CURSO DE CIÊNCIA DA COMPUTAÇÃO

Gabriel Dalcin Kothe

SUGESTÕES DE RELAÇÕES ENTRE ELEMENTOS DE UMA ONTOLOGIA ATRAVÉS DE ANÁLISES DE TEXTOS

Santa Cruz do Sul, junho de 2016.

Gabriel Dalcin Kothe

SUGESTÕES DE RELAÇÕES ENTRE ELEMENTOS DE UMA ONTOLOGIA ATRAVÉS DE ANÁLISES DE TEXTOS

Trabalho de Conclusão II apresentado ao Curso de Ciência da Computação da Universidade de Santa Cruz do Sul para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Ms. Eduardo Kroth

Santa Cruz do Sul, junho de 2016.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS

ADV	*Advérbios*
ADJ	*Adjuntos*
BD	*Banco de dados*
CONJ	*Conjunção*
CARD	*Diversos*
EI	*Extração de Informação*
EM	*Entidade mencionada*
IDF	*Inverse document frequency*
KDT	*Knowledge Discovery in Texts*
NOM	*Substantivos*
OBO	*Open Biomedical Ontologies*
OTV	*Ordem de termos e verbo*
SENT	*Pontuação*
PLN	*Processamento de Linguagem Natural*
PRP	*Preposições*
RI	*Recuperação de Informação*
T1	*Termo1*
T2	*Termo2*
V	*Verbos*
VIRG	*Virgulas*
XML	*eXtensible Markup Language*
XIP	*Xerox Incremental Parsing*

SUMÁRIO

RESUMO

Hoje em dia nos deparamos com grandes quantidades de informação espalhada pelo mundo em que vivemos e principalmente, quando estamos conectados na web. Estes dados, sozinhos, podem não ser muito significativos, porém se forem analisados e relacionados entre si, eles podem nos revelar mais informação do que aparentam inicialmente, viabilizando assim, a criação de uma grande rede de relações. O trabalho aqui proposto consiste em implementar um módulo para o software OntoClipping, que possui uma ontologia formada a partir de clipagem eletrônica realizada em paginas web, originalmente desenvolvido por Claudio Omar Correa Carvalho Junior. Este módulo terá como objetivo sugerir relações para os indivíduos que se encontram na ontologia mencionada. Por exemplo, imaginamos uma ontologia onde temos os indivíduos João e Pedro, que pertencem à classe Pessoa, e também os indivíduos Legend e Sunset que pertencem à classe Festa. Ambos, João e Pedro, já frequentaram estas festas, logo temos uma relação entre estes indivíduos. Para criarmos estas relações será necessário realizar a análise de textos que se encontram no banco de dados do OntoClipping e, em cima disso, elaborar ideias e algoritmos para a resolução do problema.

Palavras Chave: ontologia; relacionamento; mineração de texto; identificação; Ontoclipping.

ABSTRACT

Today we are faced with large amounts of information throughout the world we live in and especially when we are connected to the web. These data alone cannot be very significant, but if they are analyzed and related to each other, they can reveal more information than they appear initially, thus enabling the creation of a large network of relationships. The work proposed here is to implement a module for OntoClipping software, which has an ontology formed from electronic clipping held in web pages, originally developed by Claudio Omar Correa Carvalho Junior. This module will aim to suggest relationships for individuals who are mentioned in the ontology. For example, we imagine an ontology where we have individuals John and Peter, who belong to the Person class, and also individuals Legend and Sunset belonging to the Party class. Both John and Peter, have attended these festivals, so we have a relationship between these individuals. To create these relationships will need to perform the analysis of texts that are in OntoClipping database and, on top of that, develop ideas and algorithms for solving the problem.

Keywords: *ontology; relationship; text mining; identification; Ontoclipping;*

1 INTRODUÇÃO

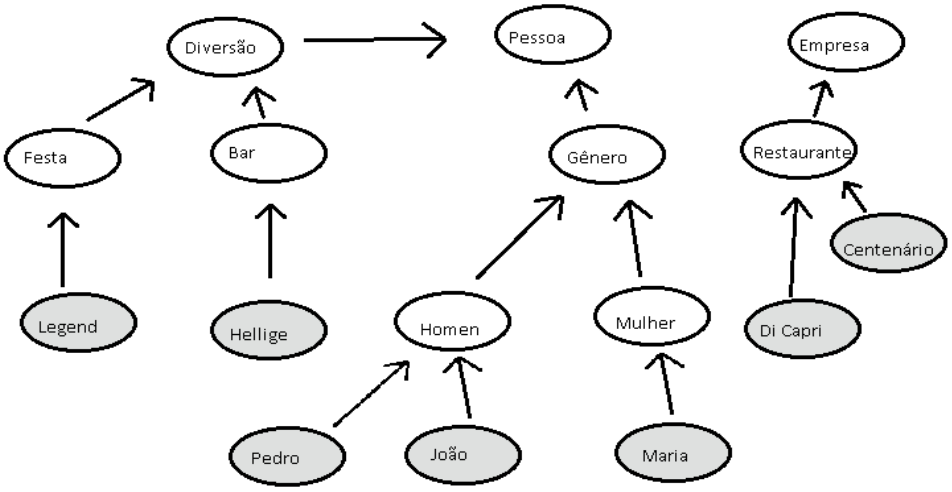
Enormes quantidades de informações atravessam o planeta a cada segundo, trafegando de cabos transatlânticos a satélites em busca de um destino, o ser humano. Estes dados são recebidos no dia a dia, seja pelo noticiário da TV, no programa da tarde do rádio, pela revista de tecnologia quinzenal ou pelo simples clique de um link. Grande parte disso tudo ignoramos e descartamos, outra parte absorvemos e organizamos para nossos interesses e seguidamente, passamos adiante. Porém muitas vezes, deixamos de perceber que essas informações que absorvemos, em cotejo com outros dados que obtemos, podem ter um valor muito maior se forem analisadas, relacionadas e organizadas entre si.

Dentro deste enorme volume de informações, por exemplo, podemos obter várias relações entre entidades que não estão explicitamente descritas em um lugar só. A hipótese do presente estudo é de que se soubermos aproveitar estes dados, podemos criar e organizar uma grande rede de relações diretas e indiretas de uma ou mais entidades específicas. Neste contexto, podem-se considerar entidades como empresas, organizações, grupos e pessoas, etc.

Visualizando uma solução computacional, tem-se uma arquitetura de software composta por um ambiente de coleta e indexação de dados, um ambiente de tratamento e identificação das possíveis relações e um terceiro ambiente de consultas. Para ilustrar o ambiente de coleta e indexação de dados, tem-se os trabalhos relacionados ao software OntoClipping desenvolvido através de trabalhos de conclusão desta universidade, este sendo “uma ferramenta de clipagem eletrônica que utiliza uma *ontologia* como forma de representação do conhecimento, juntamente com técnicas de recuperação da informação aliadas a um motor de busca de páginas web” (SCHUSTER, 2013 p.4).

Consideramos a figura 1 um exemplo de ontologia composta por classes e indivíduos, que poderia ser encontrada dentro do sistema do OntoClipping. Esses indivíduos, neste caso são os nodos que se encontram por último, no nível mais baixo da ontologia, são eles: Legend, Hellige, Pedro, João, Maria, Di Capri, Centenário. Como podemos ver, todos eles estão relacionados a uma classe, porém, no estado atual, é a única relação que temos na ontologia. Não sabemos se João já esteve na Hellige ou se ele conhece Maria, se tem alguma relação de parentesco com Pedro, ou se ele já frequentou o Centenário ou o Dicapri.

Figura 1 – Exemplo de ontologia.



Fonte: Autor.

Para identificarmos estas conexões entre os indivíduos, primeiro precisamos verificar as ocorrências destes nos textos que se encontram no banco de dados, e nestes mesmos textos procurar por outros indivíduos que estão presentes na ontologia, para que futuramente possamos desenvolver um algoritmo que possa identificar, ou pelo menos sugerir uma conexão entre eles.

É justamente na aferição da viabilidade de implementação de uma ferramenta para realização destes processos de reconhecimento e sugestão de relações que repousa o objetivo do presente trabalho.

1.1 Objetivo principal

Desenvolver um algoritmo para recomendação de relacionamentos entre indivíduos de uma ontologia a partir de textos e informações coletadas de recomendações e tentativas passadas.

1.2 Objetivos específicos.

- Especificar algoritmo para identificação de nomes próprios (indivíduos) dentro de um texto.
- Analisar e reconhecer relações entre indivíduos que foram encontrados nos textos com outros indivíduos que se encontram em uma ontologia através de um algoritmo.
- Sugerir ao usuário a melhor relação que resultará do algoritmo criado.

2 FUNDAMENTAÇÃO TEORICA

2.1 Processamento de Linguagem Natural

Quando se lê alguma reportagem sobre um determinado assunto, frequentemente nos deparamos com frases como “Homem assalta posto de gasolina, armado com um revólver e deixa um ferido.” Neste exemplo, sabemos que quem estava armado era o homem, e não o posto, por causa de nosso conhecimento contextual da língua. Porém para uma máquina, obter esta informação é um processo mais complicado e nem sempre efetivo. A área que trata destes cenários chama se de “O Processamento de Linguagem Natural (PLN)” que “é a área da Ciência da Computação que tem por objetivo estudar mecanismos de processamento da linguagem falada e escrita, com o intuito de convertê-la para uma representação mais formal, e assim torná-la manipulável por programas de computador.” (HACK et al, 2013, p.8).

O PLN, por ser uma área relativamente complexa de se trabalhar, tem sido abordado por muitos estudantes, devido à grande quantidade de informações que podemos encontrar online e, com sucesso, uma vez que pode ser aplicado em várias áreas de utilização, como sistemas de busca e recuperação de documentos, tradutores, sistemas inteligentes onde há a interação do usuário com perguntas, respostas e “*parsers*” de textos, como por exemplo, o TreeTagger -conteúdo melhor descrito na seção 2.1.1.1-. “Normalmente um sistema PLN é abordado do ponto de vista da análise do conhecimento (i) morfológico, (ii) sintático, (iii) semântico e (iv) pragmático.”(MULLER, 2002, p.33). Entretanto, essas quatro etapas de conhecimento não necessariamente precisam ser executadas em ordem e também não há necessidade de aplicá-las juntas, pois depende de como o PLN será utilizado.(HACK et al, 2013)

2.1.1 Análises

A presente seção será utilizada no intuito de apresentar as quatro etapas de conhecimento citadas acima, bem assim sua conceituação específica dada pelos autores adotados no presente estudo.

2.1.1.1 Análise morfológica

A análise morfológica, “estuda a construção das palavras, com seus radicais e afixos, que correspondem a partes estáticas e variantes das palavras, como as inflexões verbais.” (MULLER, 2002, p.33). Sua função, consiste em identificar a classe morfológica de uma palavra (substantivo, verbo, pronome, etc.), bem como de sua inflexão, seja ela nominal (ex. gênero) ou verbal (ex. Pessoa) (HACK et al, 2013). Assim, ao utilizarmos o programa TreeTagger para realizar a análise morfológica da seguinte frase : “Você é muito devagar para correr nas maratonas desta cidadezinha”. Teremos o resultado demonstrado na figura 2, senão veja-se:

Figura 2 – Exemplo TreeTagger

Análise morfo-sintática:

Você P você
é V ser
muito ADV muito
devagar ADV devagar
para PRP para
correr V correr
nas PRP+DET em
maratonas NOM maratona
desta PRP+P de
cidadezinha NOM cidade

Fonte: Autor.

Desta forma, conforme demonstra a figura 2, o TreeTagger nos dá como resposta uma lista com as palavras analisadas, seu tipo e seu significado no radical e singular, também chamado de *lemma*, que é precisamente o resultado da análise morfológica.

2.1.1.2 Análise sintática

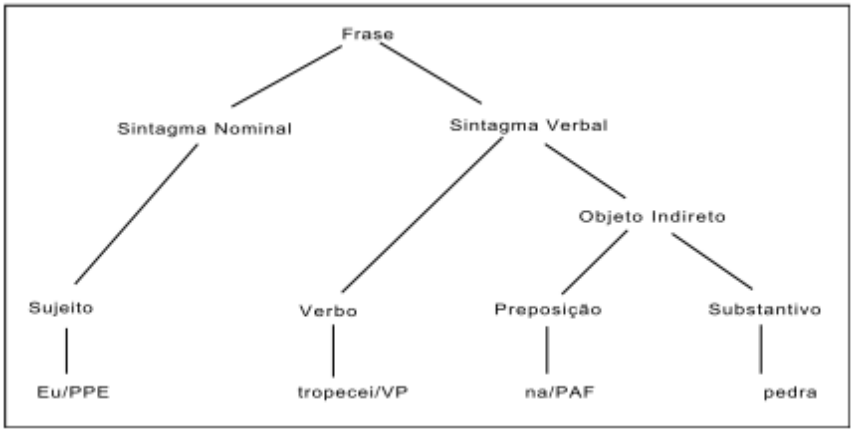
No que se refere à análise sintática, o seu objetivo é,

> estabelecer as relações formais entre as palavras de uma frase, baseado nas regras gramaticais da linguagem na qual a frase foi escrita. O processo de divisão de uma frase em sintagmas nominais e verbais consiste na divisão de uma frase em segmentos cujo núcleo (palavra principal) será um nome (sintagma nominal) ou um verbo (sintagma verba). (HACK et al, 2013, p.9)

A análise sintática (*parsing*) é o procedimento que avalia os vários modos de como combinar regras gramaticais, com a finalidade de gerar uma estrutura de árvore que represente a estrutura sintática da sentença analisada. Se a sentença for ambígua, o analisador sintático (*parser*) irá obter todas as possíveis estruturas sintáticas que a representam. (NETO, TONIN, PRIETCH, 2010, apud GONZALEZ e LIMA, 2003).

A fim de melhor exemplificar a análise sintática, veja-se a figura abaixo:

Figura 3. Exemplo de uma arvore sintática.



Fonte: (Muller, 2015).

Então, diferente da etapa da analise morfológica, que analisa e estrutura as palavras em diferentes categorias, na etapa de analise sintática temos a análise por agrupamento de frases. “A análise sintática diz respeito ao estudo das relações formais entre palavras”. (MULLER, 2002 apud JURAFSKY e MARTIN, 2000, p.33).

2.1.1.3 Análise semântica

A análise semântica tem como objetivo tentar entender o enunciado processado, tratando o significado das palavras, expressões e orações completas. Podendo também levar em conta declarações contextualizadas, apesar de esta área já ser mais voltada para a análise pragmática. Isso significa traduzir a expressão original em alguma forma de metalinguagem semântica ou sistema de representação. (JUNIOR, 2013, apud GODDARD e SCHALLEY, 2010).

> O processo de análise semântica do texto busca o mapeamento de sentenças de uma linguagem, visando a representação de seu significado, baseado nas construções obtidas nas análises morfológica e sintática. Este tipo de análise é a

> mais difícil de ser implementada, pois envolve análise do significado de palavras e expressões. (HACK et al, 2013, p.9).

Em relação a análise sintática,

> "que se preocupa com a estrutura da sentença, a semântica foca no significado contido. Nas linguagens utilizadas entre seres humanos para a comunicação é comum à semântica se sobrepor à sintaxe, devido ao objetivo ser a transmissão da informação que está contida no significado da sentença". (MACHADO, 2015, p.18)

2.1.1.4 Análise pragmática

"A análise pragmática diz respeito ao processamento da forma que a linguagem é utilizada para comunicar, como os significados obtidos na análise semântica agem sobre as pessoas e seu contexto." (MULLER, 2002, p.33).

> "Analise pragmática é o processo de análise do significado de determinados termos do texto, baseado no contexto no qual ocorrem. Esta etapa geralmente é necessária devido ao fato de que o texto geralmente é processado frase por frase. E isto pode ser um problema por que é muito comum existirem frases que dependem de outras sentenças no texto analisado para que tenham o seu significado compreendido". (HACK et al, 2013, p.11).

Hoje em dia a análise pragmática é a análise mais imatura que encontramos na área de PLN. "Sua complexidade intimidam pesquisas na área. Percebe-se, porém, que aplicações podem bem utilizar seus resultados, sobretudo aquelas que mais se aproximam do usuário final, quais sejam motores de Tradução Automática ou sistemas de pergunta–resposta". (JUNIOR, 2013, p.75).

2.1.2 Corpus

Assim como o ser humano precisa ter conhecimento da linguagem para poder processar a informação passada, sistemas que trabalham com PLN também precisam de uma base de conhecimento para realizar suas tarefas. Geralmente, encontra-se, nestes sistemas grandes arquivos de texto com muitas palavras em suas classificações da língua em questão.

Corpus é,

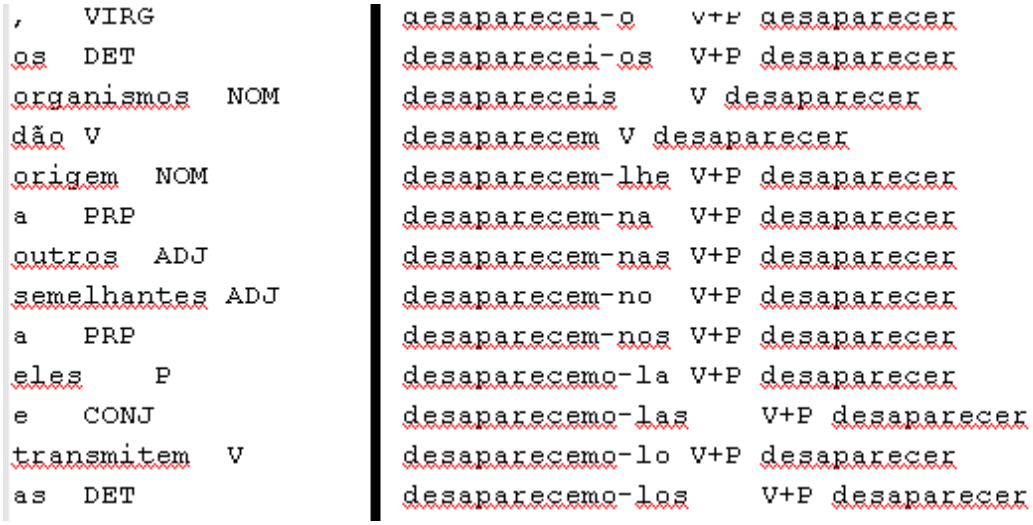
> "um conjunto finito de enunciados tomados como objeto de análise. Mais precisamente, conjunto finito de enunciados considerados característicos do tipo de língua a estudar, reunidos para servirem de base à descrição e, eventualmente, à elaboração de um modelo explicativo dessa língua. Trata-se, pois, de uma coleção de documentos quer orais (gravados ou transcritos) quer escritos, quer orais e escritos, de acordo com o tipo de investigação pretendido. As dimensões do corpus variam segundo os objetivos do investigador e o volume dos enunciados considerados como característicos do fenômeno a estudar. Um corpus é chamado **exaustivo** quando compreende todos os enunciados característicos. E é chamado **seletivo** quando

> compreende apenas uma parte desses enunciados.” (ALUÍSIO E ALMEIDA, 2006 APUD GALISSON E COSTE, 1983, p.2) .

