AUTOMAÇÃO DO PREENCHIMENTO DE UM GRAFO DE CONHECIMENTO APLICADO A ARTIGOS DE UM PORTAL DE NOTÍCIAS ESPORTIVAS

André Fernandes Bispo[1], Jackson Gomes de Souza[2]

[1] Bacharel em Engenharia de Software pelo Centro Universitário Luterano de Palmas – CEULP/ULBRA. E-mail: andre1@rede.ulbra.br.
[2] Mestre em Engenharia da Computação pela UFRN. Professor dos cursos de Ciência da Computação, Sistemas de Informação e Engenharia de Software do CEULP/ULBRA. E-mail: jackson.souza@ulbra.br.

RESUMO

Um Grafo de Conhecimento é uma base de conhecimento que utiliza nós e arestas ou triplas para organizar as informações. Para o preenchimento deste grafo foi escolhido um portal de notícias esportivas como fonte de alimentação, os dados das notícias publicadas neste portal foram extraídos, analisados pelo Processamento de Linguagem Natural que teve por objetivo analisar a notícia, tokenizando e etiquetando cada um dos elementos do texto de acordo com suas classes gramaticais, permitindo a identificação das entidades ou elementos mais importantes, então estes dados processados por um algoritmo desenvolvido na linguagem Python com o objetivo de encontrar as entidades da notícia e informações pertinentes ao trabalho. Por fim, armazenou-se os dados processados no Grafo de Conhecimento de maneira automatizada.

PALAVRAS-CHAVE: Grafo de Conhecimento; Processamento de Linguagem Natural; Automação.

1 INTRODUÇÃO

A leitura e interpretação de texto pelas máquinas é chamada de Processamento de Linguagem Natural (Natural Language Processing, em inglês, ou NLP). Segundo Silva (2008), o NLP, de forma geral, constitui a matriz das tecnologias linguísticas e um dos novos paradigmas da Língua e da Linguística. Assim, o NLP identifica as palavras presentes numa frase e realiza processos de análise que permitem extrair informações e identificar o contexto, dando um significado a cada palavra conforme o domínio. O NLP é responsável por analisar as estruturas de uma linguagem natural, como a semântica, léxica e sintaxe, sendo ela a responsável por analisar e identificar o contexto que as palavras têm em um texto, resolvendo problemas de ambiguidades, por exemplo.

O Processamento de Linguagem Natural é uma disciplina que se utiliza de conhecimentos sobre a linguagem natural humana, bem como sua comunicação, valendo-se destes preceitos para que haja uma comunicação com sistemas operacionais, bem como, uma maneira de facilitar a comunicação entre seres humanos (SANTOS, 2001, p. 229). De acordo com Vieira e Lopes (2010, p. 184), o “Processamento de Linguagem Natural é uma área da Ciência da Computação que estuda o desenvolvimento de programas de computador que analisam, reconhecem e/ou geram textos em linguagens humanas, ou linguagens naturais”. Referindo-se à linguística computacional, Vieira e Lima (2001, p. 1) proferem que é “a área de conhecimento que explora as relações entre linguística e informática, tornando possível a construção de sistemas com capacidade de reconhecer e produzir informação apresentada em linguagem natural”.

Alinhado ao NLP, está o Grafo de Conhecimento (Knowledge Graph, em inglês, ou KG) que segundo Paulheim (2016), (i) descreve as entidades do mundo real e suas relações, organizadas em um grafo, (ii) define também possíveis classes e relações de entidades em um esquema. Ainda sobre o Grafo de Conhecimento, Pan et. al (2017) descrevem que suas características primárias estão no arranjo estrutural da representação de conhecimento, nos processos de coordenação das informações e nos algoritmos de busca.

Para o preenchimento do KG, é necessário que se faça a extração dos dados que podem ser frases, textos ou conceitos, de uma determinada fonte de dados, sejam eles sites, artigos, entre outros. Destes dados, deverá ser feita a extração de entidades que são as principais palavras ou expressões de um texto.

