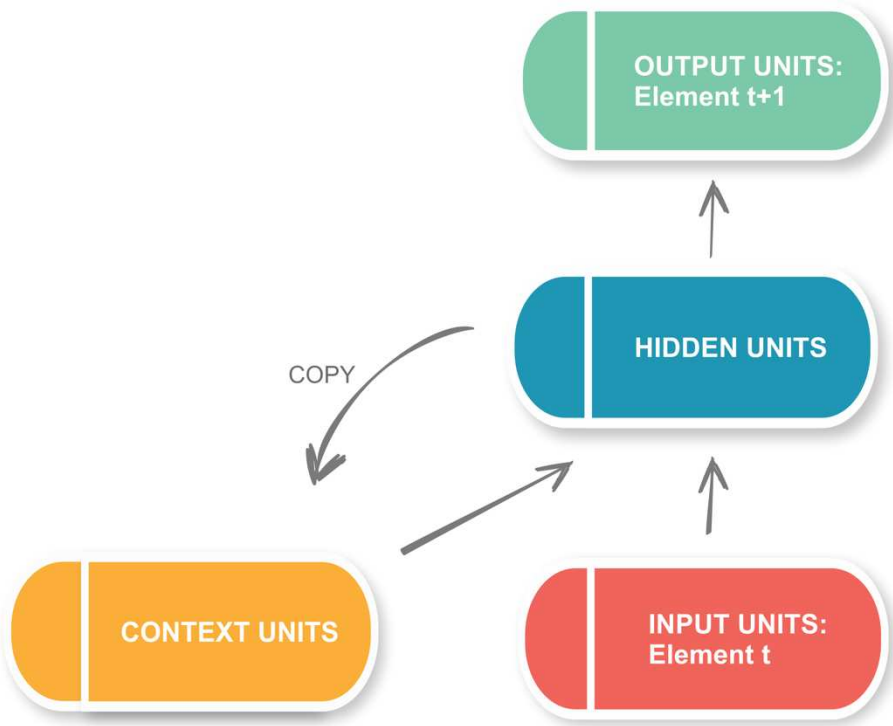
2.4.3 Redes Neurais Recorrentes

Em síntese, RNAs são conexões que não formaram ciclos. Ao desfazer dessa condição propiciando a formação de conexões cíclicas que mapeiam uma entrada para uma saída com base em uma rede de elementos de processamento altamente conectados, obtém-se uma Rede Neural Recorrente (RNN) (ELMAN, 1990), que é um modelo especializado em dados sequenciais. RNNs são componentes de rede que admitem como entrada uma sequência de itens e produzem um vetor de tamanho fixo que resume essa sequência. Uma RNN é usada como um transformador de entrada que é treinado para produzir representações informativas para a rede *feedforward* que funcionará acima dela (MEHROTRA et al., 1997).

Importante ressaltar que conexões recorrentes permitem uma memória de entradas anteriores atuar no estado interno da rede. RNNs são modelos muito sugestivos para se trabalhar com sequências, posicionando-se assim como uma ferramenta para processamento de texto ao projetar modelos que podem condicionar frases inteiras, levando em conta a ordem das palavras quando é necessário, evitando assim muitos problemas de estimativa decorrentes da dispersão de dados. A tarefa de prever a probabilidade da próxima palavra em uma sequência sugere ganhos significativos na modelagem de uma linguagem.

Conforme o modelo proposto na figura 6, em um arquitetura de rede recorrente simples, a camada de unidade oculta permite realimentação sobre si mesma, de modo que os resultados do processamento no tempo t-1 possam influenciar os resultados do processamento no tempo t. Na prática, a rede recorrente é implementada copiando o padrão de ativação nas unidades ocultas para um conjunto de unidades de contexto que se alimentem na camada oculta junto com as unidades de entrada.

Figura 6 – Modelo de Rede Recorrente Simples.



Fonte: Adaptado de Servan-Schreiber et al. (1991).

Cada caixa representa um conjunto de unidades e cada seta para frente representa um conjunto completo de conexões treináveis de cada unidade de envio para cada unidade receptora no próximo conjunto. A seta para trás da camada oculta para a camada de contexto denota uma operação de cópia.

2.5 Long Short-Term Memory

Como discutido anteriormente, um benefício do uso das redes recorrentes trata da capacidade que a rede tem de usar informações contextuais ao mapear sequências de entrada e saída. Contudo, em arquiteturas padrões de RNN, o intervalo de contexto que pode ser acessado é limitado. O problema é que a influência de uma determinada entrada na camada oculta e, portanto, na saída da rede, se perde ou transcende exponencialmente à medida que circula pelas conexões recorrentes da rede.

Na prática, essas deficiências, referidas na literatura como *vanishing gradient problem* e *exploding gradient problem*, dificultam o processo de aprendizagem de uma RNN pois o treinamento fica extremamente lento. A solução mais eficaz para resolver

o problema de dissipação de gradiente foi proposta por Hochreiter e Schmidhuber (1997), a arquitetura Long Short-Term Memory (LSTM).

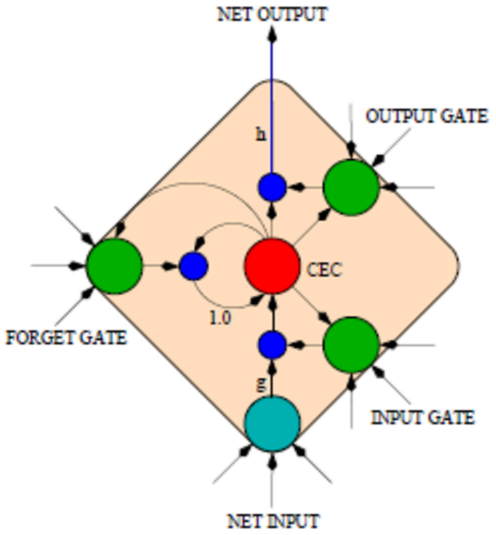
Para Goldberg (2017), a rede LSTM consiste em um conjunto de sub-redes conectadas recorrentemente, conhecidas como blocos de memória. Esses blocos podem ser considerados uma variação dos chips de memória em um computador digital. Cada bloco contém uma ou mais células de memória auto conectadas e três unidades multiplicativas (as portas de entrada, saída e esquecimento) que fornecem sinais contínuos de operações de gravação, leitura e redefinição para as células.

LSTM é uma arquitetura de RNN que lembra valores em intervalos arbitrários. Os valores armazenados não são modificados à medida que o aprendizado avança. É um tipo de rede adequada para classificar, processar e prever séries temporais com atrasos temporários de tamanho e duração desconhecidos entre eventos importantes (BYEON et al., 2015). A indiferença relativa ao comprimento do intervalo proporciona uma vantagem a rede LSTM em relação a RNNs alternativas, modelos ocultos de Markov e outros métodos de aprendizagem de sequências.

Destaca-se que, em sua forma original, a LSTM continha apenas portas de entrada e saída. Os portões de esquecimento "*forget gate"*, idealizados por Gers et al. (1999), foram adicionados posteriormente. O portão de esquecimento foi criado com o objetivo de fornecer um meio para as células de memória se reinicializarem, o que se mostrou importante para as tarefas que exigiam que a rede esquecesse entradas anteriores.

A figura 7 fornece uma ilustração de um bloco de memória LSTM com uma célula única. Uma rede LSTM, conforme citado anteriormente, é formada exatamente como uma RNN, a diferença é que as unidades não lineares na camada oculta são substituídas por blocos de memória. De fato, os blocos LSTM podem ser misturados com unidades simples, caso seja necessário. Além disso, como acontece com outros tipos de RNNs, a camada oculta pode ser anexada a qualquer tipo de camada de saída, dependendo da tarefa requerida (regressão, classificação, entre outras).

Figura 7 – Bloco de memória LSTM com uma célula.



Fonte: GRAVES et al., 2008.

De acordo com Graves et al. (2008), os portões de entrada, saída e esquecimento coletam ativações de dentro e de fora do bloco que controlam a célula através de unidades multiplicativas, representadas na figura 7 como pequenos círculos. Os portões permitem que as células armazenem e acessem informações por períodos de tempo mais longos, evitando assim o problema de dissipação de gradiente.

2.5.1 LSTM Bidirectional

O modelo LSTM tradicional supracitado tem todos os seus estados limitados a uma única direção, *forward*. Contudo, existem muitas tarefas cujo sistema é solicitado a levar em consideração informações passadas. Por exemplo, um modelo de linguagem que tenta prever a próxima palavra com base nas palavras anteriores. Nesse sentido, a palavra anterior é tão importante quanto a palavra posterior.

Schuster e Paliwal (1997) propuseram uma estrutura RNN bidirecional para superar a limitação da direção única. A ideia é dividir os neurônios de uma RNN regular em uma parte que é responsável pela direção de tempo positiva (*forward states),* e outra pela direção negativa (*backward states).* As saídas dos estados de avanço não ficam conectadas aos estados de retorno. O funcionamento da rede acontece como se houvessem duas camadas ocultas separadas, as quais são vinculadas a partir da camada de entrada e se encontram na camada de saída.