Quanto ao tamanho e tipo de corpus, devemos ter em mente o tipo de tarefa para qual ele será necessário e qual o tipo de metodologia adotada no processamento de texto. Não devemos relevar somente o número total de palavras (*tokens*) e de palavras diferentes (*types*), mas também a quantidade de categorias (gêneros discursivos, tipos de textos, datas, autores, etc.) que estarão inclusas e quantas amostras de cada teremos disponíveis. (ALUÍSIO E ALMEIDA apud SINCLAIR, 2005).

Na figura 4, temos um exemplo, o corpus do TreeTagger:

Figura 4 – Exemplo, corpus do TreeTagger.



Fonte: Autor

Note-se que do lado direito temos algumas palavras e seus tipos, e no lado esquerdo temos um exemplo das variações da palavra “desaparecer”.

2.2 Introdução à mineração de texto.

Mineração de textos é uma área que procura extrair informações de um texto em linguagem natural, analisando regularidades, padrões ou tendências em grandes volumes de textos, através de técnicas de analise e extração de dados. Importante salientar que a mineração de textos se difere da busca casual que estamos acostumados a fazer na internet, porque esta busca é realizada a partir de um conhecimento já obtido pelo pesquisador, enquanto a mineração de texto tem como objetivo descobrir informações antes não conhecidas.

> A descoberta do conhecimento em texto nasceu a partir da necessidade de descobrir informações, padrões e anomalias em textos de forma automática. Essa tecnologia permite recuperar informações, extrair dados, resumir documentos, descobrir padrões, associar regras e realizar análises qualitativas ou quantitativas em documentos de texto. (SILVA, 2010, APUD ARANHA E PASSOS, 2006, p.25).

A mineração de texto refere-se ao processo de Descoberta de Conhecimento em Texto, conhecida pela sigla em inglês como KDT(Knowledge Discovery in Texts), consiste na obtenção de informação a partir de texto em linguagem natural ou passível de interpretação. A mineração de texto extrai informação de dados estruturados ou semiestruturados. (HACK et al, 2013).

A mineração de texto é uma variação da área chamada de Mineração de Dados que, por sua vez, é:

> "uma área de pesquisa multidisciplinar, incluindo tecnologia de bancos de dados, inteligência artificial, aprendizado de máquina, redes neurais, estatística, reconhecimento de padrões, sistemas baseados em conhecimento, recuperação da informação, computação de alto desempenho e visualização de dados. Em seu sentido relacionado a banco de dados, trata-se do processo de extração ou mineração de conhecimento a partir de grandes volumes de dados." (MORAIS E AMBRÓSIO, 2007, p.5).

Portanto, diferentemente da mineração de texto, a mineração de dados trabalha com a extração de dados estruturados.

A mineração de textos abrange uma grande quantidade de áreas de conhecimento, além do PLN, temos Mineração na Web (Web Mining), Recuperação de Informação, Extração de Informações, Web Semântica, Inteligência Competitiva, entre outras, conforme veremos a seguir.

2.2.1 Extração de Informações (EI)

A técnica de Extração de Informação (*Information Extraction*) tem por objetivo localizar itens específicos dentro de um documento textual não estruturado, para então estruturá-los em formatos processáveis por máquinas, por exemplo, um banco de dados relacional ou arquivos XML. (HACK et al, 2013). Sendo assim, as analises e manipulações dos textos acabam por serem facilitadas. Importante lembrar que Sistemas de EI não realizam o entendimento completo dos documentos, pelo contrário, tais sistemas analisam partes dos documentos que possuam informações relevantes. (ALVÁREZ, 2007)

> "Extração de Informação não deve ser confundida com a desenvolvida área de Recuperação de Informação (RI), a qual seleciona, de uma grande coleção, um subconjunto de documentos relevantes baseados em uma consulta do usuário. O contraste entre os objetivos dos sistemas de EI e RI pode ser declarado da seguinte forma: RI recupera documentos relevantes de uma coleção, ao passo que EI extrai informações relevantes dos documentos." (ALVÁREZ, 2007, p.62).

Portanto, a Extração de Informação e a Recuperação de Informação são técnicas complementares para a busca de informações.

2.2.2 Recuperação de Informação

Recuperação de Informação (RI) é:

> A busca por documentos que contêm as respostas às perguntas e não tem por função específica encontrar as respostas propriamente ditas. Para alcançar este objetivo medidas e métodos estatísticos são utilizados para o processamento automático de dados de texto e comparação com a questão dada. Sistemas que executam a recuperação de documentos como por exemplo, a maioria dos motores de busca, muitas vezes também são chamados de sistemas de recuperação de informação. (Machado et al, 2010).

Assim sendo, podemos afirmar que um sistema de RI tem como meta encontrar a informação exigida para satisfazer a necessidade de informação (NI) do usuário, de modo que para atingir tal objetivo, além da recuperação propriamente dita, um sistema de RI deve ser capaz de realizar armazenamento e manutenção de informação, de modo que além do procedimento de busca, pode incluir catalogação, categorização e classificação de informação, particularmente na forma textual. (GONZALES E LIMA, 2003).

Em outras palavras, deve representar, organizar e dar acesso a itens de informação (documentos).

2.2.3 Web Semântica

A Web Semântica tem como objetivo principal criar estruturas e dar significado semântico ao conteúdo das páginas que encontramos da Internet, com a finalidade de desenvolver um ambiente onde agentes de software e usuários possam trabalhar de forma cooperativa.

> Dessa forma será possível melhorar os resultados das buscas e facilitar a automação de tarefas relevantes, evitando, sempre que possível, o desperdício de tempo e fazendo com que tanto os dados quanto as máquinas sejam utilizados de forma mais inteligente. Para ilustrar um exemplo de aplicação da Web Semântica, imagine que um usuário deseja viajar de uma localidade a outra. Para isso terá que pegar um

> avião e depois ficar hospedado em um hotel. Com o advento da Web Semântica, será possível realizar uma busca pela passagem mais barata para o destino em questão, incluindo as restrições de horário de saída e chegada. (FILHO E LÓSCIO, [201?], p.2).

2.2.4 Inteligência Competitiva

A Inteligência Competitiva (IC) é,

> "um programa formal e sistematizado que permite o acompanhamento da evolução da organização e também do comportamento dos concorrentes atuais e em potencial a fim de manter e desenvolver uma vantagem competitiva, utilizando-se de profissionais atentos todo o tempo, coletando informações implícitas constantemente, suprindo, com isso, a necessidade de utilização de informações, não só a nível operacional, mas também estratégico." (FURTADO, 2004, p.12).

Por ser uma área muito ampla, ela não só faz parte da área da mineração de texto, mas também da área de mineração de dados. Na IC é aplicado um conjunto de técnicas de ambas as áreas, para que assim se possa extrair informações suficientes para alguma tomada de decisão que o usuário precise fazer.

2.2.5 Web mining

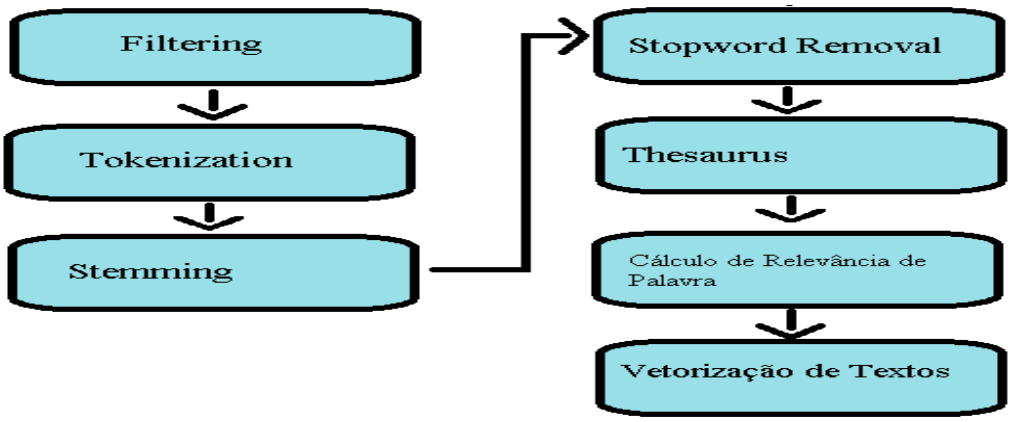
Web Mining como é o uso de técnicas de data mining para descobrir e extrair automaticamente informações relevantes dos documentos e serviços ligados a Internet, apesar de frequentemente ser associado com "Recuperação da informação", ele na verdade trata-se e de um processo mais amplo, interdisciplinar, envolvendo técnicas de Recuperação de Informação, estatística, inteligência artificial e mineração de dados. Este campo de pesquisa é extremamente extenso e por isso, muita confusão surge quando se discute sobre as tarefas centrais de Web Mining. (AMO, [201?]).

2.3 Técnicas de mineração de texto

A primeira etapa da mineração de texto é a preparação dos textos a serem analisados para obtermos um melhor resultado no final, esta etapa tentar identificar similaridades em função da morfologia ou do significado dos termos nos textos, bem como prover uma redução dimensional do texto analisado. (HACK et al, 2013)

A fim de demonstrar como funciona o fluxo de pré-processamento de textos, veja-se o diagrama abaixo, cujas etapas serão abordadas individualmente a seguir.

Figura 5 – Diagrama das etapas



Fonte: Autor.

Antes de adentrarmos na análise individual, contudo, é relevante observar que, dependendo da aplicação e objetivo que se pretende atingir, nem todas técnicas são utilizadas, e dependendo do autor, é possível encontrar a ordem das etapas um pouco diferente das demonstradas nesta seção, assim como podemos encontrar outros técnicas aqui não descritas.

2.3.1 Filtering

Nesta etapa acontece a remoção de caracteres especiais, que de modo generalizado não alteram o contexto do documento, apesar de eventualmente poderem modificar o significado de frases e outras vezes parágrafos, o significado geral do documento fica inalterado, o contexto do documento é definido pelas palavras mais significantes no texto e os caracteres de pontuação não tem valor para o seu processamento. (HACK et al, 2013).

2.3.2 Tokenization

Depois da etapa de *filtering*, aplicamos técnicas de processamento de linguagem natural para identificar as palavras das frases e também para tentar identificar o significado das frases, a partir das relações de suas palavras. Durante o processo as palavras usadas no corpo do texto são classificadas para uso nas etapas a seguir. (HACK et al, 2013).

De acordo com Junior “cada unidade é chamada de *token* e que, na grande maioria das vezes, corresponde a uma palavra do texto, podendo também estar relacionado a mais de uma palavra, símbolo ou caractere de pontuação.” (JUNIOR, 2007, p.31).

Um simples exemplo: “Jõao foi correndo e chorando para casa.”

O resultado da etapa de *tokenization* seria:

[João][foi][correndo][e][chorando][para][casa].

Como podemos observar no exemplo descrito, caractere que sempre é descartado na geração de *tokens* é o “espaço”.

2.3.3 Stemming

Esta etapa tem como objetivo reduzir a palavra à sua forma mais básica, assim eliminando os sufixos que indicam variação na forma da palavra, como plural e tempos verbais, tornando o documento mais consistente para uma futura analise. Porém, conforme bem alertado por (Junior, 2007), em casos onde o contexto da palavra é importante, esta etapa pode não ajudar muito.

Um exemplo da etapa em comento, é a reposta que o TreeTagger nos retorna na figura 2 apresentada anteriormente.

2.3.4 Stopword removal

Textos tem naturalmente um grande volume de dados, o tempo de análise de vários corpos de texto cresce rapidamente, para amenizar o tempo de processamento é necessário diminuir do total de dados que será processado removendo dados que não tem grande valor. Palavras que são muito utilizadas no texto e que não trazem significado direto a uma frase são removidas, diminuindo assim o corpo do texto.

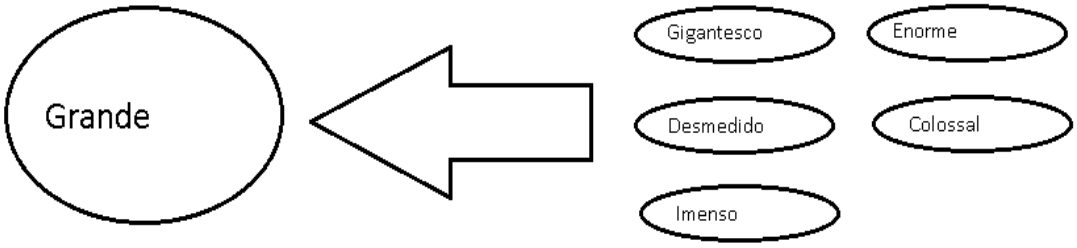
As palavras muito frequentes na escrita são consideradas como de pouco valor, ou seja, tudo aquilo que não é substantivo, adjetivo ou verbo é removido. (HACK et al, 2013).

2.3.5 Thesaurus

A presente etapa se torna necessária na medida em que palavras diferentes podem ter o mesmo significado. Assim, é imprescindível o uso de dicionário de *thesaurus*, que relaciona estas palavras encontradas no texto com seus sinônimos no dicionário, normalizando o texto

em um corpo bem padronizado e ajudando a evitar erros de cálculo na etapa de relevância de palavras. (HACK et al, 2013).

Figura 6 - Exemplo de uma relação Thesaurus.



Fonte: Autor

O exemplo da figura 6 demonstra uma relação em um possível dicionário *thesaurus*, onde as palavras “gigantesco”, “desmedido”, “imenso”, “enorme”, “colossal” seriam automaticamente substituídas, quando encontradas em textos analisados, pela palavra “grande”, ocorrendo assim a referida normalização do texto.

2.3.6 Cálculo de relevância de palavra

Nem todas as palavras presentes em um documento possuem a mesma importância. Os termos mais frequentemente utilizados (com exceção das *stopwords*) costumam ter significado mais importante, assim como as palavras constantes em títulos ou em outras estruturas. Esta etapa que faz o cálculo da relevância de uma palavra dentro de um documento, tem como objetivo obter um peso referente ao uso do termo dentro do texto analisado. Existem várias fórmulas para cálculo do peso. As mais comuns são baseadas em cálculos simples de frequência: frequência absoluta, frequência relativa, frequência inversa de documentos. (HACK et al, 2013)

2.3.6.1 Frequência absoluta

A frequência absoluta, também chamada de “frequência do termo”, calcula a quantidade de vezes que um termo aparece em um documento. Por ser a medida de peso mais simples que existe, ela não leva em conta a quantidade de palavras existentes em um documento. Com isso, conforme bem apontado por Morais e Ambrósio (2007), uma palavra pouco frequente

em um documento pequeno pode ter a mesma importância de uma palavra muito frequente de um documento grande.

2.3.6.2 Frequência relativa

Frequência relativa é o tipo de análise que leva em conta o tamanho do documento (quantidade de palavras que ele possui) e normaliza os pesos de acordo com essa informação. A frequência relativa (Frel) de uma palavra x em um documento qualquer é calculada dividindo-se sua frequência absoluta (Fabs) pelo número total de palavras no mesmo documento (N), Frel(x) = Fabs(x)/N" . (MORAIS E AMBRÓSIO, 2007).

2.3.6.3 Frequência inversa de documentos

A frequência inversa de documentos,

> "busca normalizar termos frequentes com base na sua ocorrência em todos os documentos analisados. O cálculo da frequência inversa de documentos (inverse document frequency - IDF) é realizado com base na informação da frequência absoluta do termo no documento e da frequência do termo em todos os documentos, e isto é capaz de aumentar a importância de termos que aparecem em poucos documentos, e, diminuir a importância de termos que aparecem em muitos, justamente pelo fato dos termos de baixa frequência serem, em geral, mais discriminantes". (HACK et al, 2003)

A fórmula mais comumente utilizada para cálculo do peso de um termo utilizando a frequência inversa é lembrada por Morais e Ambrósio, 2007, sendo que:

Peso*td* = Freq*td*/DocFreq*td* , onde:

Peso*td*: é o grau de relação entre o termo t e o documento d;

Freq*td*: número de vezes que o termo t aparece no documento d;

DocFreq*td*: número de documentos que o termo t aparece.

2.3.7 Vetorização de textos

A vetorização de textos, também conhecida como Modelo Espaço-Vetorial, consiste na representação de um texto na forma de um vetor de termos. A forma mais comum de vetorização de textos é quando cada documento é representado por um vetor de termos

(*tokens*) e cada termo possui um valor associado que indica o grau de importância (denominado peso) desse no documento. (MORAIS E AMBRÓSIO, 2007).

Por exemplo: (cão, peso2), (gato, peso3), (passarinho, peso1). Estes pesos podem ser calculados pelas formulas comentados anteriormente na seção de calculo de relevância de palavras.

Uma das desvantagens desta técnica é a necessidade de novo processamento da coleção de documentos quando esta é alterada e a ausência de relação semântica entre os *tokens* de uma coleção.

2.4 Ontologias

O termo ontologia é originário da filosofia. Ontologia é um ramo da filosofia que lida com a natureza e a organização do ser, onde filósofos tentam responder as questões "O que é um ser?" e "Quais são as características comuns de todos os seres?". Esse termo foi introduzido por Aristóteles em Metafísica, IV, 1. (GUIMARÃES apud Maedche, 2002).

"Aristóteles apresenta categorias que servem de base para classificar qualquer entidade e introduz ainda o termo "*differentia*" para propriedades que distinguem diferentes espécies do mesmo gênero". (ALMEIDA E BAX, 2003).

De forma simples, para elaborar ontologias definem-se categorias para as coisas que existem em um mesmo domínio, de modo que podemos dizer que Ontologia é um "catálogo de tipos de coisas". (ALMEIDA E BAX, 2003 apud SOWA, 1999).

A ontologia aplicada à Ciência da Computação e informática é um termo que denota um objeto que é projetado para um propósito: permitir a modelagem de conhecimento sobre algum domínio, sendo ele real ou imaginário. Diferente do conceito filosóficos, na área de TI as ontologias são consideradas como conjuntos de definições e conceitos. (FEZTER, 2013 apud GRUBER, 1992, pag. 11).