A extração dos dados do trabalho foi obtida por meio de um portal de notícias esportivas que é organizado nas seguintes áreas de conteúdo: página inicial; página ou blog de cada área de conteúdo, que mostra as notícias da área em questão; página para cada notícia ou artigo, que apresenta título, data da publicação, conteúdo e notícias relacionadas. Além disso, também foram utilizadas informações extraídas de portais auxiliares que contribuíram para que a taxa de assertividade do trabalho atingisse níveis satisfatórios.

Em vista disso, este trabalho originou-se do problema de automatizar o preenchimento de um Grafo de Conhecimento através do Processamento de Linguagem Natural em um determinado artigo de notícia esportiva. Para realizar este preenchimento, foi necessário extrair um artigo de um portal de notícias, aplicar o Processamento de Linguagem Natural, identificando os pontos desejados e armazená-los no Grafo de Conhecimento.

Para alcançar o objetivo deste trabalho foi necessário selecionar os portais onde os dados seriam extraídos; foi então, realizado os mapeamentos destes portais, a fim de extrair somente os dados de interesse, eliminando assim, a extração de dados indesejados, como anúncios ou textos que não compreendem a notícia. Após o mapeamento, foi utilizada a ferramenta de extração Scrapy com a linguagem de programação Python; foi utilizada a ferramenta Linguakit para realizar a análise do artigo extraído e identificou as entidades presentes no mesmo; foi implementado o algoritmo responsável por aplicar as regras que visam buscar as entidades, padrões e/ou informações pertinentes para o trabalho no artigo analisado, sendo este algoritmo desenvolvido em Python. E, por fim, inseriu-se as entidades e outras informações identificadas no artigo no grafo de conhecimento.

2 MATERIAL E MÉTODOS

Para o desenvolvimento deste trabalho, foram utilizados os seguintes materiais: a linguagem de programação Python, Linguakit e Neo4j.

O Python, segundo Guido van Rossum (1996), criador da linguagem, é uma linguagem de programação orientada a objetos desenvolvida em 1992, com enfoque na legibilidade, possuindo apenas uma possibilidade de endentação e reduzindo ao máximo as maneiras de codificação de um código específico.

O Linguakit, de acordo com o site oficial (2020), é capaz de explorar, analisar e obter informações de textos e documentos escritos, contendo entre outras ferramentas linguísticas, módulos de conjugação e tradução linguística; identificador morfossintático e analisador sintático; analisador de sentimentos e extrator de palavras-chaves.

O Neo4j, de acordo com o site oficial (2020), é um banco de dados desenvolvido para analisar os dados e seus relacionamentos, conectando os dados à medida que são armazenados, permitindo diversas formas de consultas de maneira rápida e prática. O banco de dados do Neo4j opera com o conceito de grafo nativo, possuindo uma estrutura flexível definida por relacionamento armazenados entre registros de dados. Cada vértice do grafo armazena os dados das entidades, aos quais estão ligadas através de arestas.

A delineação do trabalho ocorreu de acordo com o infográfico da Figura 1.

Figura 1. Infográfico dos processos de desenvolvimento do trabalho.

A contextualização é um ponto delicado para o desenvolvimento do trabalho, visto que ela permite um direcionamento e embasamento técnico e teórico para o trabalho como um todo. Para garantir esse embasamento, foi necessária uma fundamentação em artigos e livros que testaram os conceitos e ferramentas que foram utilizados. Livros, dissertações (mestrado ou doutorado) e artigos científicos serviram de base para a elaboração da contextualização do trabalho. Feito a contextualização do trabalho, inicia-se o desenvolvimento das funcionalidades, como apresentado na Figura 2.

Figura 2. Fluxo da Metodologia.

Na fase de implementação, foi realizado o desenvolvimento do trabalho. Primeiramente, foram codificados os extratores dos dados dos portais. Para realizar estas extrações, foi utilizado o framework Scrapy do Python, visto que possui inúmeras funcionalidades que permitem a extração adequada dos dados desejados, armazenando cada uma das extrações em arquivos distintos.