A estrutura bidirecional também pode ser aplicada a rede LSTM modificando os neurônios ocultos por blocos de memória. Ao combinar as saídas de duas RNNs que passam informações em direções opostas torna-se possível capturar o contexto de ambas as extremidades da sequência. A arquitetura resultante desse processo é conhecida como BiLSTM (Bidirectional Long Short-Term Memory) (GRAVES, SCHMIDHUBER, 2005).

A propagação acontece em duas etapas (GRAVES, SCHMIDHUBER, 2005):

1. Movimento da esquerda para a direita. Parte-se do intervalo de tempo inicial calculando os valores até alcançar o intervalo final.
2. Movimento da direita para a esquerda. Inicia-se no intervalo de tempo final, calculando os valores até chegar no tempo inicial.

O uso da BiLSTM se deu inicialmente no domínio do reconhecimento de fala, porque, como descrito por Graves e Schmidhuber (2005) há evidências de que o contexto de todo o enunciado é usado para interpretar o que está sendo dito, ao invés de uma interpretação linear. Ao utilizar duas direções de tempo, os dados de entrada do passado e do futuro, considerando o momento atual, podem ser usados.

2.6 Trabalhos Correlatos

Esta seção descreve brevemente alguns trabalhos relacionados encontrados na literatura.

Sukhareva e Chiarcos (2015) propuseram um anotador baseado em ontologias para a rotulagem de PoS usando um banco de dados heterogêneo (*corpora* com anotações diferentes). Embora os autores treinassem a rede com *corpora* com diferentes conjuntos de *tags* e grau de granularidade, as anotações eram parcialmente compatíveis. O algoritmo utilizado na pesquisa foi uma rede neural *feedforward* com Rprop (retro propagação resiliente), ao passo que as classes foram especificadas pela OLiA (*Ontologies of Linguistic Annotation*). A intenção dos autores foi executar a marcação automática de PoS aplicando as categorias morfossintáticas presentes na OLiA, o que difere da abordagem apresentada no presente trabalho, uma vez que a ontologia aqui utilizada fornece conceitos observados em sites em um nível semântico, ou seja, refletem o que é observado na web.

Chiu e Nichols (2016) apresentam o uso de uma rede neural bidirecional híbrida LSTM e CNN para a tarefa de Reconhecimento de Entidade Nomeada (NER, *Named*

Entity Recognition). NER é um tipo de anotação semântica restrita a um número pequeno de classes, como Pessoa, Organização e Localização. Desse modo, técnicas que funcionam para NER podem fornecer bons resultados na anotação semântica para um maior número de classes. O sistema resultante provou ser competitivo no conjunto de dados CoNLL-2003 e superou o desempenho do estado da arte anteriormente relatado no conjunto de dados OntoNotes 5.0 em 2,13 pontos de *F1-score*. Em outro teste, foram usados dois léxicos construídos a partir de fontes disponíveis publicamente, e como resultado o sistema estabeleceu uma nova marca no estado da arte com 91,62 de *F1-score* no CoNLL-2003 e 86,28 no OntoNotes. Esta pesquisa tem uma forte interseção com o modelo desenvolvido neste trabalho, dado que os autores utilizaram uma rede neural LSTM bidirecional para lidar com o problema de anotação semântica. A diferença é que a tarefa do presente trabalho engloba um número maior de classes e, ao considerar essa perspectiva, é importante enfatizar que os resultados alcançados foram consideravelmente melhores com uma rede mais simples.

Mendonça Junior et al. (2016) também mostram uma rede neural LSTM bidirecional e CNN para executar NER, como no trabalho mencionado anteriormente. A principal diferença é que Mendonça Junior et al. (2016) aplicaram o sistema em vários *corpora* do português brasileiro. Apoiado na hipótese de que redes neurais profundas são atualmente a melhor abordagem para este tipo de tarefa, o modelo também obteve desempenho superior aos modelos desenvolvidos com classificadores tradicionais como CRF (*Conditional Random Field*).

No trabalho proposto por Andrade (2018), um anotador semântico, que utiliza as categorias de nível topo da ontologia Schema.org como rótulo, é apresentado. O autor u como modelo de classificação o método probabilístico de predição estruturada, CRF. Nos CRFs, os dados de entrada são sequenciais e deve-se levar em conta o contexto anterior ao fazer previsões em um conjunto de dados. Para modelar esse comportamento existe a Engenharia de *Features*, que é uma técnica que requer muita memória e processamento, mas produz bons resultados de classificação. O sistema proposto rendeu resultados significativos de predição, alcançando *F1-score* acima de 85% para todas as classes e uma média geral de 93,5%. O conjunto de dados utilizado por Andrade (2018) é o mesmo do presente trabalho, e em virtude desse aspecto, tais resultados foram utilizados como medida de comparação.

3 MODELO PROPOSTO

Corpora anotados desempenham um papel cada vez mais significativo na linguística computacional pois, através deles, é possível elaborar *features* com maior carga semântica e produzir sistemas de PLN que apresentem maior acurácia. Como resultado, há um aumento na demanda por anotações linguísticas de alta qualidade que possam capturar uma gama de fenômenos, especialmente no nível semântico, para apoiar o AM e a pesquisa em linguística computacional.

Diante dessa demanda, pesquisadores reconhecem a necessidade de *corpora* anotados que sejam facilmente acessíveis e estejam disponíveis para uso público. Para atender essas necessidades, a construção de um anotador semântico automático foi viabilizada nesta dissertação, o qual utiliza como etiquetas categorias ontológicas de nível topo.

3.1 A ontologia Schema.org

A Schema.org[4] foi a ontologia selecionada para este projeto por ser uma base flexível capaz de fornecer suporte a dados estruturados mais robustos, além de ser apoiada por grandes empresas de tecnologia. De fato, a Schema.org é um vocabulário que abrange entidades, relações entre entidades e ações.

A iniciativa é instituída pelo Google, Microsoft, Yahoo e Yandex, e o vocabulário é desenvolvido por um processo comunitário aberto. Esta ontologia é resultado de um esforço conjunto para melhorar a qualidade da web, sendo um esquema de marcação de dados estruturado que é suportado pelos principais mecanismos de busca. O vocabulário da Schema.org contém certa de 598 classes, 862 propriedades e 114 valores de enumeração.

Para este projeto, apenas os conceitos de nível topo foram usados porque, de acordo com Guarino (1998) “ontologias de nível topo descrevem conceitos muito gerais como espaço, tempo, matéria, objeto, evento, ação, entre outros, que são independentes de um determinado problema ou domínio”. As categorias presentes no nível topo da Schema.org usadas como rótulos nesta pesquisa, foram: *Action, Creative_Work, Event, Intangible, Organization, Person, Place* e *Product*.

[4] http://schema.org/

3.2 Características dos *Corpora*

Os *corpora* escolhidos para realizar os testes foram o *Open American National Corpus (OANC*) (FILLMORE et al., 1998) nos testes 1 e 2, e o Wikiner (NOTHMAN et al., 2013) no teste 3. Ambos foram selecionados porque são considerados *corpora* de significância para tratamento de diferentes problemas e desafios, e além disso, todos os dados e anotações são abertos e disponíveis para qualquer uso.

3.2.1 Open American National Corpus (OANC)

O primeiro conjunto de dados escolhido para testar o modelo proposto foi o OANC. Esse *corpus* foi escolhido devido ao seu tamanho e diversidade de gêneros textuais. Os textos presentes no *corpus* incluem diversos gêneros de arquivos escritos e transcrições de arquivos de fala produzidos a partir de 1990.

O volume de dados garante que o algoritmo seja capaz de capturar a coocorrência e as relações entre as palavras. O OANC contém cerca de 15 milhões de palavras do inglês contemporâneo e está disponível *online*, gratuitamente em http://www.anc.org. O *corpus* possui as seguintes características:

- Marcação estrutural de seções, capítulos, até o nível do parágrafo;
- Limites de sentença;
- Lema e palavras (*tokens*) etiquetadas usando o conjunto de *tags Penn Treebank*;
- Marcações de substantivo;
- Verbo;
- E entidades nomeadas (*Person, Location, Organization, Date).*

Essas anotações tornam o conjunto de dados ideal para uso em aprendizado supervisionado, mais um fator considerado em sua escolha.

Devido aos custos computacionais para anotar um grande volume de documentos, foi necessário selecionar um fragmento do *corpus* para tornar o desenvolvimento do projeto gerenciável em tempo hábil. O fragmento extraído do *corpus* para treinamento e teste da rede neural foi complementado com as tags de nível topo da Schema.org elaboradas por Andrade (2018).

Também foi necessário fazer uma limpeza dos dados excluindo informações irrelevantes, como marcas de parágrafo, espaços em branco e outras marcas estruturais. A figura 8 mostra um segmento do *corpus*, um trecho de uma sentença anotada pode ser observado em destaque na cor verde.