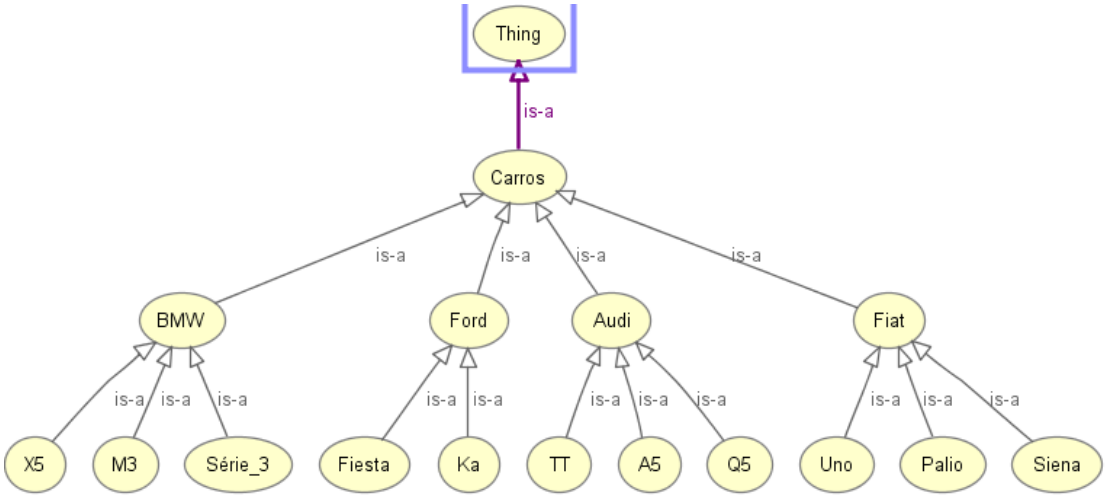
Conforme apontado por Fetzer (2013),

> Morais e Ambrósio apresentam algumas áreas da Ciência da Computação em que a ontologia tem seus usos, dentre elas são destacados a recuperação de informações na internet, comentando sobre projetos relacionados e a possibilidade de busca por grandes bancos de ontologias na internet; Processamento de linguagem natural, exemplificando com o uso da ontologia para facilitar a tradução de textos médicos que falam sobre oncologia (referente a câncer) usando uma ontologia montada a partir de textos e dicionários especializados e específicos da área; Web-semântica, apontando que "informação é dada com significado explícito, tornando mais fácil para máquinas processarem e integrarem informações disponíveis na rede automaticamente.". (FEZTER, 2013 apud MORAIS E AMBRÓSIO, 2007, pag. 11).

2.4.1 Elementos de uma ontologia.

Nem todas as ontologias são iguais, sua estrutura muito depende do objetivo e assunto que ela pretende abordar, porém a maioria das ontologias possui alguns elementos básicos, sendo os principais as classes, indivíduos, atributos e relacionamentos. (FETZER, 2013).

Figura 7 - Exemplo de uma ontologia de carros e marcas



Fonte: Fetzer, 2013, p.10

2.4.1.1 Classes

Classes podem ser compostas pode outras classes e indivíduos ou ambos, sendo normalmente organizadas em taxonomias. São conjuntos ou grupos de objetos reais ou abstratos, que representam alguma relação entre o domínio escolhido e a ontologia. Comparando com uma estrutura de árvore, as classes seriam nodos que possuem filhos. (FETZER, 2013).

Nesse sentido, como demonstrado na figura 7, temos a BWM que pertence à classe Carros, que por sua vez pertence à classe Coisa (*thing*).

2.4.1.2 Indivíduos

Indivíduos são considerados os elementos mais básicos de uma ontologia. Um dos propósitos gerais da ontologia é definir e classificar os indivíduos. Eles estão em relacionado

as classes e, comparando novamente a uma estrutura de árvore, eles seriam os nodos folhas, pois não possuem filhos. (FETZER, 2013).

Exemplificando, na figura 7, temos os indivíduos da classe Ford, que são os Fiesta e Ka. Eventualmente, eles poderiam ser transforados em classes se, por exemplo, adicionarmos os indivíduo Ka XR 1.6 (Zetec Rocam), Ka MP3 (1.0 ou 1.6) e Ka GL Image 1.0 (Zetec Rocam) que são versões do Ka. Ou seja, está é uma área sem uma definição concreta, a classificação dos indivíduos geralmente vai depender do nível de abstração da ontologia.

2.4.1.3 Atributos

Os atributos são um campo destinado a armazenar valores inerentes aos objetos pertencentes a uma classe ou indivíduo, dando a eles uma descrição mais completa.

Fetzer (2013) define os atributos como a "junção de nome e valor que é usado para guardar informações relevantes sobre determinados objetos."

Ou seja, dentro de uma classe ou indivíduo podemos ter estas propriedades especificas sobre o objeto em si. Um exemplo seria o apontamento das características peculiares do indivíduo "Ka", de acordo com seus modelos.

Assim, poderíamos ter:

Classe: KA

Indivíduo: Ka XR 1.6 (Zetec Rocam)

Atributos:

Ano: 1996

Peso: 820–962 kg

Rodas: 4

2.4.1.4 Relacionamentos

É o uso de atributos para descrever as relações entre objetos, sejam eles indivíduos ou classes. Como por exemplo, na ontologia apresentada na figura 7, temos as classes "Ford" e "carro" e a relação entre eles é "é marca de", logo a expressão completa é "Ford é marca de carro". (FETZER, 2013).

A construção destes relacionamentos será a principal área abordada neste trabalho, que é melhor descrita na seção 2.4.2.

2.4.2 Conceitos para identificação e construção de relacionamentos

Nesta seção serão apresentados alguns conceitos e métodos que podem auxiliar na construção e identificação de relações de elementos de uma ontologia.

2.4.2.1 Os 5 w's

O que são os "5W's"? Who, What, Why, When, Where, que traduzido para português, significam Quem?, O Que?, Porque?, Quando?, Onde? são consideradas perguntas básicas para a coleta de informações, usadas principalmente na área de jornalismo e investigações gerais.

A hipótese deste conceito, é que estas perguntas constituem uma fórmula para criar uma história completa em relação a um assunto específico. Se o artigo ou matéria escrita não consegue responder todas as questões que o 5w's representam, significa que ele carece de informações, porque é através delas que conseguimos criar as relações necessárias entre as entidades envolvidas no assunto e assim conseguir extrair as informações relevantes do texto que estamos lendo e analisando.

What - O que?

Geralmente trata sobre um evento que aconteceu, um problema a ser resolvido. Podemos considerar que é o núcleo da matéria.

Why - Por que?

O motivo do evento ter acontecido ou o porque da necessidade da resolução de um problema.

Who - Quem?

Quem esteve envolvido no evento ou quem faz parte do problema a ser resolvido.

Where - Onde?

Onde o evento aconteceu ou onde o problema a ser resolvido aconteceu.

When - Quando?

Quando o evento aconteceu ou em que momentos o problema a ser resolvido se manifesta.

2.4.2.2 Relacionamentos em ontologias OBO

> "As ontologias OBO (*Open Biomedical Ontologies*), que são ontologias desenvolvidas para a comunidade biológica, são bastante citadas quando se trata de relações entre entidades ontológicas. Estas relações podem ser aplicadas de duas maneiras, ou a nível de classe ou a nível de instancia. Relações a nível de classe se refere, por exemplo, dedo <é_parte_da> mão. Isso significa que para qualquer dedo no mundo real existe uma mão no mundo real. Já uma relação entre particularidades é uma relação de nível de instância, como por exemplo, o dedo polegar de João <é_parte_da> mão de João."(FETZER, 2013, p.15).

Com base nestas premissas, o autor nos apresenta a classificação das relações que se encontram na OBO:

- **Direcionalidade** – Uma relação que tem uma direção, como por exemplo, folha <é_parte_de> uma planta, mas essa relação não afirma que todas as plantas possuem folha. Para que todas as plantas tenham folhas, precisaria de outro relacionamento.
- **Simetria** – A relação é simétrica se ela se aplica ambos os sentidos. <Igual> e <está do lado> são alguns exemplos de relações simétricas.
- **Transitividade** – É quanto o relacionamento é fiel nas cadeias de ligação. Alguns exemplos são <é parte de> e <é maior que>. Se A é maior que B e B é maior que A, então A é maior que B.
- **Ciclicidade** – Quando há um ciclo de relações. Por exemplo, uma instância de A pode desenvolver uma instância de B, e mais tarde uma instância de B pode desenvolver uma instância de A. Normalmente esse tipo de relação envolve alguma forma de mudança ao longo do tempo.

Acredita-se que, eventualmente, o conceito dos relacionamentos de ontologias OBO poderá auxiliar na concretização dos objetivos do presente estudo.

2.4.2.3 HAREM

Em 2006 foi criado o evento de avaliação HAREM (Avaliação e Reconhecimento de Entidades Mencionadas), com o propósito de incentivar o desenvolvimento de sistemas voltados para a tarefa de identificar e classificar automaticamente nomes próprios em categorias previamente definidas, em textos escritos em português. (BARRETO, 2010, apud SANTOS & CARDOSO, 2008).

Dois anos depois, foi realizado um Segundo HAREM, que manteve a os passos do primeiro, em relação ao modelo semântico e ao modelo geral de avaliação, porém a incluiu

duas novas tarefas de pesquisa: i) o reconhecimento e a padronização de expressões temporais; e ii) o reconhecimento de relações semânticas entre entidades mencionadas (EM), esta tarefa foi nomeada de ReRelEM. (BARRETO, 2010).

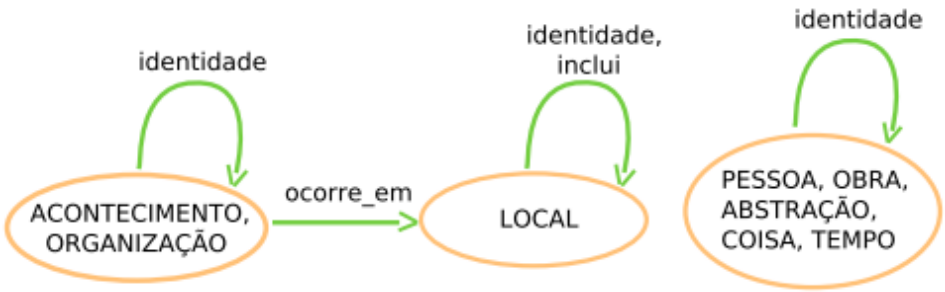
A proposta do HAREM é que todas as entidades mencionadas sejam identificadas e classificadas. As categorias de EMs no Segundo HAREM são: PESSOA, ORGANIZAÇÃO, ACONTECIMENTO (que chamamos EVENTO), LOCAL, ABSTRAÇÃO, OBRA, VALOR, COISA, OUTRO. A última categoria, OUTRO é uma forma de contemplar entidades que não se encaixem nas demais, para que não seja necessário deixar de classificar alguma entidade. (BRUCKSCHEN et al, 2008)

Figura 8- Categorias do HAREM.

Fonte: Bruckschenet et al, 2008

As relações propostas pela trilha ReRelEM, pioneira no Segundo HAREM, são: *ident* (identidade) , *inclui*, *ocorre*_em e *outra*. A seguir Bruckschenet nos explica as relações mencionadas.

Figura 9 – Exemplo de relações



Fonte: Bruckschenet et al, 2008

Primeiramente temos a relação *ident*, que é atribuída às entidades mencionadas que pertencem à mesma categoria semântica, ou seja, “referem-se à mesma coisa”. Neste caso como mostra a figura 9, ela abrange todas classes. Temos por exemplo a palavra “Salsicha”, que se refere ao personagem do desenho do Scooby-Doo que se encaixa na classe de Pessoa e podemos ter “Salsicha Perdigão” que se encaixa no termo Coisa, então neste caso não seria válido estabelecer a relação *ident*.(BRUCKSCHEN et al, 2008).

Já a segunda relação proposta Bruckschen que “*inclui”* refere-se a EMs das classes LOCAL, ORGANIZAÇÃO, ABSTRAÇÃO, TEMPO e COISA. Estas podem ter relação de inclusão entre si (uma EM da classe ORGANIZAÇÃO pode incluir apenas outra da mesma classe, por exemplo). (BRUCKSCHEN et al, 2008). Um exemplo seria a inclusão de um lugar em outro, como “Vale do Taquari” inclui “Santa Cruz do Sul”, que inclui “Bairro Higienópolis”.

A terceira relação proposta é chamada de *ocorre_em*, que cria relação entre as classes EVENTO/ORGANIZAÇÃO e de LOCAL. (BRUCKSCHEN et al, 2008). Como por exemplo “Batalha de Stalingrado” ocorre_em “Rússia”. “Festival Wacken” ocorre_em “Alemanha”.

Finalmente, a relação *outra* seria atribuída a todas as EMs que possuíssem alguma relação diferente das três anteriores (como em “José outra Pedro”, sendo José pai de Pedro, por exemplo). (BRUCKSCHENET et al, 2008).

2.4.2.4 Padrões de Hearst.

Em 2015, Machado desenvolveu um protótipo que extrai relações hiponímicas de *corpus* de textos em língua portuguesa. O protótipo foi executado sobre *corpus* de textos e os resultados obtidos foram analisados tanto por fonte de referência como por grupos de regras. Relações hiponímicas são geralmente representadas por “é_um”. Isto se deve a expressarem relações entre instâncias e classes, como também entre classes. Este tipo de relação é considerada como uma das bases para construção de ontologias.

Neste trabalho proposto por Machado, foi desenvolvida uma heurística baseada nos padrões de Hearst, que é uma lista com seis padrões léxico-sintáticos que identificam relações em textos escritos na língua inglesa. (MACHADO, 2015).

Tabela 1 – Padrões léxico-sintáticos de Hearst

	Padrão Original
1	NP *such as* {(NP,)*(*or*|*and*)} NP
2	*such* NP *as* {(NP,)*(*or*|*and*)} NP
3	NP {, NP}* {,} or other NP
4	NP {, NP}* {,} and other NP
5	NP {,} including {NP,}*{or|and} NP
6	NP {,} especially {NP,}*{or|and} NP

Fonte: Fetzer, 2013.

Um exemplo aplicado dos padrões de Hearst, e retirado de seu trabalho (Hearst, 1992), pode ser visto: "*...most European countries, especially France, England and Spain.*". Pode ser observado que esse trecho, se utilizado o padrão 6 da Tabela 1 (NP {,} especially {NP,}*{or|and} NP), resultará nas seguintes relações: ("*France*", "*European country*"); ("*England*", "*European country*"); ("*Spain*", "*European country*"). (FETZER, 2013)

Outros trabalhos em língua portuguesa adaptaram estes padrões para o português, e partir destas adaptações, Machado criou sue próprio conjunto de regras para a extração de relações, demonstradas na Tabela 2.

Tabela 2 – Padrões léxico-sintáticos de Machado

1	SN(,)? como (SN ,)*(SN (e|ou))*SN
2	SN(,)? ta(is|l) como (SN ,)*(SN (e|ou))*SN
3	SN(,)? incluindo (SN ,)*(SN (e|ou))*SN
4	SN(,)? especialmente (SN ,)*(SN (e|ou))*SN
5	(SN (ou|e|,))*<outr(a|o)(s)? sn>
6	<... tipo(s)? de sn> : (SN ,)*(SN (e|ou))*SN
7	SN(,| é| são| foram)? chamad(o|a|os|as)(de)? (SN ,)*(SN (e|ou))*SN
8	SN((,)? também)?(,|é|são|foram)? conhecid(o|a|os|as) como (SN ,)*SN (e|ou))*SN"
9	(SN (ou|e|,))*< (qualquer|quaisquer) outr(a|o)(s)? sn>
10.A	SN é < (o|a) sn>
10.B	SN é < (um|uma) sn>
11	SN são SN

Aqui Machado(2015), nos explica como devemos interpretar a tabela,

> “O formato das regras é muito semelhante à sintaxe de expressões regulares, e os sintagmas nominais são representados por “SN”. Assim como nas expressões regulares, os parênteses são utilizados para agrupar as expressões, enquanto que o “*” representa que uma expressão pode ocorrer nenhuma ou mais vezes. A interrogação significa nenhuma ou uma repetição. Outro símbolo comumente utilizado é o “|” que representa um “ou exclusivo”. Também foi utilizada a notação “<sn PALAVRA-CHAVE sn>” para identificar a ocorrência de uma palavra-chave que está contida dentro de um *chunk*. Como um Sintagma Nominal pode ser formado por outros SNs, o símbolo “sn” (minúsculo) foi empregado para identificar um Sintagma Nominal que é um dos elemento de um “SN”. Caso a palavra-chave encontre-se diretamente após o símbolo “<” ou antes do símbolo “>”, significa que ela é respectivamente a primeira ou a última palavra do *chunk*, como está exemplificado em: “<outros sn>” (a palavra-chave é representada por “outros”). (MACHADO, 2015, p.35)

3 **TRABALHOS RELACIONADOS**

Conforme se pode observar da leitura atenta do texto, o presente trabalho não é um fim em si mesmo, de modo que busca a implementação de uma função em ferramentas já existentes.

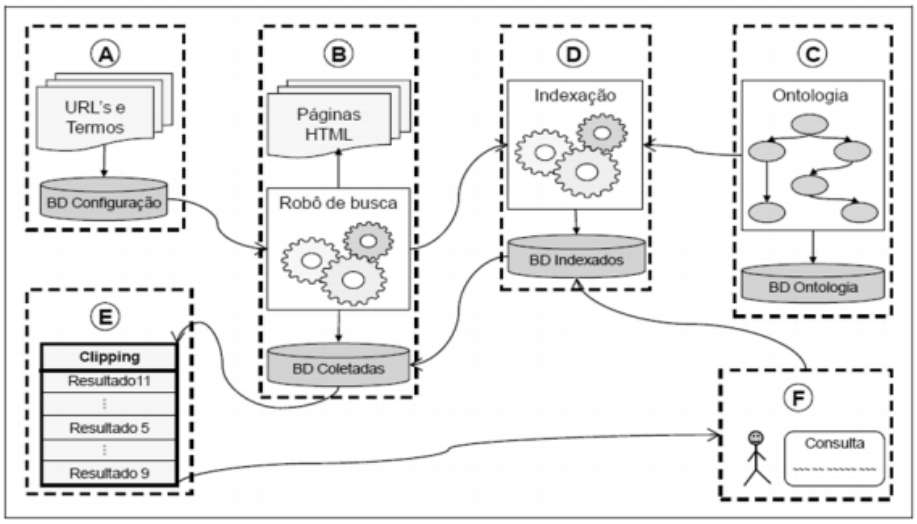
Assim, é imprescindível a análise das pesquisas e trabalhos que originaram o programa no qual se pretende desenvolver o módulo aqui proposto.

Na UNISC, temos dois trabalhos de notória relevância, cuja menção é imperiosa, são eles: OntoClipping1 e o OntoClipping2. O OntoClipping1 foi desenvolvido por Claudio Omar Correa Carvalho Junior, sendo “uma ferramenta de clipagem eletrônica que utiliza uma ontologia como forma de representação do conhecimento, juntamente com técnicas de recuperação da informação aliadas a um motor de busca de páginas web”.(SCHUSTER, 2013, p4.).