Em seguida, é realizada a configuração do Linguakit. No Linguakit, foi feita a seleção dos módulos que permitiram o desenvolvimento do trabalho, sendo feita a análise do artigo extraído na etapa anterior, onde nesta análise foi feito o Processamento de Linguagem Natural, identificando e nomeando as entidades presentes no texto.

Após a análise, foi realizado o processamento das informações extraídas pelo Linguakit, onde neste processamento, são identificadas as entidades e informações complementares no artigo, sendo organizadas para que possam ser inseridas no Grafo de Conhecimento.

Após o processamento dos dados, foi utilizado um algoritmo disponibilizado pela documentação do Neo4j, para realizar o armazenamento destes dados de maneira remota no Neo4j que utiliza o KG como estrutura para a disposição dos dados.

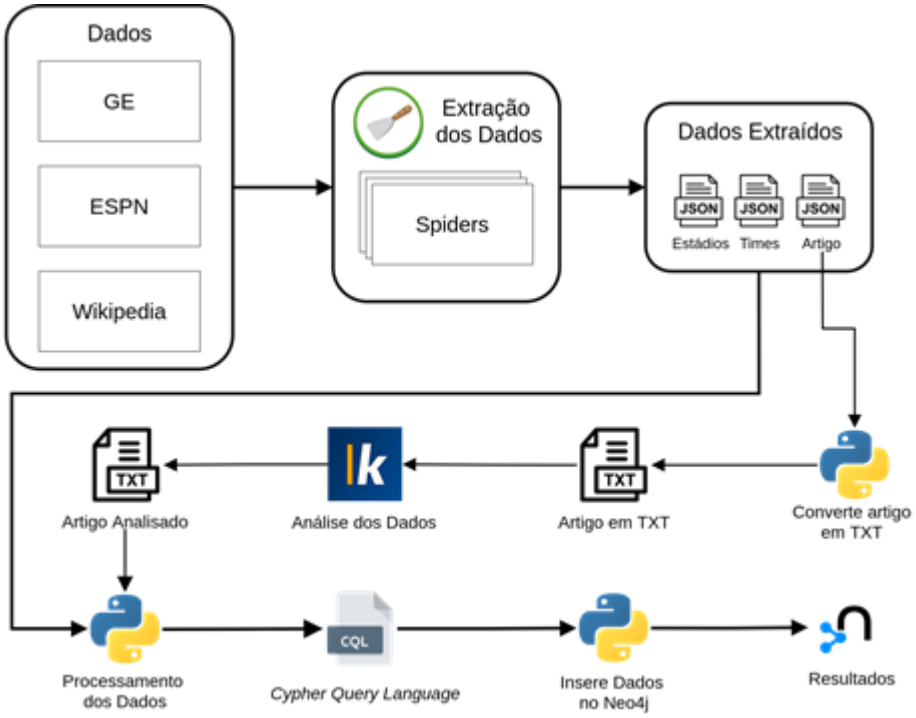
3 RESULTADOS E DISCUSSÃO

Nesta seção é apresentada a visão geral da ferramenta produzida neste trabalho, bem como o detalhamento dos passos seguidos para a conclusão da mesma. A Figura 3 apresenta a arquitetura da ferramenta construída neste trabalho.

Para o funcionamento da ferramenta, inicialmente é feita seleção das fontes dos dados que serão utilizados, sendo o portal do Globo Esporte 9 (GE) a principal fonte destes dados, visto que, corresponde ao endereço onde os artigos serão extraídos. Outra fonte corresponde ao portal *da Entertainment and Sports Programming Network* (ESPN), onde são extraídas as listas dos times de futebol. A última fonte corresponde ao portal do Wikipédia, onde são extraídas informações sobre os estádios de futebol.

Selecionadas as fontes, é realizada a extração de seus dados utilizando a biblioteca Scrapy que os armazena em arquivos JavaScript Object Notation (JSON). Em seguida, é feita a análise do artigo por meio do Linguakit, logo após, é feito o processamento das informações analisadas, com o apoio dos arquivos contendo os dados sobre os times e estádios. Por fim, as informações obtidas pela análise são inseridas no Neo4j gerando o grafo desejado.