Figura 8 – Um segmento do *corpus* OANC anotado com a ontologia Schema.

```
<s><tok base="jame" msd="NNP" ne="PERSON" schema="PERSON"> James</tok>
<tok base="t." msd="NNP" ne="PERSON" schema="PERSON">T.</tok> <tok
base="moore" msd="NNP" ne="PERSON" schema="PERSON">Moore</tok> <tok
affix="ed" base="be" msd="VBD" ne="O" schema="O">was</tok> <tok affix="ing"
base="work" msd="VBG" ne="O" schema="ACTION">working</tok> <tok base=";"
msd=";" ne="O" schema="O">;</tok><tok base="special" msd="JJ" ne="O"
schema="O">special</tok><tok base="interest" msd="NN" ne="O" schema=
"INTANGIBLE"> interest</tok>
```

Fonte: Autoria própria.

Os valores expostos na tabela 1 foram obtidos após a limpeza do *corpus*.

Tabela 1 – OANC em números.

Itens	Quantidade
Sentenças de entrada	3.295.069
Tamanho do vocabulário	222.553
Tamanho máximo da sentença	50

Fonte: Autoria própria.

Todos os textos usados como entrada para o modelo seguem o formato Unicode, com divisão de frases, cujas palavras tem uma etiqueta *tag*, que é atribuída de acordo com a classe ontológica associada: *Action, Creative_Work, Event, Intangible, Organization, Person, Place, Product e Other* para casos em que a palavra não pertence a nenhuma das demais classes mencionadas.

3.2.2 Wikiner *corpus*

O segundo *corpus* utilizado foi o Wikiner, que é o resultado do trabalho de Nothman et al. (2013). O Wikiner é um *corpus* com anotações para o reconhecimento de entidades nomeadas composto por textos extraídos da Wikipédia em vários idiomas e gêneros. No teste realizado, apenas a parte disponível em inglês foi selecionada.

Outro aspecto fundamental é que este *corpus* tem uma frase por linha, cada *token* é separado por um espaço em branco e contém três itens: um *token* de texto, uma *tag PoS* e uma *tag Beginning-Inside-Outside* (BIO), como mostra a figura 9. Foi preciso limpar o *corpus* removendo marcadores, espaços e também as iniciais do padrão BIO das classes, deixando-o em um formato mais enxuto, um trecho anotado do *corpus* pode ser observado na cor verde em destaque.

Figura 9 – Um segmento do *corpus* Wikiner após limpeza.

```
According|VBG|O to|TO|O Peter|NNP|I-PER Kropotkin|NNP|I-PER
students|NNS|O from|IN|O the|DT|O Massachusetts|NNP|I-ORG
Institute|NNP|I-ORG of|IN|I-ORG Technology|NNP|I-ORG include|VBP|
O the|DT|O US|NNP|I-LOC member|NN|O of|IN|O the|DT|O French|
NNP|I-ORG Academy|NNP|I-ORG of|IN|I-ORG Sciences|NNPS|I-ORG
```

Fonte: Autoria própria.

A tabela 2 mostra o número de ocorrências presentes no *corpus* após limpeza. Vale ressaltar que o Wikiner foi anotado com cinco classes ontológicas: *Organization, Person, Place, Miscellaneous (Misc)* e *Other*. A classe *Misc* engloba palavras marcadas como *Event, Action, Product, Intellectual Production*, e *Intangible Things* no OANC.

Tabela 2 – Wikiner em números.

Itens	Quantidade
Sentenças de entrada	142.153
Tamanho do vocabulário	7.875
Tamanho máximo da sentença	50

Fonte: Autoria própria.

3.3 O modelo

Após definir o conjunto de dados a ser usado, o modelo e os dados foram estruturados para alimentar a rede. Todo o conjunto de dados foi convertido em forma vetorial usando o algoritmo *GloVe*, idealizado por Pennington et al. (2014)

O algoritmo *GloVe* gera uma representação na forma de vetores de números reais para cada palavra do *corpus* a partir do exame das coocorrências de palavras

dentro de um texto. Antes de treinar o modelo proposto, foi preciso construir uma matriz de coocorrência X, onde uma célula X_{ij} é um valor que representa com que frequência a palavra *i* aparece no contexto da palavra *j*. O primeiro passo foi gerar um vocabulário (mapeamento de palavras para IDs de palavras inteiras) para cada palavra em cada *corpus* e definir alguns parâmetros opcionais: como o tamanho da janela de contexto e uma contagem mínima (usada para eliminar pares de coocorrência de palavras raras).

Converter as palavras do conjunto de dados em um vetor de números reais é uma etapa essencial quando se trabalha com modelos de redes neurais, uma vez que é possível codificar as sentenças através de uma *embedding layer*. No modelo implementado o vetor foi definido com 200 elementos, e esse número foi obtido a partir de testes empíricos, sendo que tais testes mostraram que vetores com maior número de elementos não apresentaram melhorias significativas no desempenho do sistema.

A rede neural implementada foi do tipo *Bidirectional Long Short-Term Memory* (BiLSTM). A rede LSTM pode lidar com sentenças de tamanho arbitrário, detectando relações entre palavras distantes na sentença. O tipo bidirecional permite que esses relacionamentos sejam detectados nas palavras anteriores e nas palavras subsequentes à palavra atual.

A execução é iniciada definindo um vocabulário $V = \{v_1, v_2, ..., v\}$ que contém as palavras extraídas do *corpus* conforme descrito na seção 3.2. Assumiu-se que, para realizar o reconhecimento das entidades das palavras em qualquer frase, bastava considerar as informações contidas na sentença em questão. Portanto, a BiLSTM recebe uma sentença por vez. Para cada sentença de entrada de *n* palavras existe um vetor *n*-dimensional *x* cujos elementos são os índices em *V* correspondentes às palavras que aparecem na frase, preservando a ordem.

A entrada *x* é enviada para uma *embedding layer* que retorna a sequência $\mathbf{S} = \{w_j \mid j = x_1, x_2, ..., x_n\}$, onde w_j é a *j*-ésima linha de uma matriz densa $W \in R^{|V| \times d}$, em que d^{inN} é um hiperparâmetro. O vetor w_j configura uma *word embedding*, enquanto *W* é a matriz correspondente. A sequência de word embeddings S é então passada como entrada para duas camadas LSTM que a processam em direções opostas (para frente e para trás), similar à arquitetura introduzida por Graves e Schmidhuber (2005).

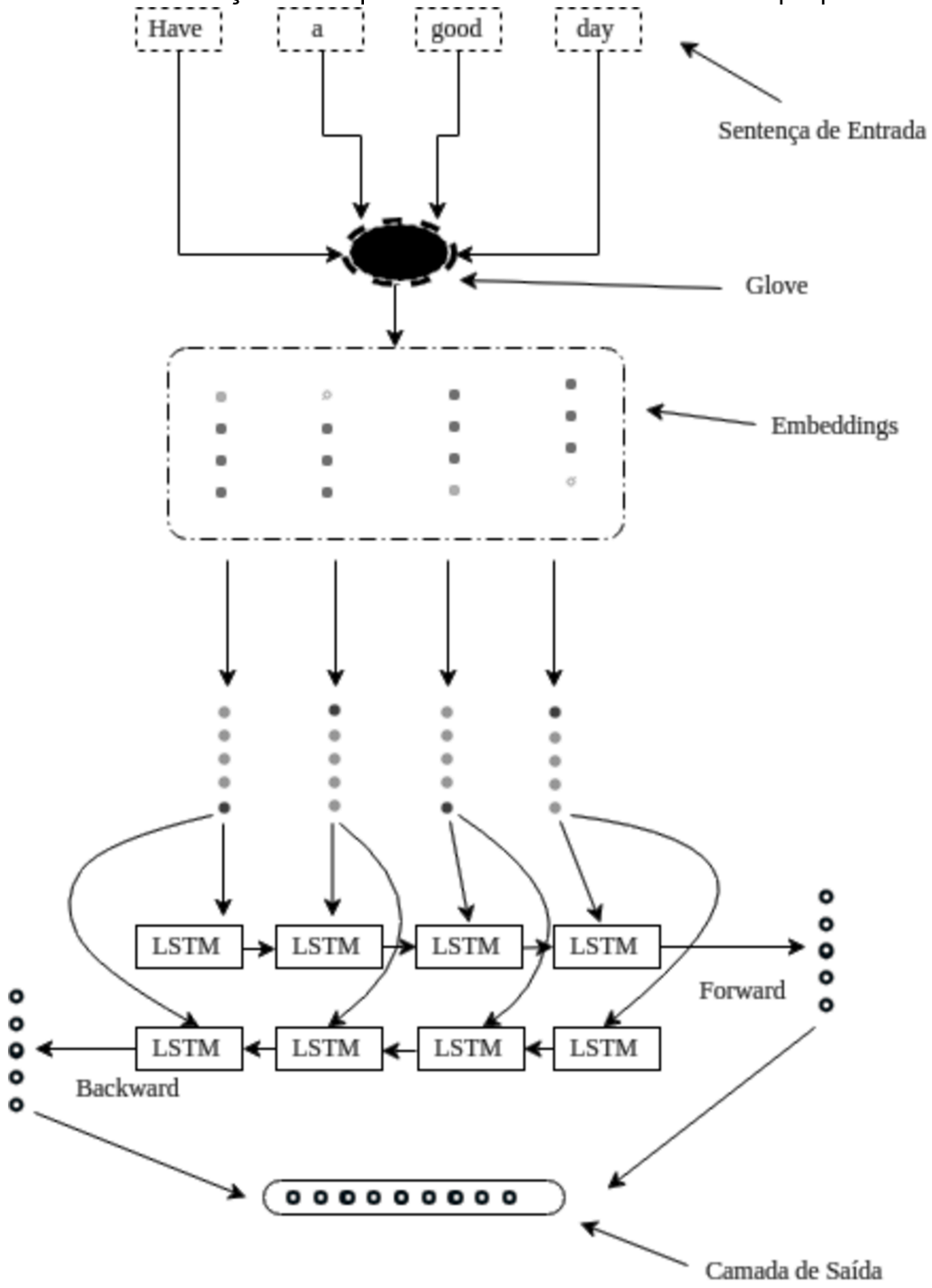
Isso significa duplicar a primeira camada recorrente na rede gerando duas redes, lado a lado, fornecendo uma sequencia de entrada real para a primeira camada e uma sequencia invertida para a segunda camada. Ao invés de carregar um peso