Então tem-se a evolução dele, que é o software OntoClipping2, desenvolvido por Roberto Antonio Schuster Filho, no qual o foco foi “o aprimoramento da ferramenta OntoClipping, através da avaliação temporal e qualitativa das informações coletadas no processo de clipping, a partir de critérios específicos.”(SCHUSTER, 2013, p.73).

Em outras palavras, o software comentado anteriormente procura criar e alimentar uma ontologia através de busca e coleta de informações em textos, tanto online quanto offline, através de algoritmos que levam em consideração várias características textuais e jornalísticas conforme se demonstrará na figura abaixo:

Figura 10 - Fluxo do OntoClipping



Fonte: Fetzer, 2013

> "A figura 10 ilustra o fluxo do OntoClipping, que possui as seguintes etapas: (A) cadastros de sites relevantes para o processo de clipping; (B) processo de clipping, ou seja, o robô de busca recupera informações a partir dos sites raízes, baseadas nas informações da etapa A; (C) criação de uma ontologia a fim de representar o conhecimento conforme as necessidades reais; (D) indexação do conteúdo coletado pelo robô de busca; (E) busca do conteúdo indexado, feito por um algoritmo próprio, utilizando o conteúdo armazenado pelas etapas anteriores B e C e pela ontologia da etapa D; (F) consulta do usuário em uma interface gráfica."(FETZER, 2013).

Em andamento, está sendo desenvolvido outro trabalho de conclusão que visa à identificação de termos próprios em bases de dados textuais, neste caso, sobre a base de dados do OntoClipping. Este trabalho de conclusão está na primeira fase e está sendo desenvolvido pelo estudante Eduardo Wegner.

Já na área de PLN, Batista et. Al, propôs uma abordagem de supervisão distante para a classificação de relações extraídas de textos escritos em português, suportada por dados extraídos da Wikipédia e da DBPédia, e baseada na medição de similaridade entre as relações a classificar e relações armazenadas numa base de dados de relações de exemplo (BATISTA et al., 2013).

Essas relações são encontradas a partir do algoritmo kNN, (heurísticas do vizinhos mais próximos) em conjunto com regras de padrões gramáticos. Alguns critérios interessantes que Batista leva em consideração na parte de reconhecimento destas relações entre as identidades são:

1. A categoria semântica das duas entidades (ex. pessoa, local ou organização).
2. As sequência de palavras entre as entidades.
3. O número de palavras entre as entidades.

> Uma relação entre duas entidades é geralmente expressa utilizando apenas palavras que aparecem em um de três padrões básicos, nomeadamente antes-e-entre (i.e., palavras antes e entre a duas entidades envolvidas na relação), entre (i.e., apenas as palavras entre as duas entidades),e entre-e-depois (i.e., palavras que ocorrem entre e depois das duas entidades)., (BATISTA et al, 2013, apud BUNESCU E MOONEY, 2005).

Nas categorias *antes* e *depois*, foi considerado uma janela de 3 palavras de distância das entidades chaves para a identificação de relações.

Outro trabalho importante foi o projeto iniciado no SINTEF Telecom and Informatics, em Oslo Noruega, que culminou com o evento denominado HAREM, comentado na seção 2.4.2.3, que aconteceu em 2008 e teve como uma das tarefas o reconhecimento de relações semânticas entre EMs, nomeado de ReRelEM.

De acordo com Santos, Namede e Baptista (2010), os resultados no ReRelEM, principalmente na área de reconhecimento de relações de *família*, não foram muito satisfatórios, então eles propuseram uma solução que trabalha somente nesta categoria, onde foram englobadas todas as possíveis relações entre indivíduos de uma família. Este algoritmo foi implementado e adicionado ao sistema XIP (*Xerox Incremental Parsing*), que é "um analisador cujo primeiro objetivo é a extração de dependências sintáticas. O analisador processa documentos em formato de texto ou XML e produz como saída uma representação sintática do conteúdo do documento." (HAGÈGE, BAPTISTA e MAMEDE, 2008, p.262). De acordo com os autores do trabalho, os resultados ainda foram insatisfatórios.

Outra proposta de importância, na descoberta de relações a partir do processamento de linguagens foi o trabalho de Machado, que foi comentado na seção 4.2.4, em conjunto aos padrões de Hearst.

Dentro da universidade também tem-se o trabalho de Fezter, que trabalha com a alimentação de uma ontologia a partir de analise de textos, este trabalho está mais bem comentado na seção 4.1.

Fora do país, também temos outros trabalhos relacionados que trabalham com a língua inglesa, como por exemplo, os sistemas TEXTRUNNER, WOE e REVERB, "que são considerados programas de 'Open Information Extraction (IE)'. Open IE são sistemas que trabalham com a análise de um corpus, extraindo relações sem exigir de um treinamento anterior em cima de dados para realizar a operação. (ETZIONI ET AL, 2011). Sendo o ultimo deles, o REVERB, um sistema que faz identificação de relações a partir de regras léxicas, sintáticas e em conjunto com outras heurísticas.

4 SOLUÇÃO DESENVOLVIDA

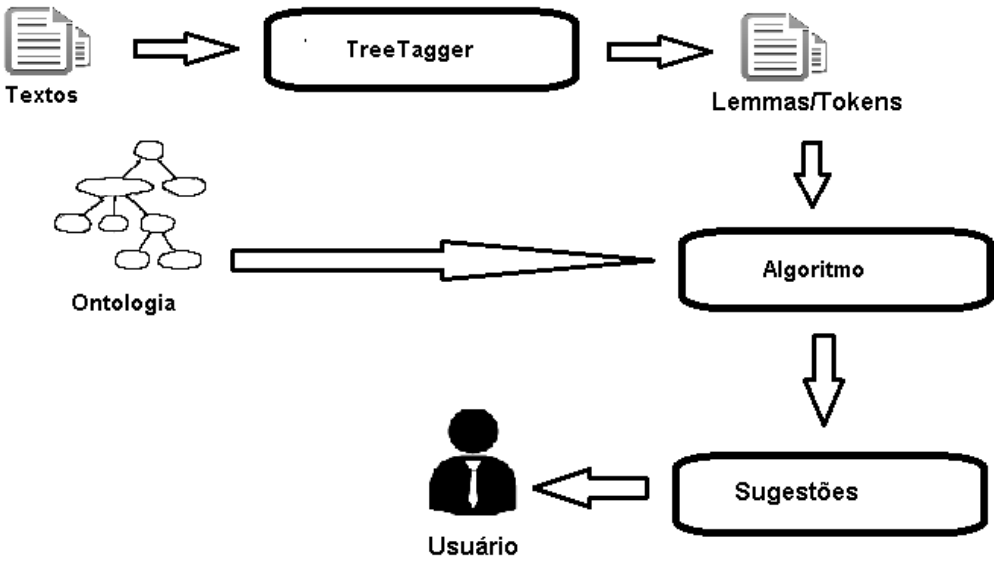
Neste capítulo é descrito a solução implementada para o trabalho realizado. São abordadas as utilidades e funcionamento das telas do sistema, o funcionamento das regras de reconhecimento de relações e o algoritmo que as aplica, as adições que foram feitas ao banco de dados e os testes realizados para a validação do trabalho.

4.1 Visão geral

A solução proposta consiste em um sistema que **auxilia** o usuário ao procurar relações entre termos de uma ontologia, preferencialmente, previamente criada, a partir de textos importados para dentro do banco de dados. Estes textos também podem ser inseridos manualmente pelo sistema.

Estes textos, quando analisados, primeiramente passam pelo processo de etiquetagem realizado pelo TreeTagger, que é explicado na seção 4.2.1. O resultado do TreeTagger é uma lista composta por *tokens* e *lemmas*. Depois o sistema trabalha com estes resultados usando um conjunto de regras parametrizáveis pelo usuário [seção 4.2.3], para finalmente oferecer sugestões de relações entre elementos da ontologia que se encontram no texto previamente processado.

Figura 11 – Fluxo



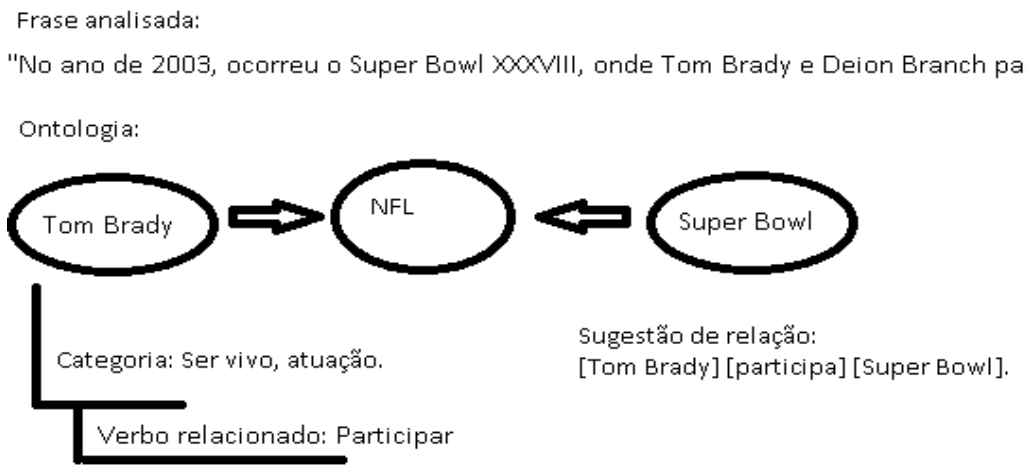
Fonte: Autor.

A solução aqui desenvolvida utiliza principalmente o conceito de associação de *categorias* aos termos que se encontram na ontologia e também aos termos novos que forem inseridos de acordo com o uso do software. Foram incluídos serão incluídos cinco tipos de categorias: SER VIVO, ORGANIZAÇÃO, LOCAL, COISA/OBJETO, OUTROS, que são baseadas nas categorias que foram usadas no HAREM, comentadas na seção 2.4.2.3.

Como mencionado anteriormente, existe uma listagem de verbos no sistema e estes verbos são associados às categorias para realizar o reconhecimento das relações entre os elementos.

Por exemplo, se existir um elemento chamado “SuperBowl”, que refere-se a partida final da liga de futebol americano, ele poderia se encaixar tanto na categoria de EVENTO como na categoria de ATUAÇÃO, e outro elemento chamado “Tom Brady” que se encaixaria na categoria “SER VIVO”. Dentro da categoria ATUAÇÃO, teríamos parametrizado o verbo “jogar”. Então quando o algoritmo encontrar o verbo “jogar” no texto e aplicar as regras de validações [seção 4.2.3] com sucesso, ele vai considerar esta relação como uma possível candidata.

Figura 12 – Exemplo de análise



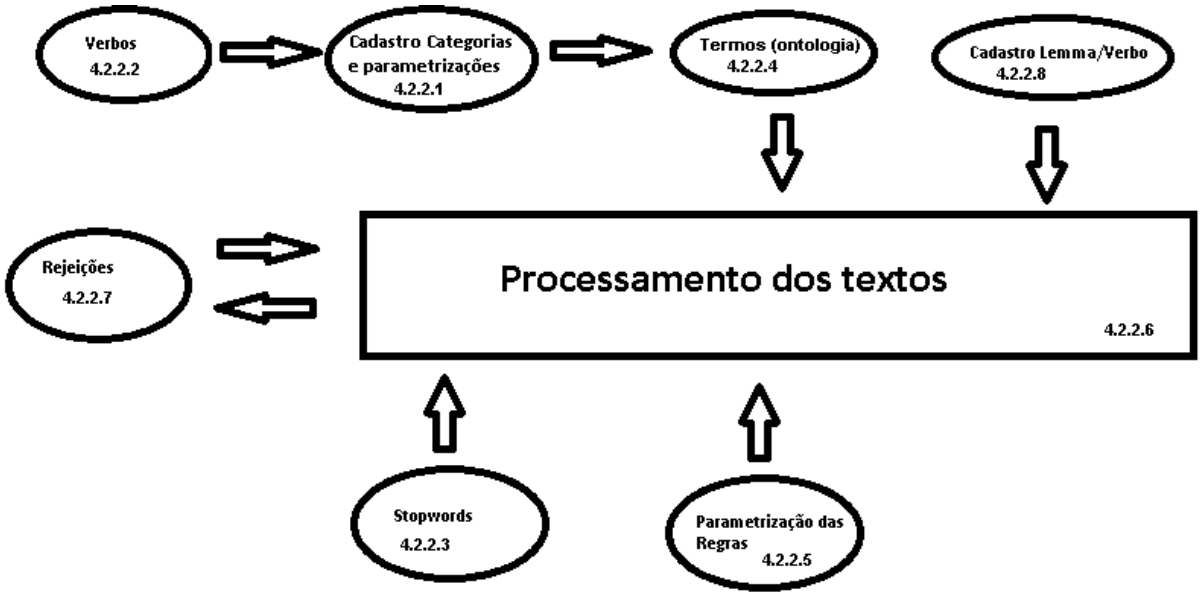
Fonte: Autor.

Importante salientar que a maioria das funcionalidades do sistema são parametrizáveis, tanto a criação de novas categorias, a associação de verbos a categorias novas/existentes, associação de categorias aos termos da ontologia e a parametrização das regras utilizadas para as validações [seção 4.2.3] podem, e na maior parte das vezes devem, ser realizadas pelo usuário do sistema. Todas estas parametrizações são realizadas a partir de telas do sistema, que são comentadas na seção 4.2.2.

Considerando que ontologias, extração de informações e as classificações que o sistema permite realizar são conceitos subjetivos, esta parametrizações concedem ao usuário uma

maior flexibilidade para ajustar o sistema aos seus próprios objetivos. Em outras palavras, as parametrizações realizadas por um usuário podem não ser adequadas para a análise de textos de um outro usuário.

Figura 13 – Diagrama de passos



Fonte: Autor.

Para a construção e alimentação da ontologia pode se usar a tela de listagem de termos, apresentada na seção 4.2.2.4, ou usar a aplicação desenvolvida por Fetzer (2013), desde que a a aplicação esteja trabalhando com o mesmo banco de dados da aplicação aqui desenvolvida.

O sistema utiliza uma expansão do banco de dados do Trabalho de conclusão do aluno Felipe Fetzer (2013), que também tem grandes semelhanças com outros trabalhos desenvolvidos dentro da UNISC, favorecendo uma possível integração futura entre os sistemas, principalmente em relação à alimentação da ontologia e de textos para análise, que são áreas que o sistema carece e que já foram abordadas por outros trabalhos.

A aplicação foi desenvolvida em JAVA WEB utilizando o servidor TOMCAT7 e o banco de dados utilizado é o MYSQL. Este ambiente de desenvolvimento foi selecionado devido a sua prévia utilização no trabalho de Fetzer (2013).

4.2 Funcionalidades

Nesta seção são apresentadas as funcionalidades do sistema.

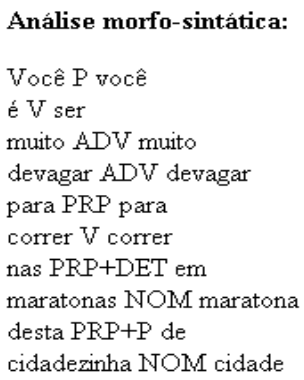
4.2.1 TreeTagger

O TreeTagger é uma ferramenta que realiza analises morfossintáticas de textos. Seu processo é simples. Usa-se um texto como entrada, e a saída desse processamento é uma lista das palavras e caracteres especiais desse texto com três atributos: (1) a palavra em si; (2) o tipo da palavra, também chamado de etiqueta; (3) a palavra em seu radical e/ou singular, também chamado de *lemma*.

As etiquetas utilizadas para a língua portuguesa são as seguintes: V para verbos; NOM para substantivos; ADV para advérbios; ADJ para adjuntos; SENT para pontuação; VIRG para virgulas; CONJ para conjunção; PRP para preposições; CARD para diversos.

Para o sistema, as etiquetas V e NOM são as mais importantes, porque é a partir delas que o algoritmo tentará realizar o reconhecimento de relações. O exemplo na figura 14 demonstra o resultado de análise da frase “Você é muito devagar para correr nas maratonas desta cidadezinha”. Exemplo já demonstrado na seção 2.1.1.1.

Figura 14 – Exemplo TreeTagger



Fonte: Autor.

Além da funcionalidade de realizar análises em textos, o TreeTagger também forneceu um grande corpus de verbos que foram importados para dentro do sistema a partir de uma função implementada separadamente do sistema. Estes verbos podem ser encontrados na tela de listagem de verbos, que é apresentada na seção 4.2.2.2.

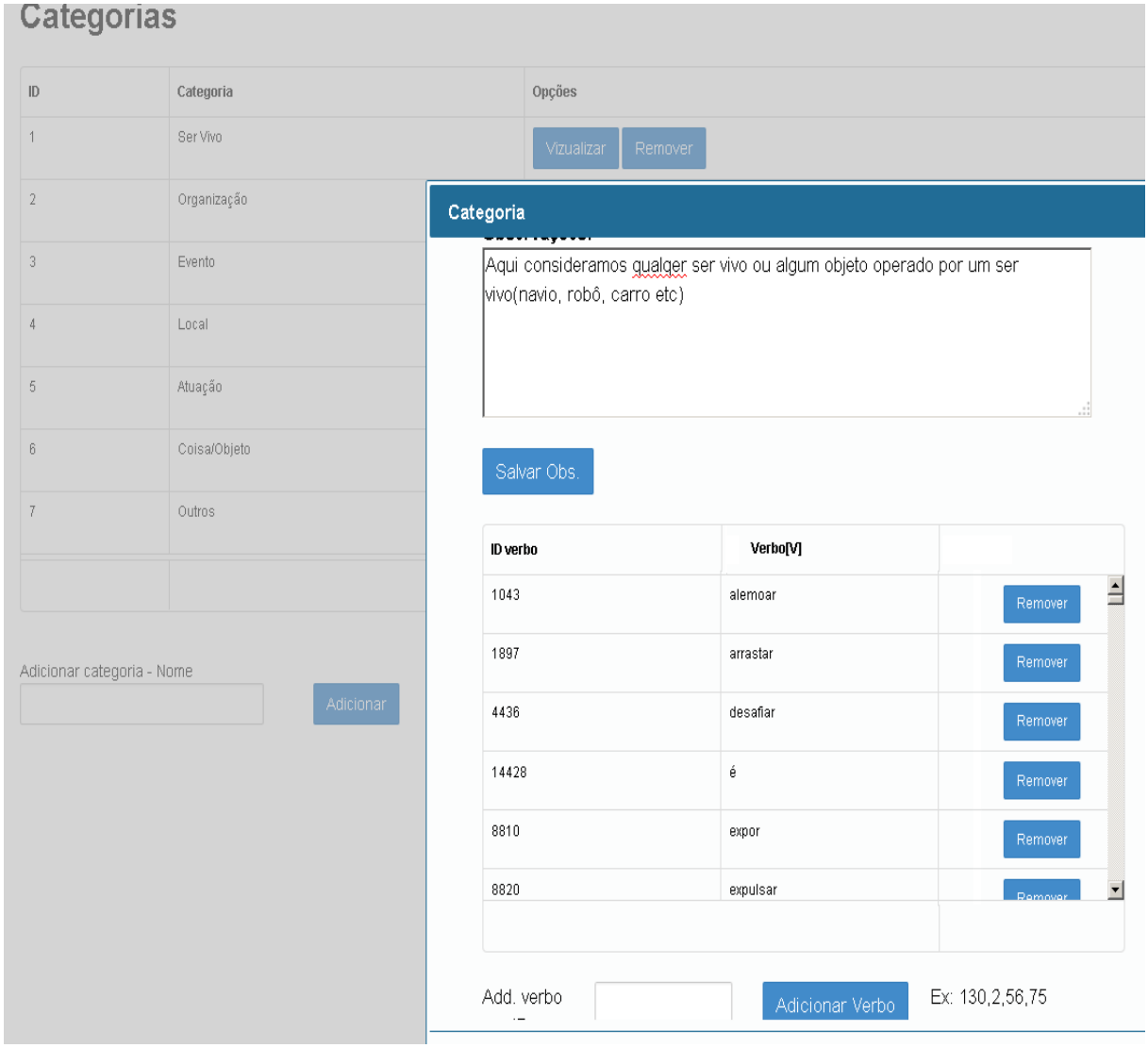
4.2.2 Telas do sistema

Nesta seção são abordadas as telas que se encontram no sistema, explicando suas funcionalidades.