Figura 3. Arquitetura da ferramenta



O primeiro passo para o desenvolvimento deste trabalho foi o mapeamento estrutural do portal de notícias. Este mapeamento tornou-se necessário devido à presença de conteúdos que não compreendem o artigo presente na página ou aos dados desejados para a extração, como anúncios, links, pop-ups, entre outros que precisam ser excluídos no momento da extração para evitar a poluição dos dados.

Inicialmente, foi feito o mapeamento do portal de notícias onde estão os artigos a serem extraídos, sendo o GE, o portal escolhido para esta extração, além disso, somente os campeonatos brasileiros masculino série A e B do profissional serão analisados. Neste portal, as notícias são organizadas em áreas e/ou blogs dos times, onde são dispostas todas as notícias que envolvem o time.

Após o mapeamento dos sites, iniciou-se o desenvolvimento do extrator dos dados selecionados. Para esta extração foi escolhido o Scrapy, um framework de extração de dados para websites que utiliza a linguagem Python para seu funcionamento. Para esta extração, o Scrapy possui um programa chamado *spider* que tem a função de navegar por meio das páginas na internet e extrair as informações pré-selecionadas. O *spider* baseia-se em três fontes de dados distintas: o portal Globo Esporte, o portal ESPN e a Wikipedia, onde cada uma possui um extrator *spider* próprio para realizar a extração.

Feita extração dos dados, o *spider* gera os respectivos arquivos JSON de cada raspagem, contendo os dados selecionados no mapeamento das páginas. Onde os arquivos contendo os

dados das raspagens do portal da ESPN e Wikipédia serão utilizados como suporte para uma aferição mais assertiva no processamento dos dados do artigo que será analisado pelo Linguakit.

O Linguakit foi a ferramenta utilizada para a análise dos dados extraídos do portal e, diferentemente da configuração do Scrapy, para configurar o Linguakit bastou realizar a clonagem dos arquivos necessários para a execução diretamente do GitHub, que é um repositório de versionamento de arquivos.

Desenvolvido com a linguagem de programação Perl, o Linguakit foi utilizado para realizar a identificação de entidades do artigo, que foi extraído pelo Scrapy na etapa anterior. Para isto, o Linguakit possui diversos módulos que ao serem executados, resultam em dados úteis para análises posteriores.

Feita a extração do artigo, bem como suas informações complementares, o Scrapy armazenou os dados em um arquivo JSON, porém, este formato não é suportado pelo Linguakit. Para resolver este problema, foi desenvolvido um algoritmo que busca os dados do arquivo JSON e retorna um arquivo no formato de texto (TXT), suportado pelo Linguakit.

Para o desenvolvimento deste trabalho, foi utilizado o módulo *tagger* acompanhado do submódulo -nec, assim, realizando a tokenização do artigo e a etiquetagem gramatical de cada um dos tokens. A execução do Linguakit com este módulo retornou 3 informações sobre cada um dos tokens que são: a forma, o lema e a etiqueta. No trecho “participou participar VMIS3S0” tem-se que: (i) a forma é o estado em que a palavra se encontra na frase (“participou”), (ii) o lema é o estado natural da palavra (“participar”), ou seja, sua forma canônica, deve ser sempre escrita no infinitivo e (iii) a etiqueta é o rótulo que classifica gramaticalmente a forma da palavra (“VMIS3S0”), onde o primeiro elemento da etiqueta, refere-se à classe gramatical da palavra, sendo “V” o código da classe Verbo e os outros elementos da etiqueta referem-se a atributos que caracterizam este verbo. No decorrer deste tópico, será apresentado mais detalhes sobre a estrutura de representação das etiquetas.

Esta etiqueta é baseada no conjunto de rótulos propostos pelo Grupo Consultivo de Peritos em Normas de Engenharia da Linguagem (EAGLES, em inglês) que, segundo Calzolari et al. (1996) tem como propósito desenvolver padrões e diretrizes para anotações morfossintáticas de léxicos e corpus.