aleatório para cada sequencia, usa-se os pesos provenientes do GloVe. A BiLSTM é então treinada em épocas. Uma nova sequência de entrada é inserida aleatoriamente a cada época para a rede ser ajustada. Isso garante que o modelo não memorize uma única sequência e, em vez disso, possa generalizar uma solução para resolver todas as possíveis sequências de entrada para o dado problema.

Uma vez treinada, a rede será avaliada em mais uma sequência aleatória. As previsões serão então comparadas com a sequência de saída esperada para fornecer um exemplo concreto da habilidade do modelo. A execução imprime a perda e a precisão da classificação nas sequências a cada época. Isso fornece uma ideia clara da eficiência do modelo para generalizar uma solução para o problema de classificação.

A figura 10 mostra as camadas da rede através da arquitetura que retornou os melhores resultados.

Figura 10 – Uma ilustração da arquitetura de rede neural BiLSTM proposta.



Fonte: Autoria própria.

A primeira camada da rede é uma camada de *embedding* que recebe as sentenças na forma de sequência de números, e substitui cada palavra por sua representação vetorizada. Como mencionado anteriormente, os vetores usados, com tamanho 200, são resultado do treinamento de cada *corpus* com o algoritmo *GloVe*.

A camada de *embedding* recebe cada sentença na forma de índices inteiros e os codifica usando a matriz *GloVe*, passando o resultado para a próxima camada.

Cada camada LSTM contém *k* células de memória LSTM, a saída de cada camada pode ser dada em *H*. Em seguida, concatena-se $H_{forward}$ e $H_{backward}$, obtendo um vetor $\mathbf{p} \in \mathbf{R}^{2kn}$. Assim, a camada seguinte é uma camada BiLSTM com 32 elementos ocultos. Esta camada é responsável por aprender qual sequência de anotação deve ser associada à sentença de entrada, e ainda por retornar toda à sequência e não apenas o resultado da última etapa de execução. Os demais parâmetros foram deixados em seus valores padrões.

Além desses aspectos, a última camada do modelo é uma camada densa envolta por uma função de distribuição de tempo que, de acordo com o autor de Keras, François Chollet (2015), “aplica a mesma operação da camada densa (totalmente conectada) para cada intervalo de tempo de um tensor 3D”. Isso significa que em todas as saídas ao longo do tempo de execução a função de custo será calculada. A camada densa usa a função *softmax* como função de ativação para aproximar a probabilidade de cada uma das classes ontológicas.

3.4 Hiperparâmetros

Os hiperparâmetros do modelo foram definidos empiricamente. Nessa abordagem, cada hiperparâmetro foi ajustado individualmente e as melhorias foram percebidas no desempenho. Depois de concluir todas as execuções de ajuste, os valores dos hiperparâmetros foram definidos conforme tabela 3.

Tabela 3 – Hiperparâmetros utilizados no modelo.

Hiperparâmetros	Valores
Backend	Tensorflow
Word Embedding Vector Size	200
Batch Size	32
Dense Layers elements	9
Activation Function	Softmax
Loss Function	Categorical Crossentropy
Optimizer	Adam

Number of epochs	50
Metrics	F1
Maximum sequence length	50

Fonte: Autoria própria.

Além do mais, o valor definido em 32 para *Batch Size* foi com o intuito de facilitar velocidade na convergência, e, ao mesmo tempo, evitar que o modelo ficasse preso em um mínimo local. A *tangente hiperbólica* foi a função de ativação definida na camada LSTM, e na camada densa final foi empregada a função *softmax*, que é uma função de ativação comumente aplicada quando o problema é do tipo de classificação multiclasse, que é o caso deste estudo.

A *softmax* atribui probabilidades decimais a cada classe em um problema multiclasse, tais probabilidades decimais devem totalizar 1.0, gerando neste caso uma distribuição de probabilidades sobre os 9 diferentes resultados possíveis. Cada valor gerado representa a probabilidade da palavra pertencer a uma das 9 classes ontológicas.

A função de perda escolhida foi a *categorical crossentropy* equação (4), sendo esta a sugerida na literatura para um problema de classificação multiclasse. Esta função minimiza a distância entre as distribuições de probabilidade artificial, geradas pela rede, e a distribuição real (CHOLLET, 2015).

$$\mathcal{L}(y, y') = -\sum_{n} \times \sum_{i} y_i^n \, log \, y'^n_i \tag{4}$$

em que:

y *corresponde as probabilidades reais da classe.*
y' *representa a probabilidade prevista do modelo.*
i *é o índice da classe.*
n *é o índice da amostra.*

O otimizador Adam (*Adaptive Moment Estimation*), uma versão melhorada do gradiente descendente que incorpora um parâmetro para momento e taxa de aprendizado adaptativa, foi o algoritmo de otimização escolhido para o modelo de

aprendizagem profunda desenvolvido. De acordo com os autores, Kingma e Ba (2014), o Adam é um algoritmo que faz basicamente uma descida de gradiente, mas aplica médias móveis com o passo anterior e com a derivada obtida via *back propagation*, realizando atualizações mais suaves. Ele calcula as taxas de aprendizado adaptativo para cada parâmetro e também mantém uma média exponencialmente decrescente de gradientes passados, semelhante à abordagem de momento usada em outros algoritmos de otimização da descida de gradiente estocástica.

Redes neurais profundas com um grande número de parâmetros são poderosas máquinas de aprendizado. Contudo, grandes redes são lentas, o que dificulta lidar com a sobrecarga. Para resolver este problema aplica-se uma técnica chamada *dropout* (GAL, GHAHRAMANI, 2016). A ideia é descartar, aleatoriamente, unidades (junto com suas conexões) da rede neural durante o treinamento. Isso impede muitas unidades de ajuste. No momento do teste, é fácil aproximar o efeito da média das previsões de todas as redes diluídas usando, simplesmente, uma única rede não diluída com pesos menores. Isso reduz significativamente a sobrecarga e melhora outros métodos de regularização.

4 RESULTADOS

A rede LSTM foi implementada por meio do framework Keras, versão 1.2.0 (CHOLLET, 2015) que, por sua vez, foi executado na camada de alto nível do Tensorflow (ABADI et al., 2016). Keras é uma biblioteca escrita em Python que permite compilar redes neurais ao combinar camadas de diferentes dimensões e funções de ativação. O usuário pode gradualmente especificar as camadas das redes ou definir suas relações funcionais tornando o ciclo de desenvolvimento de novos modelos de aprendizado de máquina muito mais rápido. O modelo implementado foi executado em um computador com um processador Intel I7 de oitava geração e 16 GB de memória RAM, equipado com uma placa gráfica NVIDIA GeForce GTX 680. O sistema operacional que opera a máquina é o Linux Mint 18.3. Devido aos custos computacionais para executar o modelo, o Google Colaboratory[5], uma máquina virtual com GPU disponível para processamento, também foi utilizado durante a execução dos testes. Pelo Colaboratory, o Google disponibiliza para qualquer usuário que tenha uma conta Google (acesso ao google drive) uma GPU Tesla K80 para usar Keras e Tensorflow (ambos pré-instalados).

Foram realizados três tipos de testes cujas características estão detalhadas a seguir:

Teste 1: a compilação deste teste serviu para validar e finalizar a estrutura da rede BiLSTM. Apenas uma pequena parte do *corpus* foi utilizada para validar a rede neural implementada, decidir o otimizador ideal, função de perda e métricas. Neste teste, foram reconhecidas 4 classes de entidades nomeadas, as únicas presentes no trecho selecionado (*Action, Organization, Person* e *Other*).