4.2.2.1 Tela de categorias

Nesta tela tem-se a relação de categorias cadastradas no sistema, assim como a opção de adicionar categorias novas, remover e visualizar informações de categorias existentes.

Figura 15 – Tela de categorias



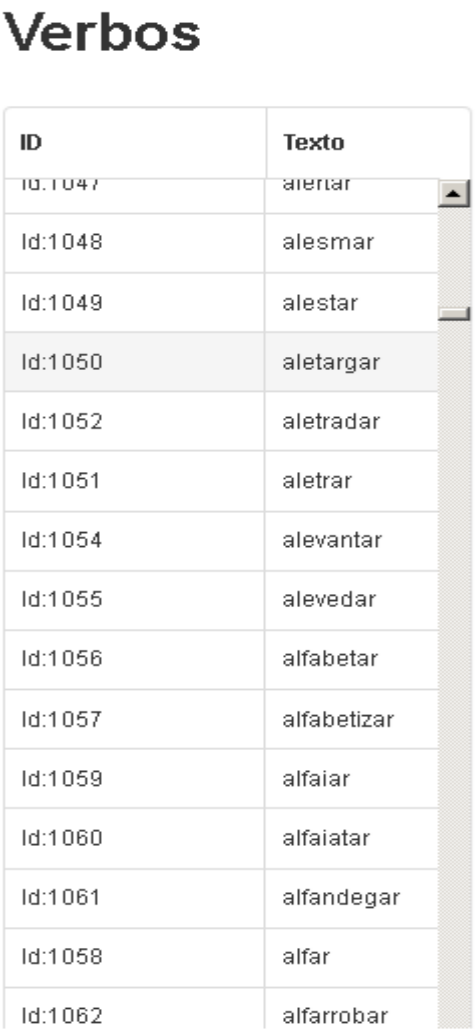
Fonte: Autor.

A figura 14 mostra as informações da categoria “Ser vivo”. No *dialog* de informações apresenta-se a opção de escrever observações sobre a categoria e de adicionar e remover verbos que fazem estão relacionados a esta categoria. A adição de verbos é feita através de seu “id”, podendo-se adicionar mais de um ao mesmo tempo, separando-os com vírgula. Exemplo: 130, 2, 56, 75.

4.2.2.2 Tela de verbos

Nesta tela é mostrada a listagem de todos os verbos importados do corpus do TreeTagger e suas respectivas “ids”.

Figura 16 – Tela de verbos

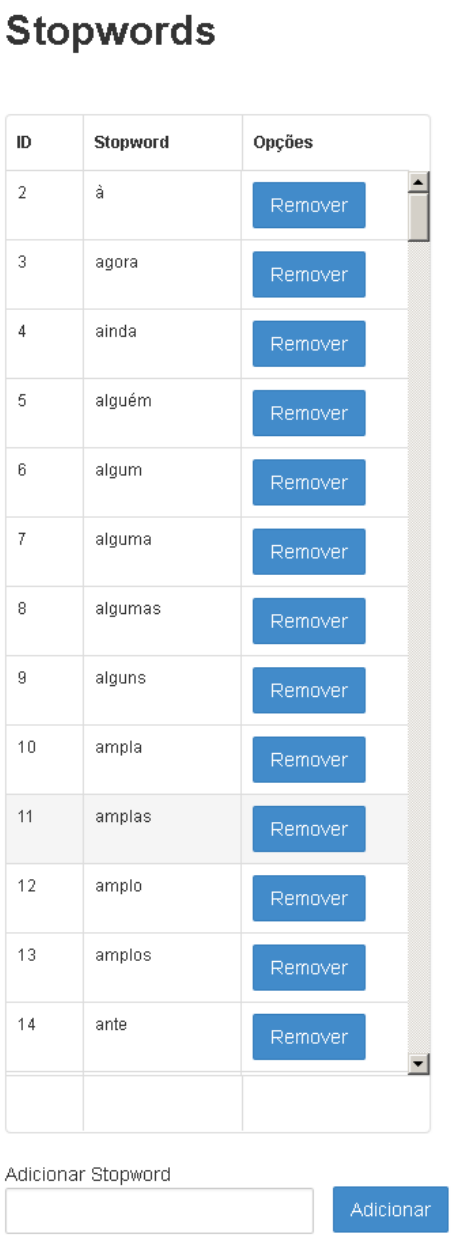


Fonte: Autor.

4.2.2.3 Tela de stopwords

Nesta tela encontra-se a listagem das *stopwords* usadas pela regra de Stopwords, explicada na seção 4.2.3.2, assim como a opção de adicionar novas palavras e remover outras já cadastradas no sistema. Esta base de stopwords foi previamente importada/criada por Fetzer(2013).

Figura 17 – Tela de listagem de stopwords



Fonte: Autor

4.2.2.4 Tela de termos da ontologia

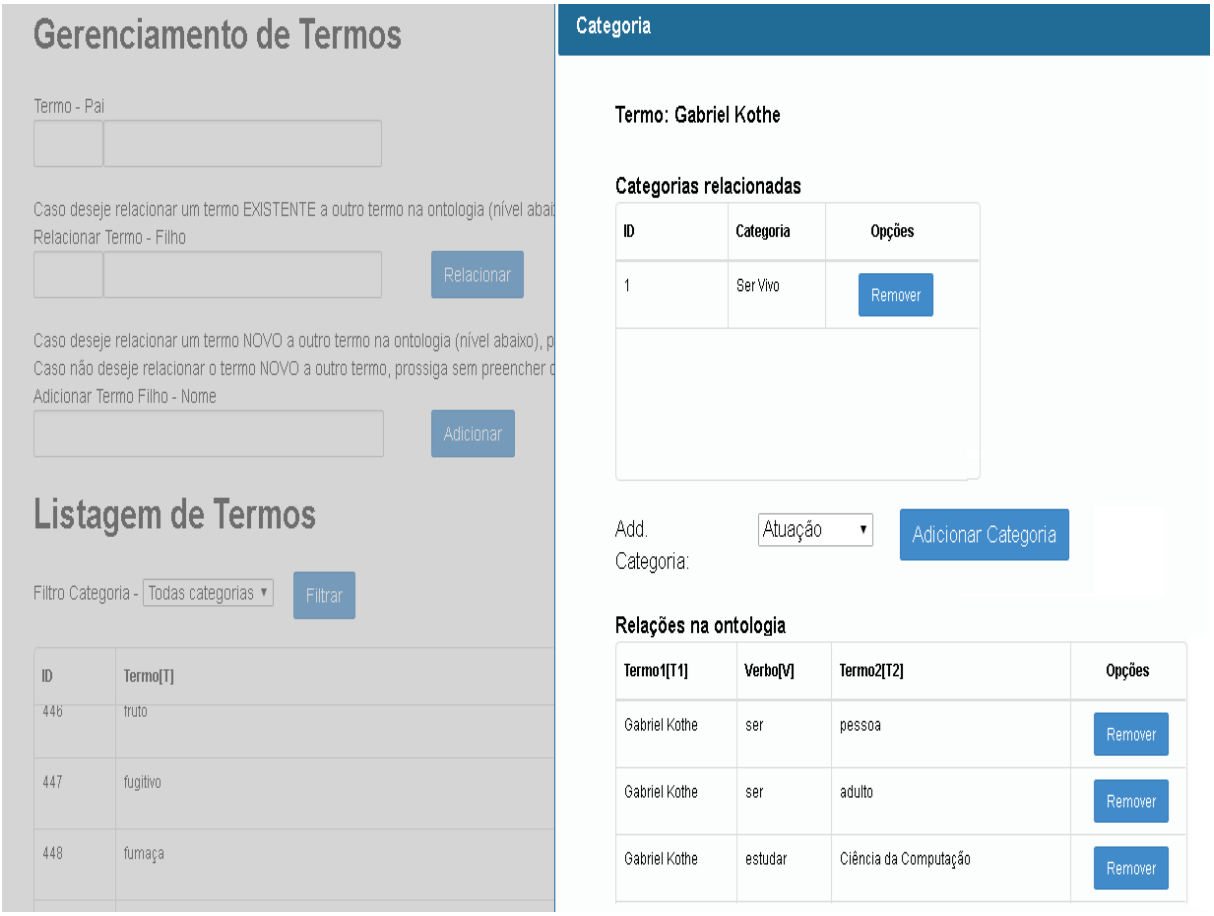
Nesta tela encontra-se uma relação de todos os termos que se encontram na ontologia, assim como opção de adicionar novos termos, remover e visualizar informações de termos existentes. Para adicionar um termo, usa-se o campo “adicionar termo filho – nome”, toda vez

que um termo for adicionado sem termo pai ele automaticamente é relacionado ao termo de "id" 1, que o sistema considera ser o "topo da ontologia".

Para adicionar um termo com relação a outro termo deve se preencher o campo "termo - pai". Também existe a opção de relacionar dois termos já existentes na ontologia a partir dos campos "termo - pai" e "relacionar termo – filho".

Se o usuário deseja encontrar o termo recém adicionado na listagem, o usuário deve selecionar o filtro de categoria "sem categorias". Para este termo ser encontrado pelo sistema em textos ***deve-se*** relaciona-lo a uma categoria que contenha verbos parametrizados.

Figura 18 – Tela de listagem de termos da ontologia



Fonte: Autor

No *dialog* de informações encontra-se uma listagem das categorias relacionadas ao termo e também a opção de remover e adicionar categorias a este termo. No mesmo *dialog*, tem-se também uma lista das relações encontradas pelo sistema e aceitas pelo usuário junto e a opção de remoção das mesmas.

4.2.2.5 Tela de parâmetros

Esta tela apresenta os parâmetros do algoritmo de identificação de relações e o campo para informar a localização do TreeTagger dentro do servidor, que é necessário para que o sistema possa realizar as análises das palavras. As regras são explicadas na seção de funcionamento do algoritmo [seção 4.2.3].

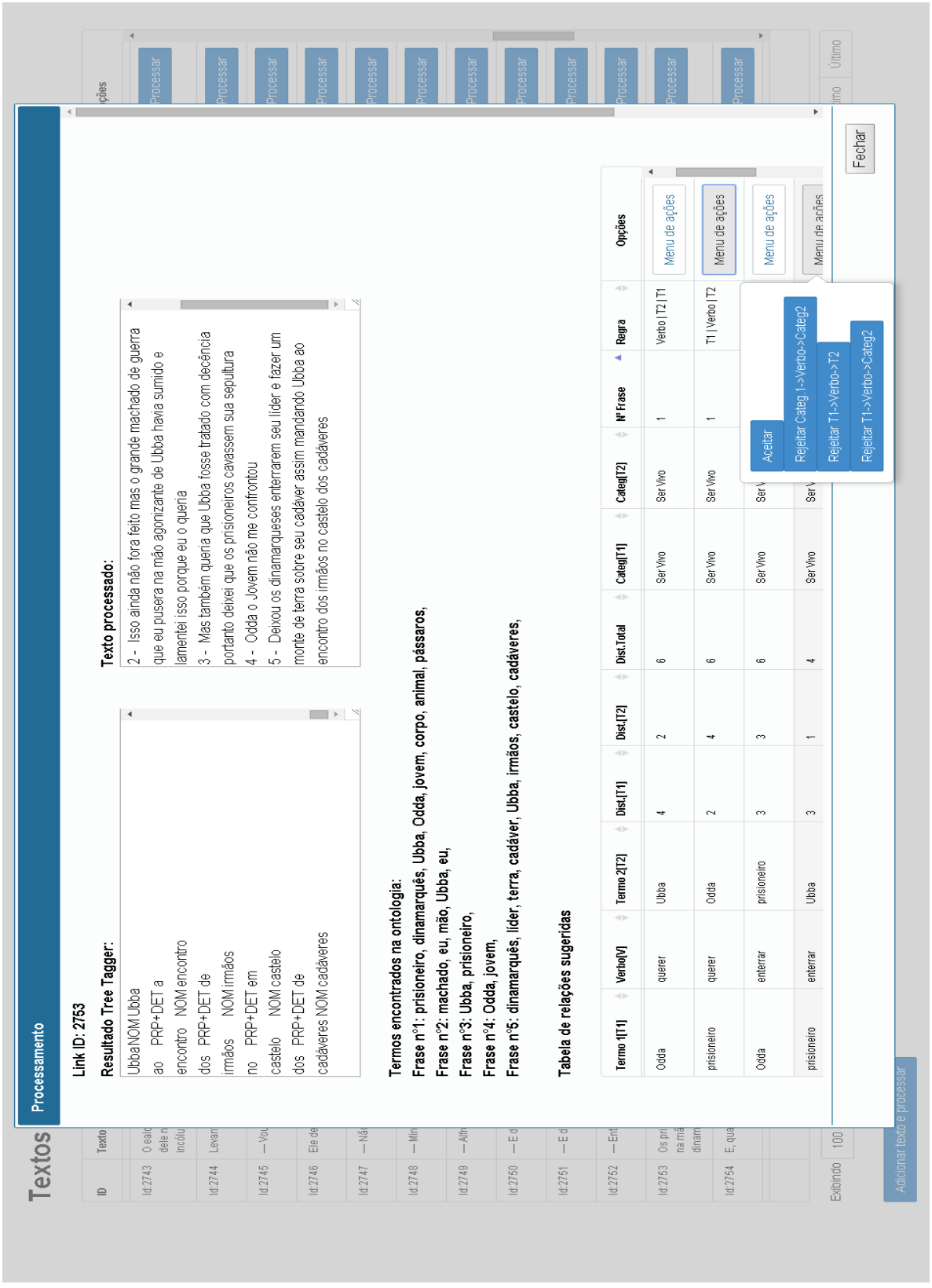
Figura 19 – Tela de parâmetros

Fonte: Autor

4.2.2.6 Tela de listagens de textos e processamento

Esta tela contém uma listagem de todos os textos que se encontram no banco de dados e para cada texto temos a opção de processamento que nos traz o *dialog* mostrado na figura 18. Também temos a opção de adicionar textos manualmente.

Figura 20 – Tela de listagem de textos e processamento



Fonte: Autor

Neste *dialog* tem-se os seguintes quadros de informações:

Resultado TreeTagger - Aqui o sistema nos retorna o processamento das palavras que o TreeTagger realizou com o texto.

Texto processado - Neste campo encontra-se o texto que foi processado e dividido em frases numeradas para uma melhor identificação nos outros quadros de informações.

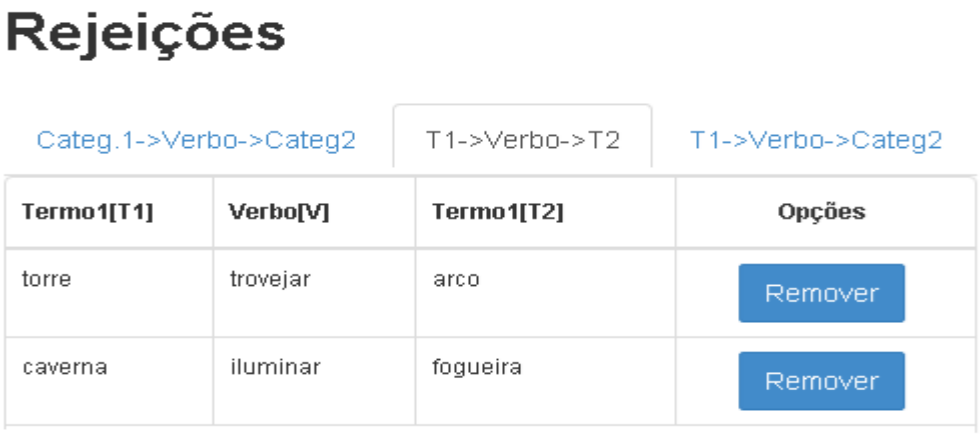
Termos encontrados na ontologia - Aqui é apresentado os termos encontrados no texto que se encontram cadastrados no sistema e também em qual frase ele foi encontrado. Se um termo aparecer mais de uma vez na mesma frase, ele será listado duas vezes.

Tabela de relações sugeridas – Aqui temos as relações sugeridas pelo sistema. Além dos campos de termos e verbo, que representam a relação sugerida, temos: os campos de distância que são referentes à regra de distância [seção 4.2.3.1], as categorias relacionadas aos termos encontrados, a frase onde a sugestão foi encontrada, a regra de OTV [seção 4.2.3.3] e as ações que podem ser realizadas. As ações são as opções de rejeição referente às regras de rejeição [seção 4.2.3.6] e a confirmação da sugestão.

4.2.2.7 Tela de rejeições

Nesta tela encontram-se a listagem das rejeições informadas pelo usuário durante a etapa de processamento com a opção de remoção, caso o usuário considerar que a rejeição não é mais válida. Estas rejeições estão organizadas em três abas separadas, uma para cada regra de rejeição [seção 4.2.3.6].