O processamento dos dados tem por função buscar e organizar informações nos artigos extraídos, como as entidades ou adjetivos que descrevem estas entidades, por exemplo. Para a organização destes dados, foram estabelecidas 6 regras que descrevem e determinam como esta organização se dará.

Regra 1: relacionar entidades simples às entidades compostas, para tratar situações em que a mesma entidade é nomeada de forma diferente, como em “**André Fernandes** joga bem no Flamengo, porém, o futuro de **Fernandes** é incerto no clube".

Regra 2: identificar lista de entidades, para lidar com listas de entidades, como em “O Flamengo terá o reforço de jogadores como Pedro, Vitinho e Hugo Souza”.

Regra 3: substituir etiquetas dos tokens dos times, para diferenciar nomes de times que são semelhantes a nomes de outras entidades, como em “O Flamengo vai enfrentar o Palmeiras em São Paulo” em que o nome não é de um time, e sim de uma localidade.

Regra 4: encontrar adjetivos, como em “Pelé foi genial”.

Regra 5: buscar o dia da próxima partida, como em “A próxima partida do Flamengo será neste domingo, às 16h, contra o Fluminense, no Maracanã”.

Regra 6: relacionar pessoas aos times, para identificar a relação que indica que uma pessoa é um jogador de um time de futebol.

Aplicando-se as regras e com os dados devidamente processados e organizados, deve-se inseri-los no Neo4j, mas para que isso ocorra, os dados precisam ser convertidos para a linguagem *Cypher Query Language* (CQL), que é uma linguagem de consulta de grafo declarativa que permite a consulta, atualização e administração do grafo utilizado no Neo4j (NEO4J, 2021). O resultado é composto por dois arquivos que criam nós correspondentes às entidades e suas relações. Parte do conteúdo desses arquivos é ilustrado pelas Figuras 4 e 5.

Figura 4. Instruções para definição das entidades

```
CREATE (flamengo:Time {id:1, nome:'Flamengo'})
CREATE (palmeiras:Time {id:2, nome:'Palmeiras'})

CREATE (gabigol:Jogador {id:1, nome:'Gabigol'})
CREATE (arrascaeta:Jogador {id:2, nome:'Arrascaeta'})
CREATE (bruno_henrique:Jogador {id:3, nome:'Bruno Henrique'})
```

Figura 5. Instruções para definições de relações entre as entidades

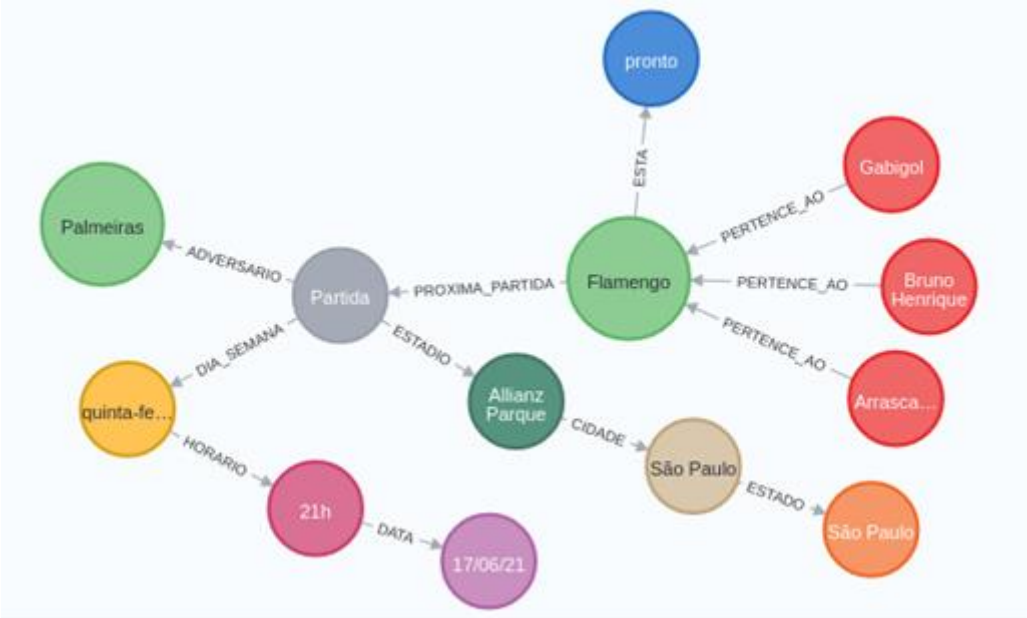
```
CREATE (gabigol)-[:PERTENCE_AO]->(flamengo)
CREATE (arrascaeta)-[:PERTENCE_AO]->(flamengo)
CREATE (bruno_henrique)-[:PERTENCE_AO]->(flamengo)

CREATE (flamengo)-[:ESTA]->(pronto)

CREATE (flamengo)-[:PROXIMA_PARTIDA]->(partida)
CREATE (partida)-[:DIA_SEMANA]->(quinta_feira)
```