A avaliação de desempenho para o teste 1 contou as métricas *accuracy* e *F1-score* equação (5), ambas as métricas apresentaram valores similares de precisão.

1000 sentenças foram selecionadas aplicando a divisão de 80% para treinamento e 20% para teste. O objetivo deste experimento foi verificar o desempenho do modelo implementado, avaliar e ajustar os parâmetros estabelecidos.

[5] https://colab.research.google.com

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

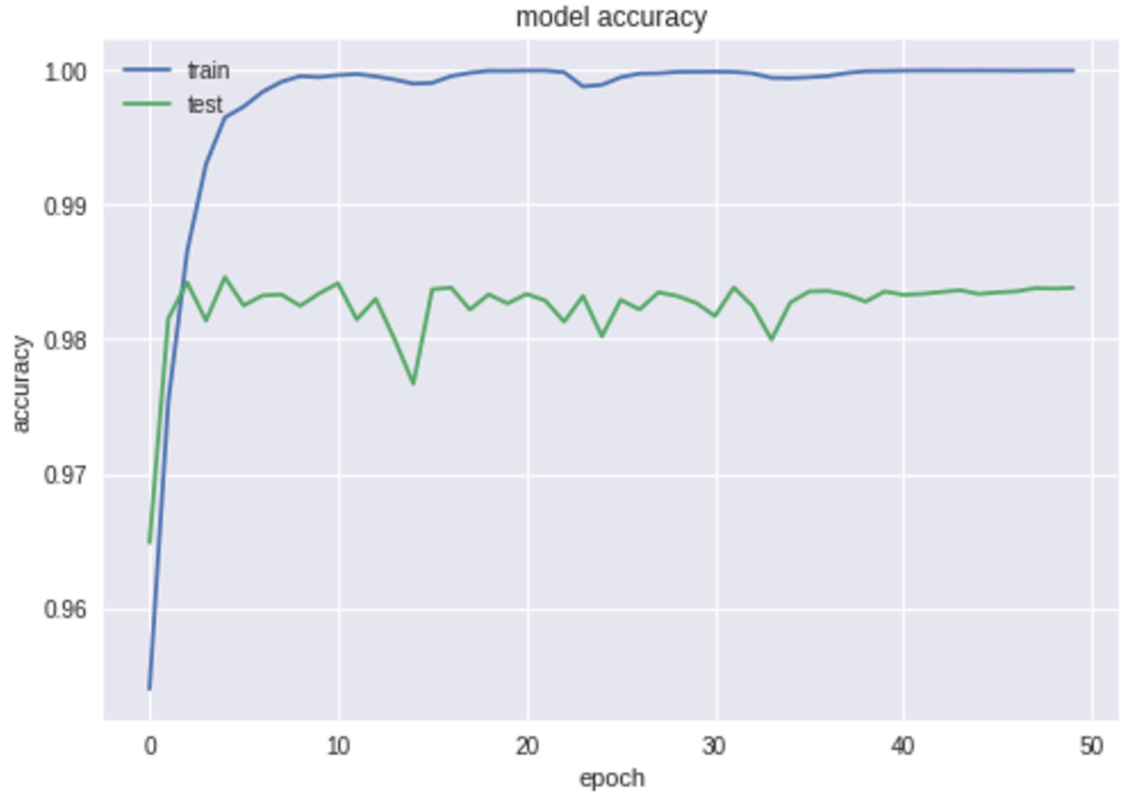
$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (5)$$

Teste 2: caracterizado pelo uso total do OANC. Mais de 3 milhões de sentenças foram consumidas pela rede BiLSTM, com emprego das 9 classes ontológicas mencionadas anteriormente como *tags*. Este teste foi a base para verificar o desempenho da rede com um maior número de dados. Com base no teste 1, a *F1-score* foi a métrica usada nesta fase. Com um número muito baixo de épocas, o algoritmo pode não convergir para um mínimo global e, por outro lado, caso o número seja muito grande corre-se o risco do algoritmo se "acostumar" com os dados de teste e apresentar uma performance ruim em dados reais. Tendo consciência dessa complexidade o número de épocas definido em 50 foi obtido por meio de testes empíricos realizados no conjunto de Testes 1. A função *fit* retorna um histórico de métricas. Com auxílio dessa função foi possível observar a precisão do algoritmo ao longo do tempo, quanto mais épocas de treino se passam, mais preciso o algoritmo se torna. A evolução do aprendizado pode ser observada na figura 11.

Tais curvas ajudam a entender até que ponto o modelo funciona bem e o quanto os resultados são úteis através da precisão histórica do algoritmo. A matriz de confusão gerada com o *corpus* OANC, levando em conta o segundo teste e selecionando o melhor resultado para atribuição das classes ontológicas, é apresentada na tabela 4.

Figura 11 – Curvas de precisão geradas pelo modelo com o *corpus* OANC.



Fonte: Autoria própria.

Pela tabela 4 é possível observar que a classe com o maior número absoluto hipotético de classificação é, como esperado, a classe *Other*, devido ao grande número de elementos que a mesma possui. É também a classe que recebe o maior número de classificações errôneas das outras classes, sendo exceção a classe *Place* que possui o maior número de classificações errôneas atribuídas à classe *Organization*, seguida por marcações da classe *Place* à classe *Person*. Tal inferência aponta que as marcações equivocadas entre *Place*, *Organization*, e *Person* são de fato esperadas, pois existem limitações dentro do contexto de busca dado o problema da ambiguidade (por exemplo, ao realizar uma busca, a entidade Lima pode conter tanto *tag* organização, quanto local, quanto pessoa).

Tabela 4 – Matriz de confusão *corpus* OANC.

	Action	Organiz.	Person	Prod.	Creat.	Place	Event	Intang.	Other
Action	**32097**	49	124	1	6	36	7	9	1047
Organiz.	6	**30241**	735	44	1	962	0	3	5225
Person	2	727	**40216**	8	2	392	0	0	3115
Prod.	0	38	25	**8456**	3	1	0	1	41
Creat.	2	1	2	2	**10771**	2	0	0	67
Place	6	1285	786	3	0	**23573**	0	1	2615
Event	12	2	0	0	0	0	**2713**	0	5
Intang.	13	3	5	1	1	1	0	**16728**	305
Other	746	8291	6955	24	6	2798	5	220	**2086321**

Fonte: Autoria própria.

Ao considerar os referidos aspectos evidencia-se que existe uma grande desvantagem no uso deste conjunto de dados, que por não ser balanceado, o aprendizado das relações relacionadas à identificação de uma determinada classe pode ser prejudicado.

A tabela 5, coletada no teste com o melhor resultado, mostra a precisão obtida em cada classe.

Cada classe ontológica foi avaliada individualmente e os valores de *Precision, Recall* e *F1-score* apresentados na tabela 5 evidenciam quão eficiente foi o anotador. O modelo apresentou uma precisão média de 98% de *F1-score*. Os resultados alcançados com o algoritmo foram significativos pois mostrou que o modelo desenvolvido na pesquisa superou o trabalho de referência. A classe que obteve o menor *F1-score* foi *Organization*, atingindo 78%. Cinco das nove classes receberam uma pontuação de *F1-score* maior que 96%.

Tabela 5 – Resultados por classe do Teste 2.

	Precision	Recall	F1-score
Action	0.98	0.96	0.97
Organization	0.74	0.81	0.78
Person	0.82	0.90	0.86
Product	0.99	0.99	0.99
Creative_Work	1.00	0.99	1.00
Place	0.85	0.83	0.84
Event	1.00	0.99	0.99
Intangible	0.99	0.98	0.98
Other	0.99	0.99	0.99
avg/ total	**0.98**	**0.98**	**0.98**

Fonte: Autoria própria.

Na tabela 6 são apresentados os resultados gerados pelo Teste 2 em conjunto com a proposta de um classificador CRF utilizado por Andrade (2018), de forma a comparar o desempenho de uma rede neural profunda com um classificador tradicional. Na comparação entre os dois modelos fica evidente a diferença de 7,6 pontos percentuais de *F1-score*, o que comprova a eficiência de modelos de aprendizado profundo em contraposição aos métodos tradicionais.

Tabela 6 – Comparação dos resultados com o trabalho de Andrade (2018).

Modelo	Precision	Recall	F1-score
CRF	91.9	91.1	91.4
Teste 2	**98.0**	**98.0**	**98.0**

Fonte: Autoria própria.