Figura 21 – Tela de rejeições

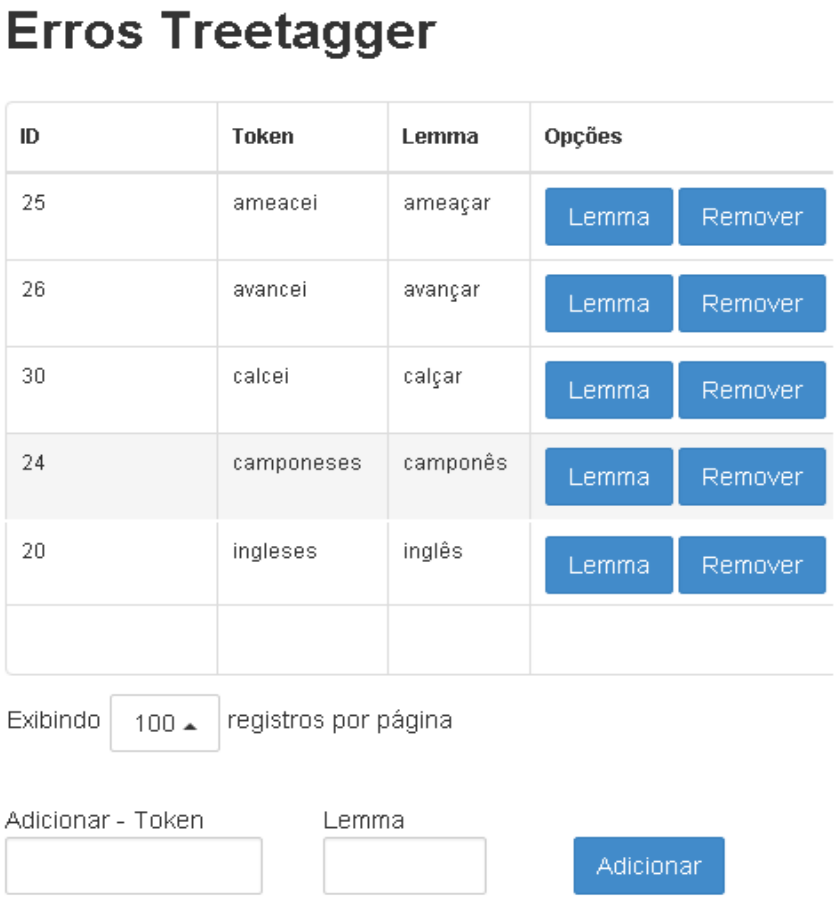


Fonte: Autor

4.2.2.8 Tela de erros do TreeTagger

Durante o processamento de textos, em alguns casos o TreeTagger retornou *lemmas* com problemas de *enconding* que causaram erros no sistema. Para solucionar este problema, foi criada uma tabela onde temos a relação das palavras e *lemmas* que causaram erro, para que o sistema possa consultar o *lemma* correto. Toda vez que acontecer algum erro durante o processamento, o sistema realizara a tentativa de alertar o usuário e adicionar automaticamente a palavra que causou o problema na tabela de erros, para que assim o usuário possa inserir manualmente o *lemma* correto. O *lemma* correto pode ser consultado no site http://gramatica.usc.es/~gamallo/php/tagger/TaggerPT.php. Caso o erro retorne com alguma mensagem incompreensível, o usuário deve acessar o log do Tomcat e procurar o resultado do TreeTagger para aquele processamento e identificar a palavra com problema. Por exemplo: “inglesa ADJ ingl?s”. O carácter “?” representa um problema de *encoding*.

Figura 22 – Tela de erros do TreeTagger



Fonte: Autor

4.2.2.9 Tela da ontologia

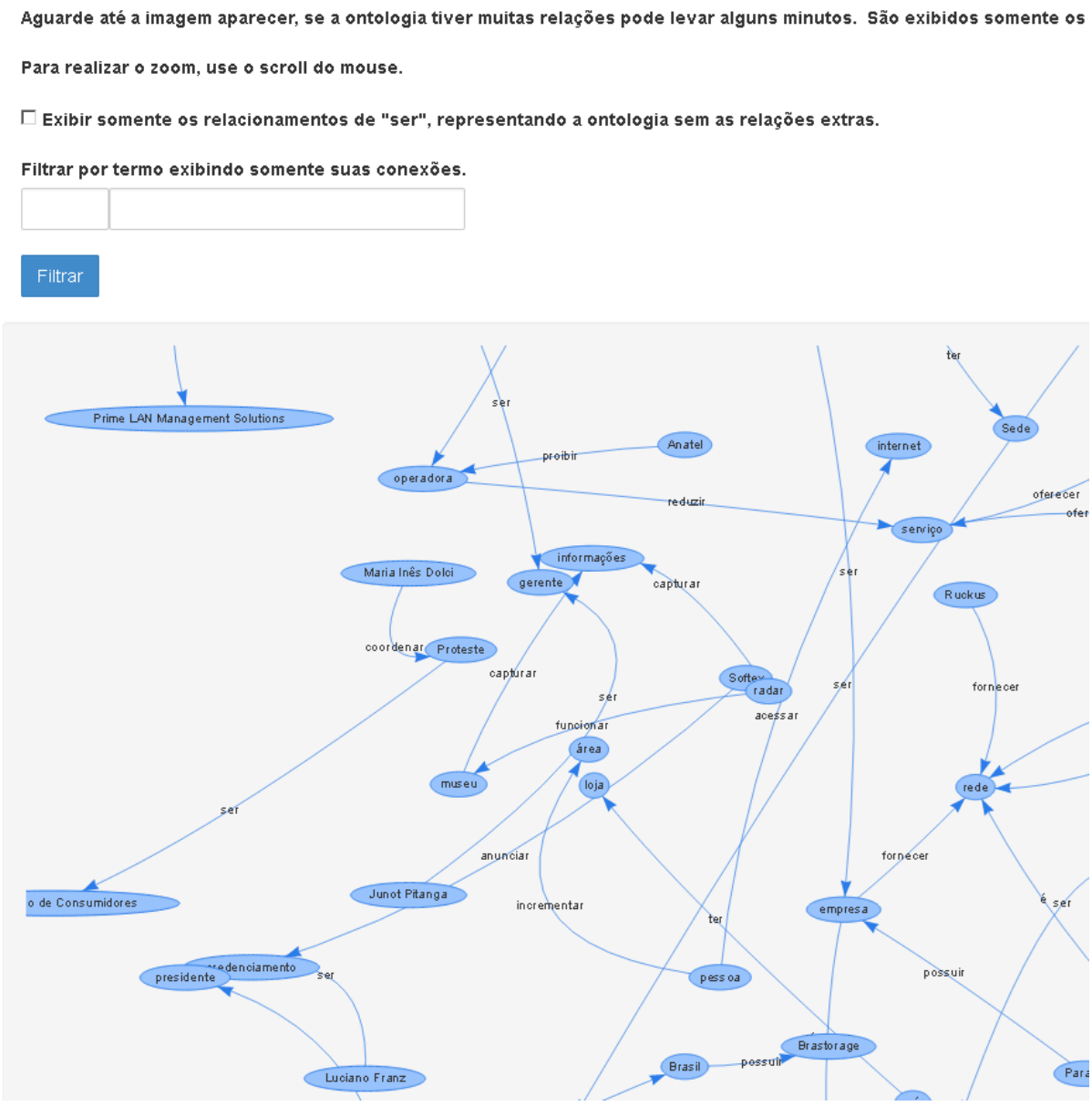
Esta tela mostra graficamente as relações encontradas e aceitas entre os termos da ontologia. Nela temos duas opções de filtros:

1º - “Exibir somente os relacionamentos de "ser", representando a ontologia sem as relações extras”.

2º - Filtrar por termo exibindo somente suas conexões.

Para a visualização da ontologia foi usado o plugin Vis.js, encontrado em http://visjs.org/# .

Figura 23 – Tela da ontologia



Fonte: Autor.

4.2.2.10 Tela de lemmas/verbos

Nesta tela encontra-se a parametrização de *lemmas/verbos*, que se refere à regra de lemma/verbos [seção 4.2.3.8].

Para realizar a parametrização, deve-se escrever o lemma desejado no campo "Adicionar - lemma", e o relaciona-lo com o verbo desejado no campo "verbo". O verbo pode ser pesquisado via uma função de *autocomplete* no campo maior (a direita) ou diretamente pelo seu ID no campo menor (a esquerda). Também tem-se a opção de remover parametrizações desejadas.

Figura 24 – Tela de erros lemmas/verbos

Fonte: Autor.

4.2.3 Regras de validações para sugestão de relações

Nesta seção, são comentadas as regras parametrizáveis que são utilizadas pelo algoritmo desenvolvido para identificar as possíveis relações.

4.2.3.1 Regra de distância

Nesta regra o sistema considera os parâmetros de Distância Máxima Individual (DMI) e Distância Máxima Conjunta (DMC). No contexto geral, a distância significa a quantidade de palavras entre os termos e verbos, a distância mínima sempre será 1. Deste modo pode se presumir que quanto menor a distancia, maior a probabilidade de uma sugestão ser válida.

A DMI nos permite parametrizar a distância máxima entre um termo e o verbo encontrado. Desta forma, diminuindo a quantidade de relações que provavelmente serão inválidas.

Exemplo: “João, mesmo cansado, foi na padaria”. Distância de João[T1] até foi[V] é de 3 e a distância de foi[V] até padaria[T2] é 2. Caso a distância encontrada entre os termos e o verbo seja maior que o parâmetro de DMI, a relação não será sugerida.

A DMC nos permite parametrizar a distância máxima somada das distâncias individuais. Levando em consideração o exemplo anterior, a distância conjunta seria 5. Porém se tivéssemos parametrizado a DMC com o valor 4, a DMI com valor 3. A relação anterior não seria sugerida pelo sistema devido ao valor máximo. Esta regra ajuda a filtrar termos e verbos muito distantes um dos outros.

4.2.3.2 Regra das stopwords

Esta regra já implementada anteriormente por Fetzer(2013), se selecionada, remove um conjunto de palavras do texto, que podem ser parametrizadas através do sistema. Estas palavras geralmente não nos trazem informações úteis em relação ao encontro de relações, podendo assim, serem filtradas do texto.

4.2.3.3 Regra de ordem de termos e verbos [OTV]

Esta regra permite parametrizar 6 opções de ordens de Verbos[V] e Termos[T1/T2] que devem ser consideradas no processamento das relações. São elas:

1- T1 | Verbo | T2
2- T2 | Verbo | T1
3- Verbo | T1 | T2
4- Verbo | T2 | T1
5- T1 | T2 | Verbo
6- T2 | T1 | Verbo

Exemplo: “João, mesmo cansado, foi na padaria”, a relação João[T1] foi[V] Padaria[T2] se encaixaria na primeira opção. Outro caso, “A caverna era iluminada pela fogueira”, seria incorreto afirmar que caverna[T1] ilumina[V] fogueira[T2] que representa a opção 1, porém é correto afirmar que fogueira[T2] ilumina[V] caverna[T1], o que representa a segunda opção. Caso a segunda opção não estivesse selecionada, a relação correta não seria sugerida.

4.2.3.4 Regra de advérbios

Se esta opção for selecionada, o sistema ignorará relações onde existem advérbios de negação, ou seja, se as palavras “não”, “nem”, “nunca”, “jamais” e “tampouco” foram encontradas antes do verbo, a relação não será sugerida.

Exemplo: “João não foi na padaria”, normalmente o sistema usaria a OTV1 para sugerir a relação João[T1] foi[V] Padaria[T2], mas como temos a palavra “não” antes do verbo, a relação será ignorada.

Para a análise semântica de frases, advérbios são importantes, porém, este tipo de análise não está dentro dos objetivos deste TC.

4.2.3.5 Regra de nomes compostos

Se esta opção for selecionada, o sistema tentará reconhecer, a partir dos resultados que o TreeTagger retornar, nomes compostos (termos com mais de uma palavra) que se encontram na ontologia.

Exemplo: Ciência da computação, normalmente esta palavra seria dividida em três lemmas. [Ciência] [da] [computação]. Com esta regra ativada, ele reconhecera a palavra como um todo [Ciência da computação].

4.2.3.6 Regra de rejeições

Esta opção visa a filtragem de relações indesejáveis a medida que o usuário usar o sistema. Temos três tipos de rejeições.

1 – Categoria1 -> Verbo ->Categoria2 [R1]

2 – T1 -> Verbo ->Categoria2 [R2]

3 – T1 -> Verbo ->T2 [Re]

Na opção R1, é feita uma rejeição entre duas categorias.

Exemplo: Torre[Lugar] Ficou[V] Faca[Objeto]. Se o sistema oferecer esta relação em algum processamento e o usuário julgar que [Lugar] [ficar] [Objeto] é uma relação que sempre deve ser evitada, ele tem a opção de marcar esta relação como uma rejeição para que ela não apareça mais durante o processamento de texto. Estão opção é a mais abrangente de todas porque envolve todos os termos encontrados nas duas categorias, logo o usuário deve usa-la com cuidado, caso contrario pode acabar criando uma rejeição entre termos que ocasionalmente poderia ser válida.

Na opção R2, é feita uma rejeição entre o termo analisado[T1], Verbo e a categoria do T2.

Exemplo: Pássaro[Ser vivo] forjou[verbo] espada[Objeto]. Se o sistema oferecer esta relação em algum processamento e o usuário julgar que [Pássaro[T1]] [Forja] [Objeto[CategoriaT2]] é uma relação que não é valida, ele tem a opção de marcar esta relação como uma rejeição para que ela não apareça mais durante o processamento de texto. Não tão

abrangente como a R1, mas ainda assim, se não for usada com cuidado pode ocasionar rejeições que poderiam ser válidas.

Na opção R3, é feita uma rejeição entre os termos [T1][T2] e o Verbo.

Exemplo: Pássaro[Ser vivo] cozinhou[verbo] João[Ser vivo]. Se o sistema oferecer esta relação em algum processamento e o usuário julgar que [Pássaro[T1]] [Cozinhou] [João[T2]] é uma relação que não é valida, ele tem a opção de marcar esta relação como uma rejeição para que ela não apareça mais durante o processamento de texto. Está é a rejeição mais simples, porque se refere diretamente aos termos e o verbo em questão.

4.2.3.7 Regra de "ser"

Esta regra leva em consideração a possibilidade de um relacionamento de "ser" entre termos que são adjacentes.

Exemplo: "João, diretor da escola Ernesto Alves, foi viajar.". Esta frase implica João é diretor. Como "João" e "diretor" estão adjacentes. O sistema irá sugerir a relação de [João] [ser][diretor].

4.2.3.8 Regra de lemma/verbos

Baseando-se no conceito do dicionário de Thesaurus [seção 2.3.5], esta regra contempla substituir *lemmas* do texto processado por verbos parametrizados.

Exemplo: "João sempre foi comandante do navio". Esta frase implica que "João comanda navio", porém o sistema não iria conseguir identificar esta relação por causa da falta do verbo "comandar". Então se o usuário parametrizar o *lemma* "comandante" e relaciona-lo com o verbo "comandar", todas as ocorrências de "comandante" no texto serão trocadas pelo verbo "comandar".

Importante salientar que, se neste exemplo, existisse o termo "comandante" na ontologia, e está regra estivesse ativa, o sistema nunca faria sugestões de relação com o termo "comandante", devido a sua substituição.

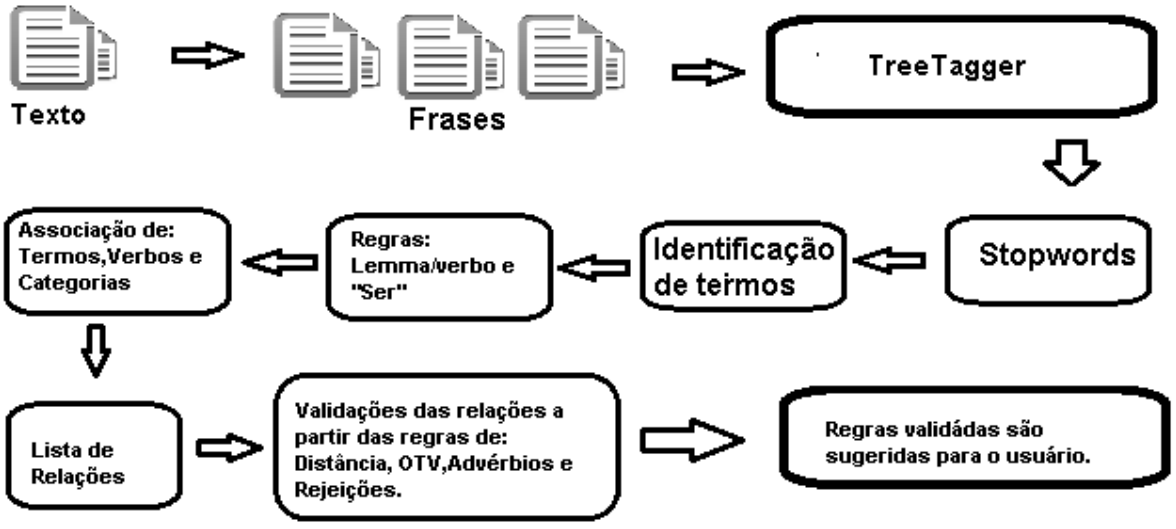
4.2.4 Funcionamento do algoritmo

O algoritmo recebe o ID do texto que se encontra no sistema para então busca-lo no banco de dados, em seguida o texto é dividido em frases para o processamento.

Depois de o texto ser divido, o TreeTagger é aplicado em cada frase e seus resultados armazenados na memória. Se a regra de *stopwords* estiver ativa, neste momento é realizada a remoção das palavras parametrizadas. Então para cada frase, procuramos na ontologia quais termos que ali se encontram, levando em consideração a regra de nomes compostos se estiver ativa. Neste momento também aplicamos a regra de lemmas/verbos e a regra de "ser". Então, para cada termo encontrado realizamos a busca de verbos, dentro da mesma frase, que estejam associados a sua categoria para depois procurar o segundo termo, também dentro da frase, e sugerir uma relação.

Quando uma possível relação é encontrada são aplicadas as regras parametrizadas de Distância, OTV, Advérbios e Rejeições. Caso a relação passe por todas as validações, a sugestão é retornada para o usuário.

Figura 25 – Algoritmo



Fonte: Autor

4.2.5 Banco de dados

Como já mencionado anteriormente, o banco usado é uma expansão do trabalho de Fetzer (2013). Foram realizadas as seguintes adições/modificações:

Categorias: Nesta tabela serão mantidas as categorias que o sistema usará para relacionar com as classes. Nela temos os campos nome e observação. Novas categorias poderão ser criadas se o usuário desejar.

Categorias_verbo_rejeito: Nesta tabela serão armazenadas as relações de verbo rejeitadas entre duas categorias, que são relativas às regras de rejeição [seção 4.2.3.6**].**

Erros_enconding_tagger: Nesta tabela encontram-se uma relação de palavras que resultaram em erros durante o processamento dos textos devidos a problema de *enconding*. A tabela tem como funcionalidade guardar associações manuais entre palavras e *lemmas*.

Parametros: Nesta tabela são armazenados os parâmetros do sistema, principalmente em relação ao algoritmo de reconhecimento de relações.