O resultado da execução destas instruções no Neo4j é a representação do grafo de conhecimento, que é ilustrado pela Figura 6.

Figura 6. Parte do grafo de conhecimento gerado com a execução do processo descrito neste trabalho



Na Figura 6, é possível observar os nós que representam as entidades de um artigo que foi analisado e processado pela ferramenta em questão. Em verde estão as entidades correspondentes aos times de futebol, que são as principais entidades do artigo, em vermelho e, ligados por arestas nomeadas aos nós dos times, estão os nós dos nomes de jogadores encontrados no decorrer do artigo, onde estes jogadores são devidamente ligados aos times a qual pertencem. Por fim, tem-se as informações complementares encontradas no artigo, no caso da Figura 6, foram encontrados dados sobre a hora, dia e local da próxima partida da entidade “Flamengo”. Ligado à entidade “Flamengo”, foi encontrado o adjetivo “pronto” cuja aresta foi nomeada com o verbo de ligação “ESTA”, organizando esta informação em uma frase: “Flamengo está pronto”.

4 CONCLUSÃO

Este trabalho teve por objetivo desenvolver uma ferramenta que automatizasse o preenchimento de um grafo de conhecimento aplicado a artigos de um portal de notícias esportivas. Para isso, o trabalho foi dividido entre extração dos dados, processamento e alimentação do grafo de conhecimento.

Na extração dos dados, o Scrapy demonstrou ser uma ferramenta eficaz quanto a extração dos artigos, suportando a entrada de diversos links, extraindo cada um dos dados indicados em seus parâmetros por meio de *tags*. Além disso, permitia realizar o tratamento dos dados antes de armazenamento, evitando o desenvolvimento de algoritmos externos para esta função.

O processamento dos dados pode ser dividido em dois pontos principais: análise dos dados por meio do Linguakit e o processamento, propriamente disto, destes dados.

No geral, o Linguakit mostrou ser uma boa alternativa para a tokenização e etiquetagem do texto de acordo com suas classes gramaticais, embora ainda apresente limitações quanto a inferência correta das etiquetas em determinados contextos.

Essa falha é evidente quando há ambiguações nas entidades, como no caso da entidade "Fortaleza" que representa tanto uma localidade, quanto uma organização, o Linguakit, por vezes, etiquetava a entidade "Fortaleza" como uma localidade, porém, no contexto que a entidade estava inserida, sua representação correta era de uma organização.

Outro ponto importante, envolve os apelidos das entidades, onde "Flamengo" pode ser representado como "Mengão", Palmeiras como "Porco" ou Gabriel Barbosa como "Gabigol", por exemplo. Quando haviam apelidos inclusos no artigo, o Linguakit não conseguia aferir da maneira correta qual a etiqueta mais adequada para cada situação, uma vez que "Porco" era, por vezes, etiquetado como "NP00SP0" (pessoa), quando deveria ser "NP00O00" (organização). Embora, em alguns casos a etiquetagem fosse totalmente assertiva, como pode ser visto na seção 4.6.1, Regra 2.