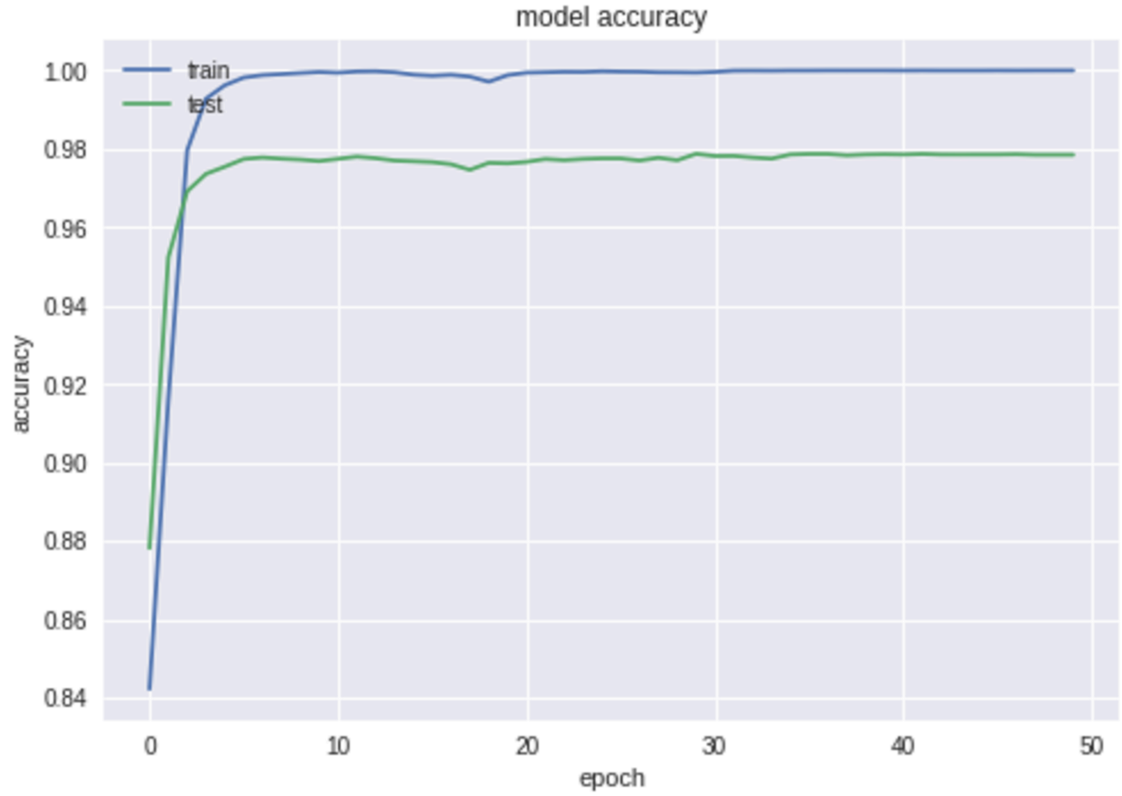
Teste 3: o Wikiner foi o *corpus* empregado no teste 3 com cerca de 150.000 sentenças. O principal objetivo deste experimento foi validar o modelo desenvolvido, comparando seus resultados com outro modelo que utilizou o mesmo *corpus*. Os resultados obtidos com este teste foram comparados com os de Rondeau e Su (2016) cujo F1-score atingiu 89,28% de precisão geral.

Como mostra a figura 12, neste teste os conjuntos de dados de treinamento e teste exibiram curvas de precisão similares. O terceiro teste é resultado de uma

reformulação do segundo com aplicação de um novo conjunto de dados para um bloco reduzido de classes ontológicas.

Dada a validação do modelo proposto, o Wikiner também foi escolhido por ter sido utilizado no trabalho de Mendonça Junior et al. (2016). Apesar dos autores realizaram um experimento com a parte escrita em Português do *corpus*, os dados foram válidos para comparação diante da proposta de utilizar uma rede neural LSTM com *word embeddings*.

Figura 12 – As curvas de precisão do modelo com o *corpus* Wikiner.



Fonte: Autoria própria.

Além do *score*, a performance do modelo também pode ser visualizada na matriz de confusão. A proporção de verdadeiros positivos mostra a capacidade do sistema em predizer corretamente a condição para casos que realmente a têm. De acordo com a tabela 7, a classe *Misc* apresentou o maior número de falsos negativos devido à variedade de entidades envolvidas nesta classe.

Tabela 7 – Matriz de confusão do *corpus* Wikiner.

	Organization	Place	Misc	Person	Other
Organization	**5800**	13	12	10	450
Place	130	**4880**	80	30	170
Misc	20	40	**3250**	80	790
Person	60	20	10	**4390**	310
Other	60	59	150	60	**102690**

Fonte: Autoria própria.

Na tabela 8 são apresentados os resultados do teste realizado e pode-se concluir que a rede BiLSTM gerou resultados melhores que os obtidos pelos autores mencionados anteriormente. Quatro das cinco classes avaliadas apresentaram resultados superiores a 89%, dispondo de 95% *F1-score* geral.

Tabela 8 – Resultados por classe Teste 3.

	Precision	Recall	F1-score
Organization	0.91	0.87	0.90
Place	0.95	0.92	0.95
Misc	0.87	0.78	0.84
Person	0.96	0.90	0.93
Other	0.98	1.00	0.99
avg/ total	**0.97**	**0.93**	**0.95**

Fonte: Autoria própria.

Tabela 9 – Comparação dos resultados com o trabalho de Mendonça Junior et al. (2016).

Modelo	Precision	Recall	F1-score
ParamopamaWNN	86.45	89.77	88.08
Teste 3	**97.0**	**93.00**	**95.00**

Fonte: Autoria própria.

Na tabela 9 são mostrados os resultados do Teste 3 em comparação com o cenário de experimento intitulado ParamopamaWNN realizado por Mendonça Junior et al. (2016). O teste 3 do presente trabalho obteve cerca de 7 pontos percentuais sobre o ParamopamaWNN, o que sugere que o modelo com BiLSTM alcançou melhor desempenho.

5 CONCLUSÃO

O capítulo anterior apresentou os resultados obtidos na fase de testes do modelo. Para verificar a viabilidade do uso em sistemas de anotação os tempos de execução foram medidos. O algoritmo leva em média 30 horas para executar 50 épocas com o *corpus* OANC e cerca 6 horas com o Wikiner em computador com placa gráfica K80 da NVIDIA. Treinamentos com redes neurais consomem tempo, mas os tempos apresentados estão dentro do esperado.

O objetivo desta dissertação foi verificar se o uso de RNNs seria capaz de equiparar as ferramentas do estado da arte no campo da anotação semântica. Em particular, concentrou-se na aplicação da ontologia Schema.org em um modelo baseado na rede neural LSTM bidirecional. A abordagem exploratória deste tema permitiu perceber que técnicas de aprendizado profundo apresentam melhores resultados para modelos que demandam grande tempo na fase de engenharia de requisitos.

Como os recursos computacionais existentes são escaláveis, o processo de anotação é facilitado, e isso significa redução de custos e otimização da forma de trabalhar, seja pela utilização do processamento paralelo em CPU's e GPU's ou mesmo através do processamento em ambientes *online*. Apesar dessa facilidade, é claro que existe todo um trabalho durante o desenvolvimento e a configuração de um modelo experimental que pode ser moroso, quer pelas dificuldades encontradas, complexidade de implementação ou mesmo pelo ajuste dos dados e hiperparâmetros do modelo.

Para a tarefa de avaliação da anotação, foi necessário verificar diferentes formas de avaliação deste tipo de sistema presentes na literatura. Com base nesse levantamento, além de ser um tipo de avaliação empregado em trabalhos similares a este que serviu de parâmetro de comparação, a métrica *F1-score* foi escolhida para apresentar a performance do modelo criado. Os resultados obtidos nos Testes 2 e 3, que propõe rotular *tokens* do OANC e do Wikiner de acordo com as categorias da ontologia Schema.org, foram melhores que dois trabalhos realizados anteriormente com o mesmo cenário de experimentação, com um total geral de *F1- score* em 98% e 97% em cada teste.

A principal contribuição deste trabalho se dá na obtenção de bons resultados quanto a atribuição de rótulos de categorias ontológicas a itens lexicais por meio de

um modelo de rede neural LSTM bidirecional bastante simples. Além disso, foi mostrada a importância de usar vetores de *word embeddings* sobre grandes quantidades de dados, pois assim é possível obter uma boa extração automática de diversas características para o aprendizado da rede. A anotação semântica proposta neste trabalho conseguiu atingir resultados competitivos dado que o algoritmo funciona com uma estrutura básica, levando em consideração outros modelos que adotam arquiteturas híbridas, utilizam *embeddings* em nível de caractere ou mesmo trabalham com extração de *features*.

Devido à hipótese de que a anotação semântica ontológica não depende tanto de características no nível de caractere, mas de características relacionadas à coocorrência de palavras, *embeddings* em nível de caractere não foram manipulados. Os resultados corroboram a tese de que modelos de redes neurais recorrentes, especialmente as variações de LSTM e GRU, são a escolha atual para o processamento de sequências no âmbito do PLN.

Uma característica do modelo implementado é a independência do conjunto de etiquetas de entidade nomeada. Isso permite que ele seja utilizado em futuras pesquisas com diferentes *corpora* e conjunto de etiquetas. Para que isso aconteça, basta manipular os dados a serem inseridos tornando-os compatíveis com o formato de entrada aceitável pelo algoritmo.