Palavra_verbo: Nesta tabela são armazenadas as parametrizações relativa a regra de lemma/verbo [seção 4.2.3.8]

Relacao: Esta tabela sofreu uma única modificação que foi a adição do campo *id_verbo*. Este campo representa a relação entre os termos. Por exemplo: João[ID_TERMO_PAI] cortou[id_verbo] a árvore[ID_TERMO_FILHO].

Termo_categoria: Nesta tabela temos a classificação de categorias para os termos da ontologia. Por exemplo: João está associado a categoria Ser vivo.

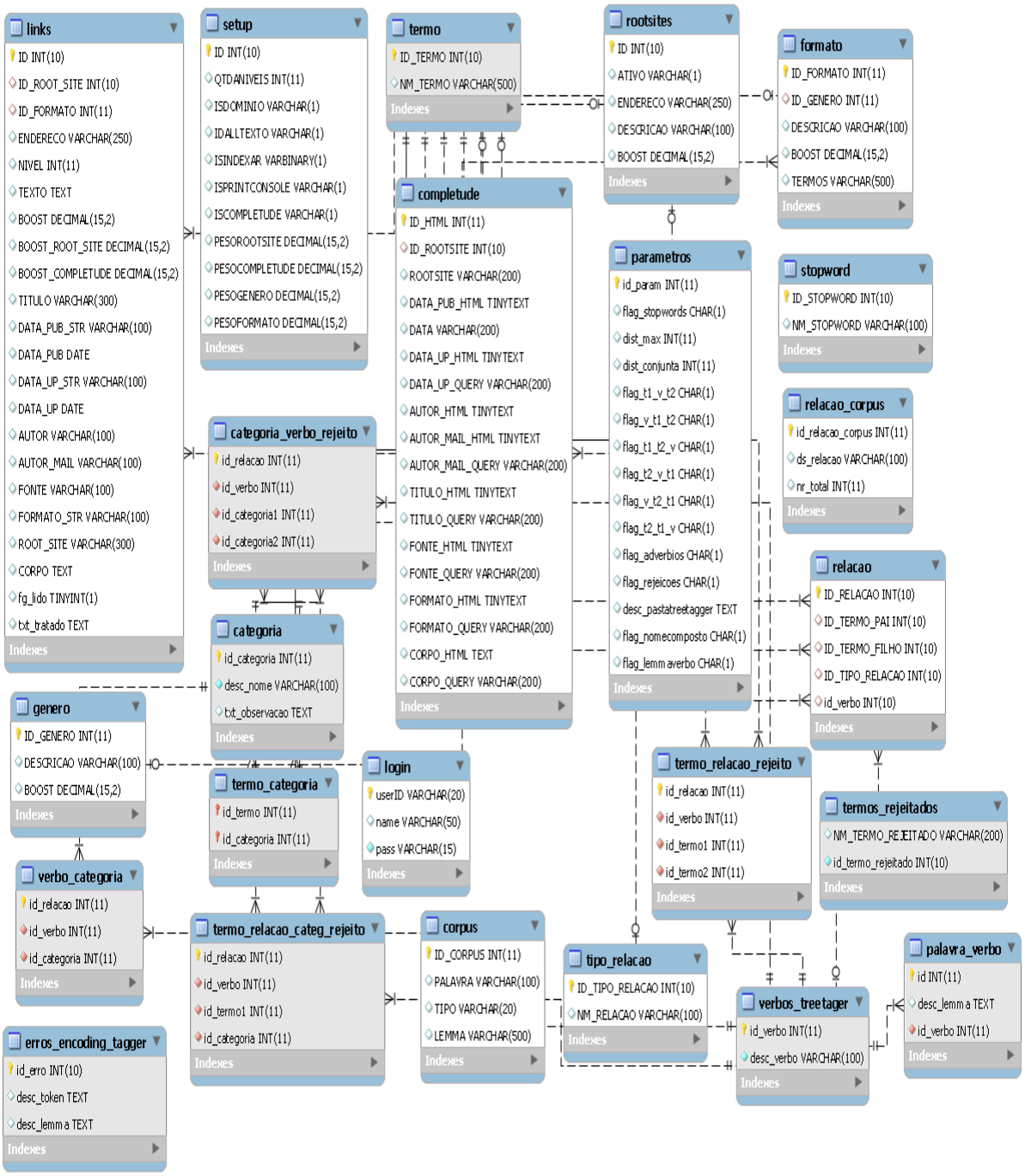
Termo_relacao_categ_rejeito: Nesta tabela são armazenadas as relações de verbo rejeitadas entre um termo e uma categoria, que são relativas às regras de rejeição [seção 4.3.2.6**].**

Termo_relacao_rejeito: Nesta tabela são armazenadas as relações de verbo rejeitadas entre dois termos, que são relativas às regras de rejeição [seção 4.2.3.6**].**

Verbo_categoria: Nesta tabela encontram-se a relação entre verbos e categorias, por exemplo: verbo caminhar está associado à categoria ser vivo.

Verbo_treetager: Nesta tabela encontram-se todos os verbos importados do corpus do TreeTagger.

Figura 26 – Modelo ER do sistema



Fonte: Autor.

4.3 Validação e resultados da solução desenvolvida

Nesta seção será descrito os testes e situações que foram encontradas durante a implementação e durante os testes finais.

4.3.1 Implementação

Inicialmente, para ajudar com testes durante a implementação da solução desenvolvida, foi importado um livro para dentro do banco de dados, o nome do livro é “As crônicas saxônicas. Livro 1. O último reino”, escrito pelo autor Bernard Cornwell. Depois de importado, o livro foi processado pelo TreeTagger, para que então todos os verbos que ali se encontravam, aproximadamente de 1200 verbos, pudessem ser extraídos e usados para realizar a parametrização das categorias de “Ser vivo, Organização, Local, Coisa/Objeto, Outros)”. Também foram extraídas praticamente todas as *tokens* que continham a etiqueta de classificação “NOM” para serem adicionadas como termos para a ontologia.

Para os testes de implementação foram usadas diversas parametrizações, inclusive algumas regras, que não tinham sido consideradas anteriormente para o projeto, foram criadas a partir de situações que foram econtradas nesta etapa. São elas: Regra de advérbios, Regra de ser (esta regra foi criada durante a validação final [4.3.2], Regra de nomes compostos, Regra de lemma/verbos, Regras de OTV.

4.3.2. Primeira validação

Depois de a implementação estar considerada como finalizada, o orientador Eduardo sugeriu que extraíssemos notícias do website http://www.baguete.com.br/noticias, então 30 notícias foram lidas, analisadas e importadas manualmente. A partir destes dados, foram inseridos termos na ontologia na medida que eu os julgava interessante para o futuro reconhecimento de relações. Foi decidido que para complementar a base de dados, manter os termos previamente importados do livro. Ocasionalmente, foram parametrizados alguns verbos extras que foram encontrados durante a leitura dos textos que ainda não se encontravam dentro das categorias.

Para a validação destas notícias, foram usados os seguintes parâmetros:

Stopwords: desativada.

DMI: 5

DMC: 9

Nesta validação foi optado por manter os textos com as *stopwords* e usar uma DMI/DMC um pouco mais elevada, por que as vezes a regra das *stopwords* podem acabar removendo palavras de mais entre alguns termos e verbos, assim afetando a consistência da DMI/DMC.

Todas opções da regra de OTV ativadas.

Neste caso, foram ativadas todas opções já que o objetivo era extrair o máximo de relações possíveis dos textos. A contrapartida de ter todas as opções ativas é que muitas relações que são sugeridas são inúteis e o tempo da análise que o usuário deve fazer também acaba aumentado, devido ao fato de ter muitas relações sugeridas.

Regra de advérbios: Ativa.

Regra de rejeições: Ativa.

Apesar de a regra de rejeições estar ativa e muitas das sugestões que foram retornadas serem inválidas, foi optado por não rejeitar permanentemente nenhuma sugestão. Como comentado já na seção 4.2.3.6, o usuário deve tomar cuidado a rejeitar relações porque ocasionalmente pode acabar rejeitando algo que poderia ser válido em futuras análises, e outro motivo pelo qual foi decidido não realizar a rejeição, foi que aumentaria significantemente o tempo de análise manual de cada notícia. O tempo médio de análise por notícia estava aproximadamente de 20 a 25 minutos, considerando a leitura, extração de termos, processamento e análise manual das sugestões retornadas.

Regra de nomes compostos: Ativa.

Regra de "ser": Ativa.

Regra de lemma/verbos: Ativa.

Vale salientar que a regra de "ser" foi implementada durante as validações devido a alguns padrões encontrados. Na regra de lemma/Verbos, também foram realizadas algumas parametrizações depois do processamento de alguns textos iniciais devido ao mesmo padrões encontradas para a regra de "Ser". Para entender estas modificações, primeiramente devemos observar os seguintes padrões que foram encontrados em muitas ocasiões durante a leitura dos textos. Consideramos os exemplos:

1 -"Jõao, presidente da empresa TudoBom",

2 - "Maria, CEO da PreçosBaratos, empresa de revenda de materiais industriais.".

Quanto ao exemplo 1, nota-se que existe uma relação entre João[T1], empresa[T2] e TudoBom[T3]. Porém não existe nenhum verbo para podermos sugerir estas relações. Em

casos assim, foi parametrizado na seção lemma/verbos algumas palavras como “presidente”, “chefe”, “CEO”, “administrador”. No lugar destas palavras foram relacionados verbos como: “comanda”, “administra”. Então, quando o texto for processado a relação João[T1] administra[V] empresa[T2] e João[T1] administra[V] TudoBom[T3] será sugerida.

Quanto a regra de “ser”, nota-se que muitos termos aparecem adjacente uns aos outros, implicando o verbo ser entre eles, por exemplo: “..da PreçoBaratos[T1] empresa[T2] de revenda...”. Claramente, o PreçosBaratos é uma empresa, porém não existe nenhum verbo na frase que possa identificar esta relação. Então para contemplar estas situações foi implementada a regra de “ser”. Em contrapartida, quando esta regra está ativa, muitas situações como “empresa X firmou pareceria com as empresas Y,W e Z.” Acabam retornando sugestões invalidas, por exemplo: Y[T1] ser[V] W[T2], W ser Y, W ser Z e etc.

4.3.3 Resultados

Levando em consideração as parametrizações da seção 4.3.2, foi realizado um levantamento das relações e termos mais encontrados durante a validação.

Relações encontradas que foram aceitas: 215.

Relações que mais tiveram aparições: apresentadas na figura 25.

Figura 27 – Tabela de relações mais frequentes

Quantidade	Verbo
55	ser
16	ter
16	é
10	administrar
9	comandar
7	atender
5	oferecer
4	adotar
4	fornecer
4	liderar
3	capturar
3	coordenar
3	crescer
3	envolver
3	firmar
3	integrar
3	manter

Fonte: Autor

Apesar de os verbos ‘ser’ e ‘é’ estarem separados na tabela devido ao processamento do TreeTagger, eles podem ser considerados iguais, ou seja, com o mesmo sentido dentro do contexto da ontologia.

Termos que mais tiveram aparições nas relações aceitas: apresentados na figura 26.

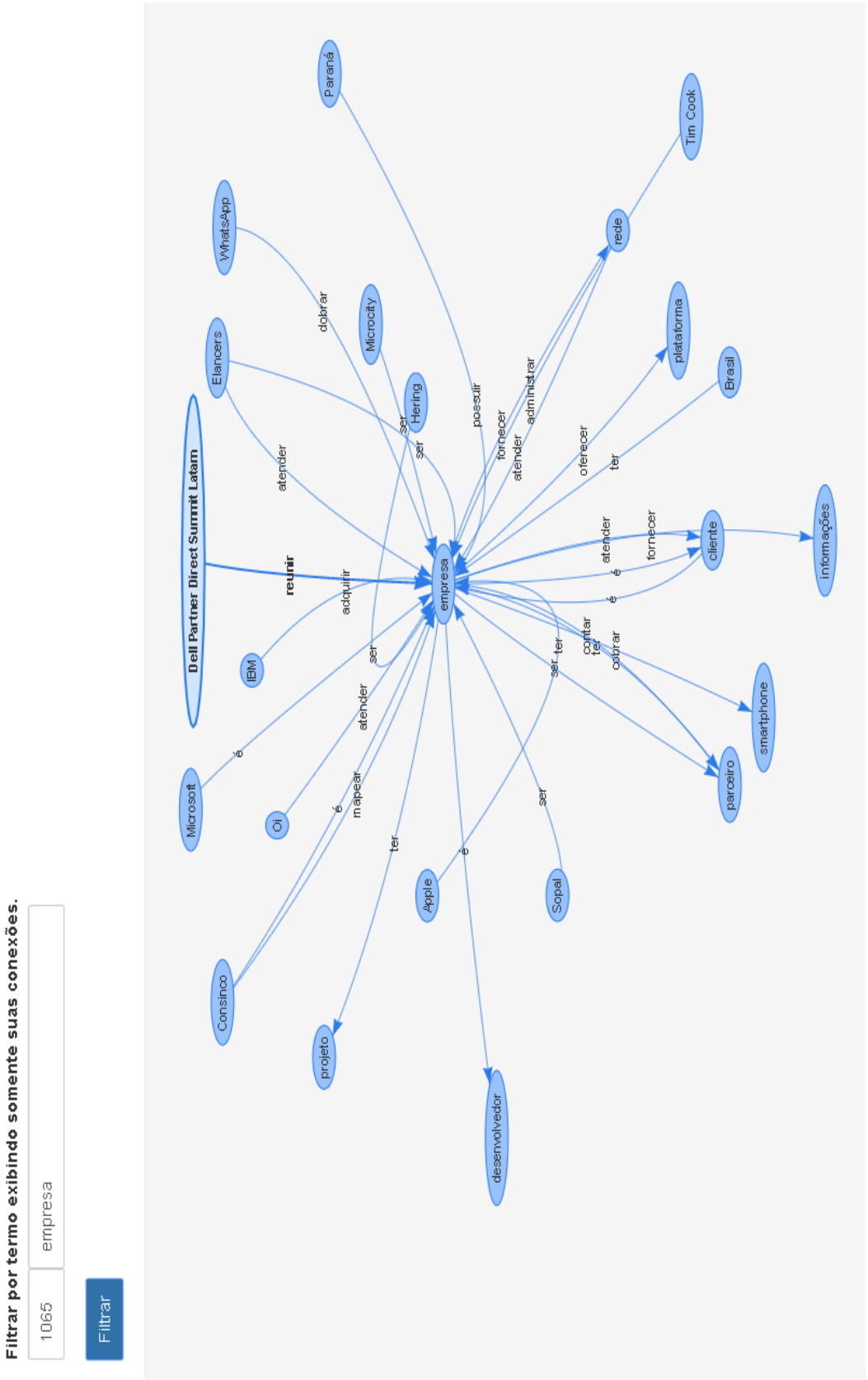
Figura 28 – Tabela de termos mais frequentes

Quantidade	Termos
29	empresa
9	IBM
8	mercado
8	parceiro
8	Brasil
8	Consinco
7	plataforma
7	rede
7	cliente
6	operadora
6	Dell
5	TI
5	tecnologia
5	projeto
5	Dataplace
5	Trombini

Fonte: Autor

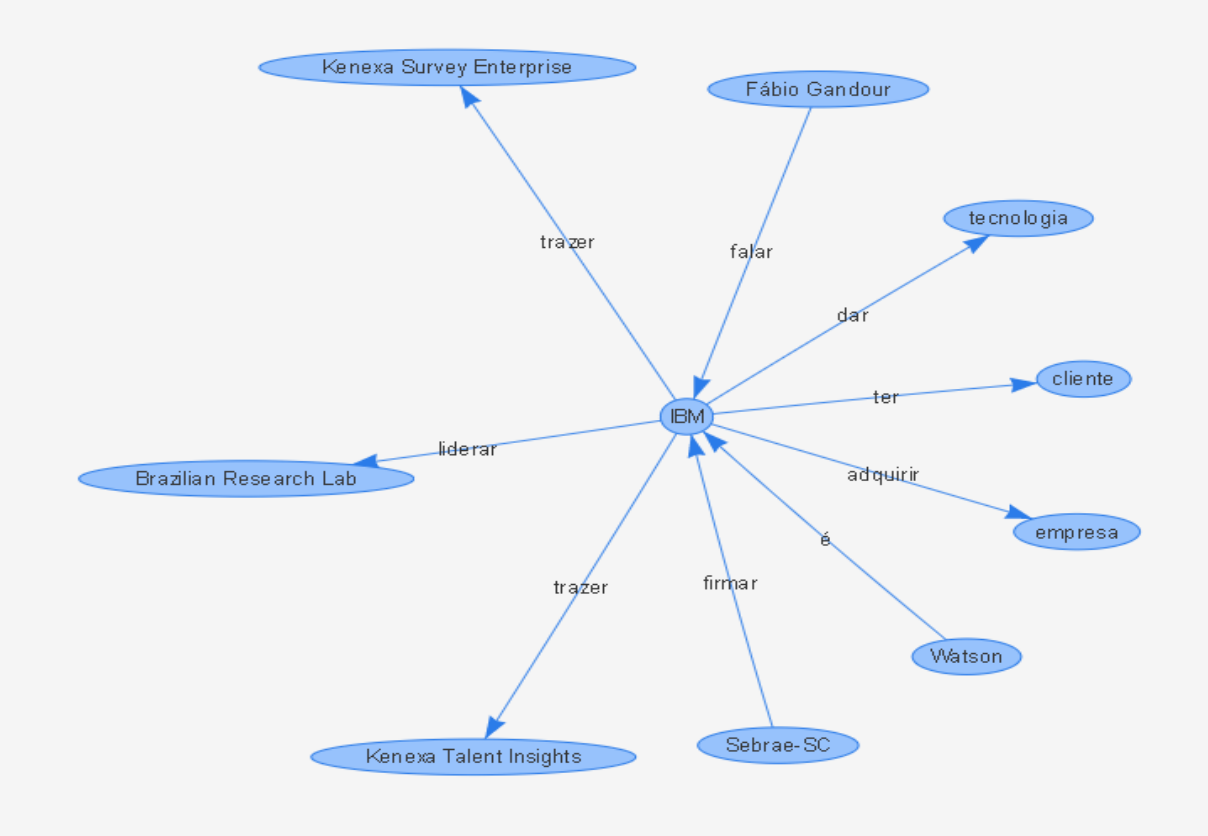
A seguir, são apresentadas algumas imagens da ontologia com alguns dos termos mais frequentes e suas relações aceitas.

Figura 29 – Relações encontradas para o termo “empresa”.