Para trabalhos futuros, além da correção das problemáticas mencionadas acima, novas funcionalidades podem ser adicionadas na ferramenta, como por exemplo: a busca por novos padrões que incidem na descoberta de mais informações sobre as entidades, encorpando o resultado final apresentado no grafo; a ampliação da extração dos dados presentes no artigo, como o conteúdo que não se encontra no texto corrido, sendo um exemplo disto, as citações dos entrevistados pelo repórter; por fim, a aplicação da ferramenta em múltiplos artigos de maneira simultânea, visto que o trabalho, como se encontra atualmente, funciona de maneira eficiente para um artigo por vez, se mais de um artigo for utilizado na ferramenta, haverá duplicações no de entidades no grafo, ou seja, se em um artigo estiver citando sobre o Flamengo e no segundo artigo, também, haverá dois nós "Flamengo" no grafo.

5 REFERÊNCIAS

AMARAL, Daniela Oliveira Ferreira do. **Reconhecimento de Entidades Nomeadas na Área da Geologia**: Bacias Sedimentares Brasileiras. Orientador: Profa. Renata Vieira. 2017. 107 p. Dissertação (Doutorado) - Faculdade de Informática Programa de Pós-Graduação em Ciência da Computação Doutorado em Ciência da Computação, Porto Alegre, Brasil, 2017.

LINGUAKIT (ed.). **Sobre Linguakit**: Linguakit, 2020. Disponível em: https://linguakit.com/pt/sobre-linguakit. Acesso em: 12 novembro 2020.

LOPES, Dener Cesar Ferreira. **Grafos de Conhecimento**: Perspectivas e Desafios para a Organização e Representação do Conhecimento. Orientador: Prof. Dr. Rogério Ap. Sá Ramalho. 2020. 71 p. Dissertação (Mestrado) - Universidade Federal de São Carlos, São Carlos, 2020.

PAN, Jeff Z.; VETERE, Guido; PEREZ, Jose Manuel Gomez-; WU, Honghan. **Exploiting Linked Data and Knowledge Graphs in Large Organizations**. Cham, Suiça: Springer, 2017. 265 p. ISBN 978-3-319-45654-6. *E-book*.

PAULHEIM, Heiko. **Knowledge graph refinement**: A survey of approaches and evaluation methods. Semantic Web Journal, [S. l.], ano 2017, v. 8, n. 3, p. 486-508, 6 dez. 2016. DOI 10.3233/SW-160218.

NEO4J (Malmö, Sweden). **O que é Neo4j?** Estados Unidos: Neo4j, 2020. As referências foram retiradas dos tópicos: "O que é Neo4j" e "A vantagem do Native Graph". Disponível em: https://neo4j.com/. Acesso em: 24 set. 2020.

ROSSUM, Guido van. **Foreword for "Programming Python" (1st ed.).** 1. ed. Virginia, Estados Unidos: Python, 1996. Disponível em: https://www.python.org/doc/essays/foreword/. Acesso em: 24 set. 2020.

SANTOS, Diana. **Introdução ao processamento de linguagem natural através das aplicações**. Caminho, Lisboa, Portugal, p. 229-259, 2001.

SILVA, Daniela Filipa Macedo Braga Moreira da. **Algoritmos de Processamento da Linguagem Natural para Sistemas de Conversão Texto-Fala em Português**. 187 p. Dissertação (Doutorado) - Facultade de Filoloxía da Universidade da Coruña, [*S. l.*], 2008.

SINGHAL, Amit. **Introducing the Knowledge Graph**: things, not strings. 16 maio 2012.

VIEIRA, Renata; LIMA, Vera Lucia Strube. **Linguística computacional**: princípios e aplicações. In: IX Escola de Informática da SBC-Sul. Luciana Nedel (Ed.) Passo Fundo, Maringá, São José. SBC-Sul, 2001.

YAN, Jihong; WANG, Chengyu; CHENG, Wenliang; GAO, Ming; ZHOU, Aoying. **A retrospective of knowledge graphs**. Frontiers Computer Science, Shanghai, p. 55-74, 26 set. 2016. DOI: https://doi.org/10.1007/s11704-016-5228-9.