Como proposta de trabalho futuro, pode-se investigar o uso de redes neurais profundas para anotação semântica ontológica com aprendizado não supervisionado. Uma abordagem direta seria utilizar redes neurais profundas para prever entradas futuras para dados sequenciais.

REFERÊNCIAS

ABADI, Martin *et al*. Tensorflow: A system for large-scale machine learning. **USENIX Symposium on Operating Systems Design and Implementation**, n. 12, p. 265-283, 2016.

ALPAYDIN, Ethem. **Introduction to machine learning**. The MIT Press, 2014.

ALUÍSIO, Sandra Maria; ALMEIDA, Gladis Maria de Barcellos. O que é e como se constrói um corpus? Lições aprendidas na compilação de vários corpora para pesquisa linguística. **Calidoscópio**, v. 4, n. 3, p. 156-178, 2006.

ANDRADE, Guidson Coelho. **Semantic enrichment of American English corpora through automatic semantic annotation based on top-level ontologies using the crf classification model**. 2018. Tese de Mestrado (Mestrado em Ciência da Computação) - Universidade Federal de Viçosa, 2018.

BARRETO, Jorge Muniz. **Inteligência artificial no limiar do século XXI**. 3. ed. Florianópolis: Duplic, 2001. ISBN 9788590038252.

BIBER, Douglas. Representativeness in corpus design. **Literary and linguistic computing**, v. 8, n. 4, p. 243-257, 1993.

BISHOP, Christopher M. **Pattern recognition and machine learning**. Springer Science+ Business Media, 2006.

BISHOP, Christopher M. **Neural networks for pattern recognition**. Oxford University Press, 1995.

BYEON, Wonmin; BREUEL, Thomas M.; RAUE, Federico; LIWICKI, Marcus. Scene labeling with lstm recurrent neural networks. **Proceedings of the IEEE ConferenceComputer Vision and Pattern Recognition**, p. 3547-3555, 2015.

CARRERAS, Xavier; MÀRQUEZ, Luís. Introduction to the CoNLL-2005 shared task: Semantic role labeling. **Proceedings of the ninth conference on computational natural language learning** : (CoNLL-2005), p. 152-164, 2005.

CARVALHO, Wesley Seidel. **Reconhecimento de entidades mencionadas em português utilizando aprendizado de máquina**. 2012. Tese de Mestrado (Mestrado em Ciência da Computação) - Universidade de São Paulo, 2012.

CHIU, Jason P. C.; NICHOLS, Eric. Named entity recognition with bidirectional LSTM-CNNs. **Transactions of the Association for Computational Linguistics** , n. 4, p. 357-370, 2016.
CHOLLET, François. **Keras.** 2015.

CISCO GLOBAL , Cloud Index. Cisco global cloud index: Forecast and methodology. *In*: **Cisco global cloud index: Forecast and methodology**. 2016-

2021. Disponível em: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/white-paper-c11-738085.html. Acesso em: 19 mar. 2019.

CRUSE, Alan. **Meaning in language: An introduction to semantics and pragmatics**. Oxford University Press UK, 2011.

CRUSE, Alan. **Lexical semantics**. Cambridge University Press, 1986.

DUDA, Richard O.; HART, Peter E.; STORK, David G. **Pattern classification**. 2. ed. John Wiley & Sons, 2001.

ELMAN, Jeffrey L. Finding structure in time. **Cognitive science**, v. 14, n. 2, p. 179-211, 1990.

ERB, Randall J. Introduction to backpropagation neural network computation. **Pharmaceutical research**, v. 10, n. 2, p. 165-170, 1993.

FILLMORE, Charles J.; IDE, Nancy; JURAFSKY, Daniel; MACLEOD, Catherine. An American national corpus: A Proposal. **Proceedings of the First Annual Conference on Language Resources and Evaluation**, p. 965-969, 1998.

FILLMORE, Charles J. Frame semantics. *In*: COGNITIVE linguistics: Basic readings. Berlin: Mouton de Gruyter, 2006. cap. 10, p. 373-400. ISBN 9783110190847.

FIRTH, John R. A synopsis of linguistic theory 1930-55. **Studies in linguistic analysis**, 1957.

FRIEDMAN, Jerome; HASTIE, Trevor; TIBSHIRANI, Robert. **The elements of statistical learning**. 10. ed. New York: Springer series in statistics, 2001. v. 1.

GAL, Yarin; GHAHRAMANI, Zoubin. A theoretically grounded application of dropout in recurrent neural networks. **Advances in neural information processing systems**. 1019-1027, 2016.

GANGEMI, Aldo; GUARINO, Nicola; MASOLO, Claudio; OLTRAMARI, Alessandro; SCHNEIDER, Luc. Sweetening ontologies with DOLCE. **International Conference on Knowledge Engineering and Knowledge Management**, Berlin, p. 166-181, 2002.

GEERAERTS, Dirk. **Theories of lexical semantics**. Oxford University Press, 2010.

GERS, Felix A.; SCHMIDHUBER, Jürgen; CUMMINS, Fred. Learning to forget: Continual prediction with LSTM. **Int. Conf. on Articial Neural Networks**: Proc. ICANN‘99, London, v. 2, p. 850-855, 1999.

GOLDBERG, Adele E. Constructions: A new theoretical approach to language. **Trends in cognitive sciences**, v. 7, n. 5, p. 219-224, 2003.

GOLDBERG, Yoav. Neural network methods for natural language processing. **Synthesis Lectures on Human Language Technologies**, v. 10, n. 1, p. 1-309, 2017.

GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. **Deep learning**. Cambridge: MIT press, 2016.

GRAVES, Alex; LIWICKI, Marcus; FERNÁNDEZ, Santiago; BERTOLAMI, Roman; BUNKE, Horst; SCHMIDHUBER, Jürgen. A novel connectionist system for unconstrained handwriting recognition. **IEEE transactions on pattern analysis and machine intelligence**,v. 31, n. 5, p. 855-868, 2008.

GRAVES, Alex; SCHMIDHUBER, Jürgen. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. **Neural networks**, v. 18, n. 5-6, p. 602-610, 2005.

GRIES, Stefan. Polysemy: The notion of polysemy. *In*: COGNITIVE Linguistics. Walter de Gruyter GmbH, 2019. cap. 2, p. 23-43.

GUARINO, Nicola. Some organizing principles for a unified top-level ontology. **AAAI Spring Symposium on Ontological Engineering**, p. 57-63, 1997.

GUARINO, Nicola. Formal ontology in information systems. **Proceedings of the first international conference**: FOIS'98, Trento, Italy, v. 46, 1998.

GUHA, Ramanathan V.; BRICKLEY, Dan; MACBETH, Steve. Schema. org: evolution of structured data on the web.. **Communications of the ACM**, v. 54, n. 2, p. 44-51, 2016.

HANDSCHUH, Siegfried; STAAB, Steffen. **Annotation for the semantic web**. Amsterdam: IOS press, 2003. v. 96.

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

HOCHREITER, Seep; SCHMIDHUBER, Jürgen. Long short-term memory. **Neural computation**, v. 9, n. 8, p. 1735-1780, 1997.

HORNIK, Kurt; STINCHCOMBE, Maxwell; WHITE, Halbert. Multilayer feedforward networks are universal approximators. **Neural networks**, v. 2, n. 5, p. 359-366, 1989.

GARETH, James; WITTEN, Daniela; HASTIE, Trevor; TIBSHIRANI, Robert. **Neural networks**: An introduction to statistical learning. New York: Springer, 2013. v. 112.

JURAFSKY, Dan. **Speech & language processing**. Pearson Education India, 2000.

JURAFSKY, Daniel; MARTIN, James H. **Speech and Language Processing**: An introduction to Natural Language Processing, Computational Linguistics and Speech Recognition. 3. ed. New Jersey: Prentice Hall, 2019. Disponível em: https://web.stanford.edu/~jurafsky/slp3/ed3book.pdf. Acesso em: 22 out. 2019.

KINGMA, Diederik P.; BA, Jimmy. Adam: A method for stochastic optimization. **ArXiv preprint arXiv**: 1412.6980, 2014.

KIRYAKOV, Atanas; TERZIEV, Ivan; MANOV, Dimitar; OGNYANOFF, Damyan. Semantic annotation, indexing, and retrieval. **Journal of Web Semantics**, v. 2, n. 1, p. 49-79, 2004.

KOFFKA, Kurt. **Principles of Gestalt psychology**. Routledge, 2013.

KOTSIANTIS, Sotiris B.; ZAHARAKIS, I.; PINTELAS, P. Supervised machine learning: A review of classification techniques. **Emerging artificial intelligence applications in computer engineering**, v. 160, p. 3-24, 2007.

LEECH, Geoffrey. Introducing corpus annotation. *In*: CORPUS Annotation: Linguistic Information from Computer Text Corpora. 1. ed. London: Routledge, 1997. cap. 1, p. 11-28.

LENAT, Douglas B.; GUHA, Ramanathan V.; PITTMAN, Karen; PRATT, Dexter; SHEPHERD, Mary. Cyc: toward programs with common sense. **Communications of the ACM**, v. 33, n. 8, p. 30-49, 1990.

LEWIS, Michael; GOUGH, Cherry; MARTÍNEZ, Ron; POWELL, Mark; MARKS, Jonathan; WOOLARD, George C.; RIBISCH, Kar Heinz. **Implementing the Lexical Approach**: Putting Theory into Practice. Hove: Language Teaching Publications, 1997. v. 3. ISBN 1899396608.

MAEDCHE, Alexander; STAAB, Steffen. Ontology learning for the semantic web. **IEEE Intelligent systems**, v. 16, n. 2, p. 72-79, 2001.

MANNING, Christopher D; SCHÜTZE, Hinrich. **Foundations of statistical natural language processing**. Cambridge: MIT press, 1999.

MITCHELL, Marcus; SANTORINI, Beatrice; MARCINKIEWICZ, Mary Ann. Building a Large Annotated Corpus of English:: The Penn Treebank. **Penn Engineering Technical Reports (CIS)**, Pensilvania, 1993.
MATTHEWS, Peter Hugoe. **The concise Oxford dictionary of linguistics**. 3. ed. United Kingdom: Oxford University Press, 2014.

MCCULLOCH, Warren S.; PITTS, Walter. A logical calculus of the ideas immanent in nervous activity. **Bulletin of mathematical biophysics**, v. 5, n. 4, p. 115-133, 1943.

MEHROTRA, Kishan; MOHAN, Chilukuri K.; RANKA, Sanjay. **Elements of artificial neural networks**. 2. ed. Cambridge: MIT press, 2000.

MENDONÇA JUNIOR, Carlos A.; BARBOSA, Luciano A.; MACEDO, Hendrik T. Uma arquitetura híbrida LSTM-CNN para reconhecimento de entidades nomeadas em textos naturais em língua portuguesa. **Encontro Nacional de Inteligência Artificial e Computacional**, Recife, ed. XIII, p. 241-252, 2016.

MITKOV, Ruslan. **The Oxford handbook of computational linguistics**. New York: Oxford University Press, 2004.

MONAGHAN, Frank. Judging a word by the company it keeps: The use of concordancing software to explore aspects of the mathematics register. **Language and Education**, v. 13, n. 1, p. 59-70, 1999.

MOREIRA, Alexandra; ALVARENGA, Lidia; OLIVEIRA, Alcione de Paiva. O nível do conhecimento e os instrumentos de representação: tesauros e ontologias. **DataGramaZero-Revista de Ciência da Informação**, v. 5, n. 6, p. 1-25, 2004a.

MOREIRA, Alexandra; ALVARENGA, Lidia; OLIVEIRA, Alcione de Paiva. Thesaurus and Ontology: A Study of the Definitions Found in the Computer and Information Science Literature, by Means of an Analytical Synthetic Method. **Knowledge Organization**, v. 31, n. 4, p. 231-244, 2004b.

MÀRQUEZ, Lluís; CARRERAS, Xavier; LITKOWSKI,, Kenneth C.; STEVENSON, Suzane. Semantic role labeling: an introduction to the special issue. **Computational Linguistics**, v. 34, n. 2, p. 145-159, 2008.

NILES, Ian; PEASE, Adam. Semantic role labeling: Towards a standard upper ontology. **Formal Ontology in Information Systems**: Proceedings of the International Conference, v. 2001, p. 2-9, 2001.

NORVING, Peter; LAKOFF, George. Taking: A study in lexical network theory. **Annual Meeting of the Berkeley Linguistics Society**, v. 13, p. 195-206, 1987.
NOTHMAN, Joel; RINGLND, Nicky; RADFORD, Will; MURPHY, Tara; CURRAN, James. Taking: Learning multilingual named entity recognition from Wikipedia. **Artificial Intelligence**, Australia, v. 194, p. 151-175, 2013

OLIVEIRA, Alcione de Paiva. Parâmetros das redes neurais. *In*: **Parâmetros das redes neurais**. Universidade Federal de Viçosa, 2018. Slides.

PENNINGTON, Jeffrey; SOCHER, Richard; MANNING, Christopher. Glove: Global vectors for word representation. **Empirical Methods in Natural Language Processing**: (EMNLP), Stanford, p. 1532-1543, 2014.

PUSTEJOVSKY, James. The generative lexicon. **Computational linguistics**, v. 17, n. 4, p. 409-441, 1991.

PUSTEJOVSKY, James; STUBBS, Amber. **Natural Language Annotation for Machine Learning**: A guide to corpus-building for applications. 1. ed. Cambridge: O'Reilly Media, Inc, 2012.

RAJASEKARAN, Sanguthevar; PAI, GA Vijayalakshmi. **Neural networks, fuzzy logic and genetic algorithm**: synthesis and applications. Coimbatore: PHI Learning Pvt, 2003.

RAVIN, Yael; LEACOCK, Claudia. **Polysemy:**: Theoretical and computational approaches. New York: Oxford University Press, 2000.

REEVE, Lawrence; HAN, Hyoil. Survey of semantic annotation platforms. **Symposium on Applied computing**: Proceedings of the 2005 ACM, p. 1634-1638, 2005.

RONDEAU, Marc-Antoine; SU, Yi. LSTM-Based NeuroCRFs for Named Entity Recognition. **Interspeech**, San Francisco, p. 665-669, 2016.

RUMELHART, David E.; HINTON, Geoffrey E.; WILLIAMS, Ronald J. Learning internal representations by error propagation. **California Univ San Diego La Jolla Inst for Cognitive Science**, n. ICS-8506, 1985.

RUPPENHOFER, Josef; ELLSWORTH, Michael; SCHWARZER-PETRUCK, Myriam; JOHNSON, Christopher R.; SCHEFFCZYK, Jan. **FrameNet II**: Extended theory and practice. Berkeley: International Computer Science Institute, 2006. Disponível em: https://www.researchgate.net/publication/242582900_FrameNet_II_Extended_Theory_and_Practice. Acesso em: 16 out. 2018.

RUSSELL, Stuart; NORVING, Peter. **Artificial intelligence**: a modern approach. Malaysia: Pearson Education Limited, 2016.

SANTOS, Alcione Miranda dos; SEIXAS, José Manoel de; PEREIRA, Basílio de Bragança; MEDRONHO, Roberto de Andrade. Usando redes neurais artificiais e regressão logística na predição da hepatite A. **Revista Brasileira de Epidemiologia**, v. 8, n. 2, p. 117-126, 2005.

SARDINHA, Tony Berber. **Linguística de corpus**. Editora Manole, 2004. ISBN 9788520416761.

SCHUSTER, Mike; PALIWAL, Kuldip K. Bidirectional recurrent neural networks. **IEEE transactions on Signal Processing**, v. 45, n. 11, p. 2673-2681, 1997.

SERVAN-SCHREIBER,, David; CLEEREMANS,, Axel; MCCLELLAND, James L. Graded state machines: The representation of temporal contingencies in simple recurrent networks. **Machine Learning**, v. 7, n. 2-3, p. 161-193, 1991. DOI https://doi.org/10.1007/BF00114843. Disponível em: https://link.springer.com/article/10.1007/BF00114843. Acesso em: 6 fev. 2019.

SINCLAIR, John. **Trust the Text**: Language, Corpus and Discourse. 1. ed. London: Routledge, 2004. 224 p. ISBN 9780203594070.

SINGH, Yashpal; CHAUHAN, Alok Singh. Neural Networks in Data Mining. **Journal of Theoretical & Applied Information Technology**, v. 5, n. 1, 2009.

SRIVASTAVA, Nitish; HINTON, Geoffrey; KRIZHEVSKY, Alex; SUTSKEVER, Ilya; SALAKHUTDINOV, Ruslan. Dropout: a simple way to prevent neural networks from overfitting. **The journal of machine learning research**, v. 15, n. 1, p. 1929-1958, 2014.

SUKHAREVA, Maria; CHIARCOS, Christian. An ontology-based approach to automatic part-of-speech tagging using heterogeneously annotated corpora. **Proceedings of the Second Workshop Natural Language Processing and Linked Open Data**, Hissar, p. 23-32, 2015.

SUTSKEVER, Ilya; VINYALS, Oriol; LE, Quoc V. Sequence to sequence learning with neural networks. **Advances in neural information processing systems**, p. 3104-3112, 2014.

SUTTON, Richard S.; BARTO, Andrew G. **Reinforcement learning**: An introduction. 2. ed. Cambridge: MIT press, 2018.

VAPNIK, Vladimir. **The nature of statistical learning theory**. 2. ed. atual. New York: Springer, 2013. ISBN 9781441931603.

VIEIRA, Renata; LIMA, Vera L. Linguística computacional: princípios e aplicações. **Jornada de Atualização em Inteligência Artificial**: XXI Congresso da SBC, v. 3, p. 47-86, 2001.