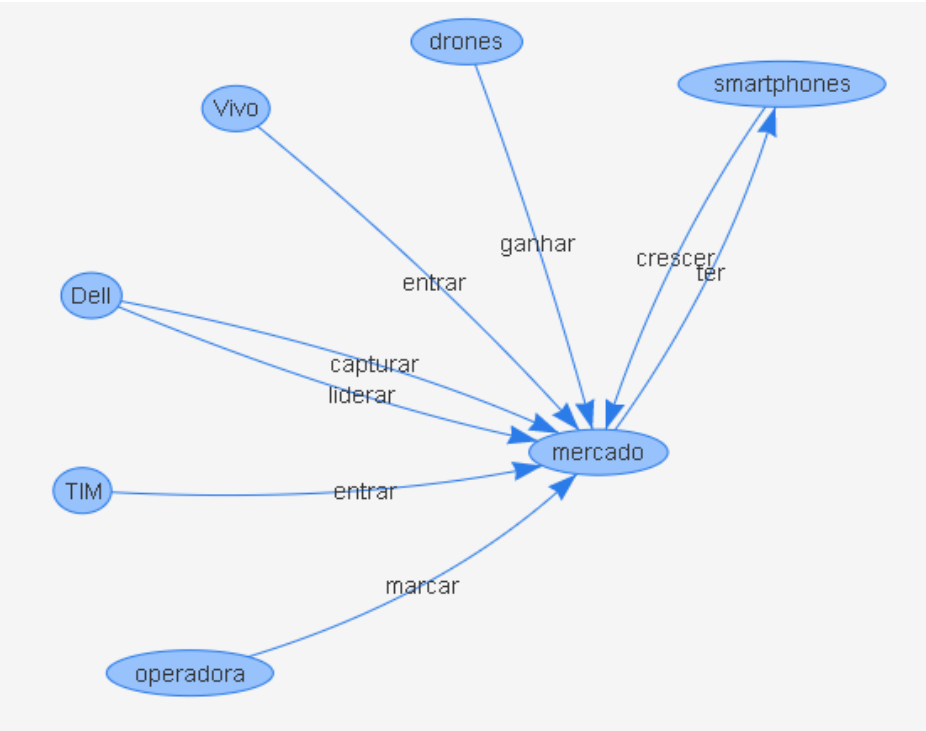
Fonte: Autor.

Figura 30 – Relações encontradas para o termo “IBM”.



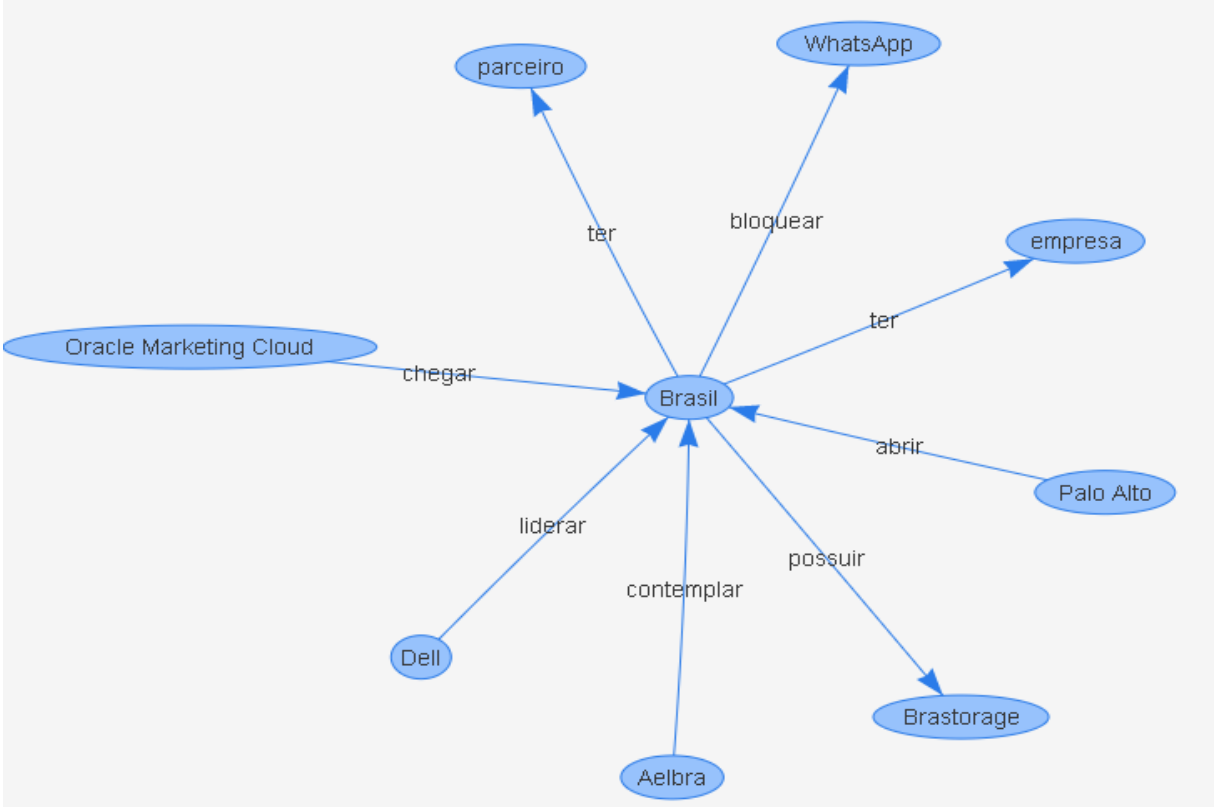
Fonte: Autor

Figura 31 – Relações encontradas para o termo “mercado”.



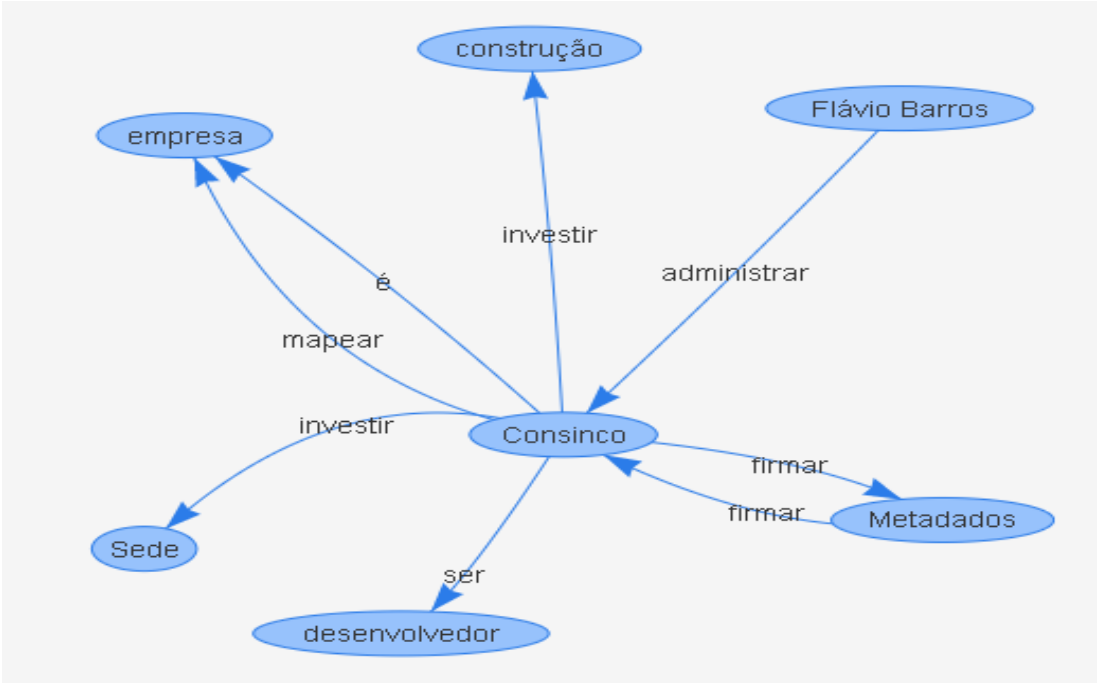
Fonte: Autor

Figura 32 – Relações encontradas para o termo “Brasil”.



Fonte: Autor

Figura 33 – Relações encontradas para o termo “Consinco”.



Fonte: Autor

4.3.4 Segunda validação

Depois de a primeira validação estar finalizada e adiantada em à data de entrega do trabalho, foi optado por enriquecer a ontologia através da inserção de mais notícias na base de dados. Então outras 20 notícias foram processadas usando os mesmos parâmetros da primeira validação. Em relação à escolha de quais notícias foram escolhidas para serem processadas, em quanto na primeira validação não teve um padrão, nesta segunda validação, na maioria dos casos foi dada preferência por notícias relacionadas a termos que representavam empresas que já se encontravam na ontologia.

4.3.5 Resultados

Nesta seção são apresentados os resultados de validação **todas** as 50 notícias, ou seja, da primeira e segunda validação em conjunto.

Relações encontradas que foram aceitas: 374.

Relações que mais tiveram aparições: apresentadas na figura 34.

Figura 34 – Tabela de relações mais frequentes

Quantidade	Verbo
96	ser
25	é
23	ter
15	comandar
13	administrar
9	atender
7	envolver
7	estar
6	liderar
5	adotar
5	criar
5	oferecer
5	firmar
4	fazer
4	atuar
4	coordenar
4	representar
4	adquirir
4	fornecer

Fonte: Autor.

Termos que mais tiveram aparições nas relações aceitas: apresentados na figura 37.

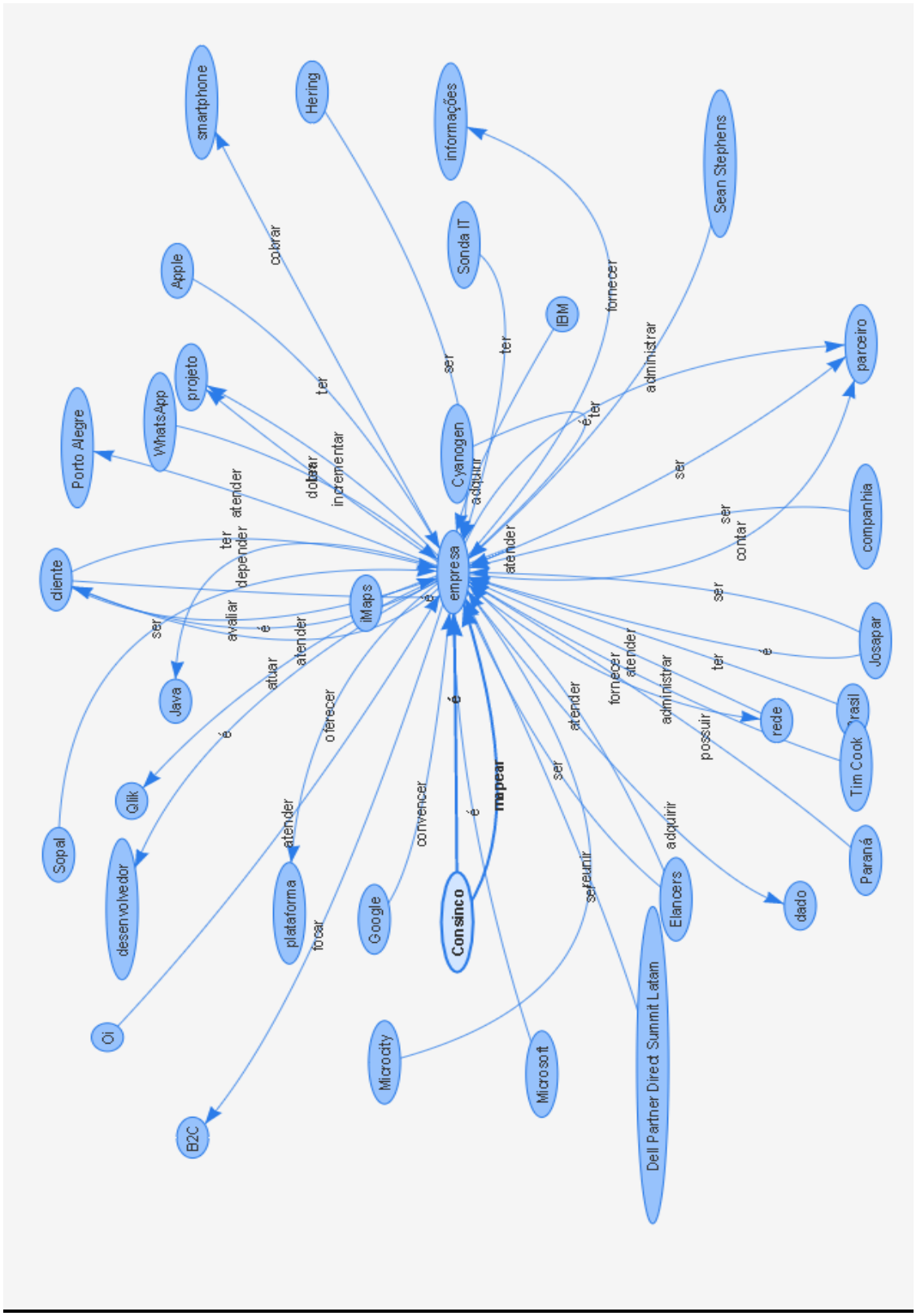
Figura 35 – Tabela de termos mais frequentes

Quantidade	Termo
45	empresa
17	Apple
14	projeto
13	plataforma
13	IBM
13	Google
11	Microsoft
10	cliente
10	Brasil
9	mercado
9	Android
8	software
8	rede
8	parceiro
8	Consinco

Fonte: Autor.

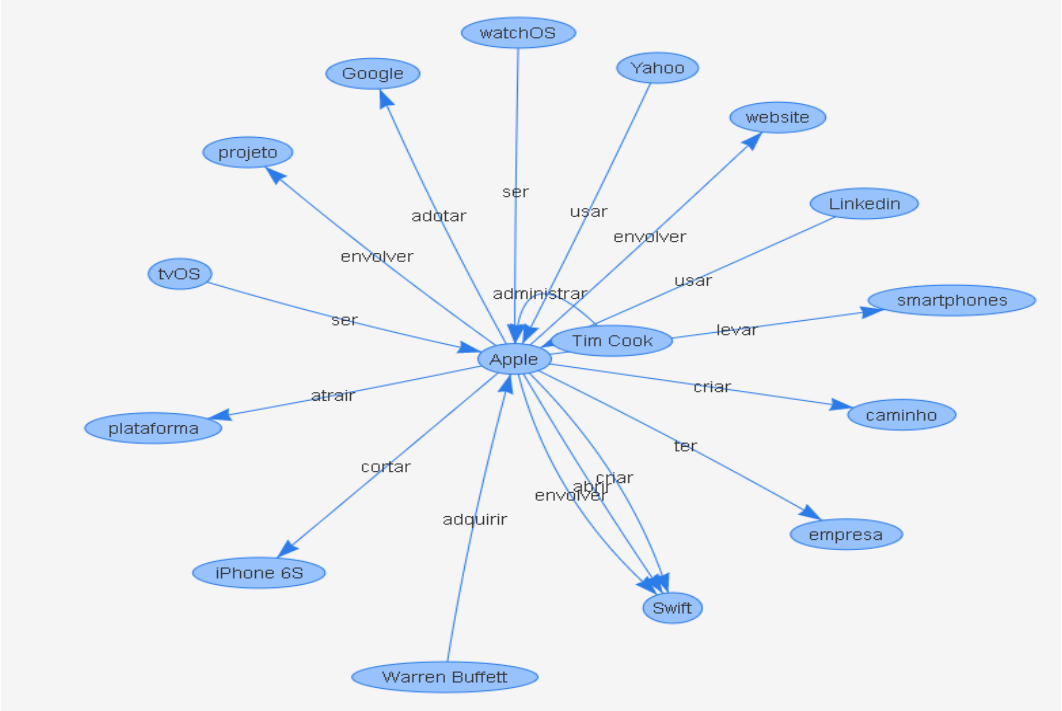
A seguir, são apresentadas algumas imagens da ontologia com alguns dos termos mais frequentes e suas relações aceitas.

Figura 36– Relações encontradas para o termo “empresa”.



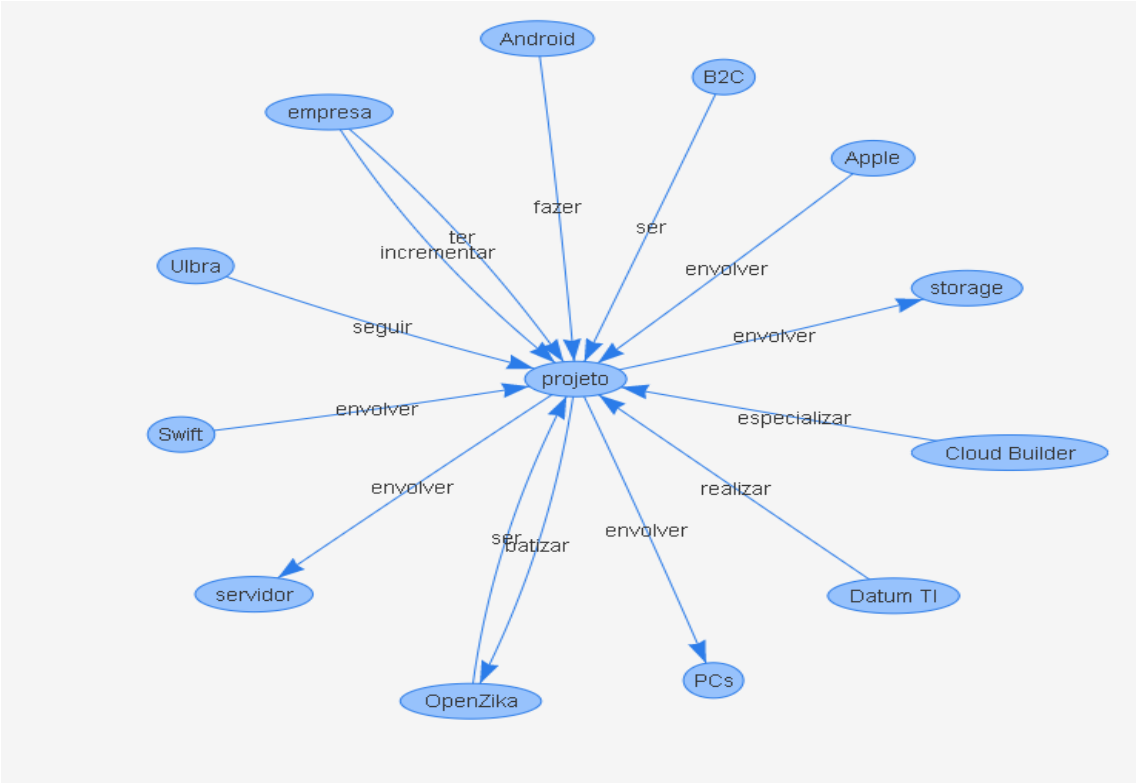
Fonte: Autor.

Figura 37– Relações encontradas para o termo “Apple”



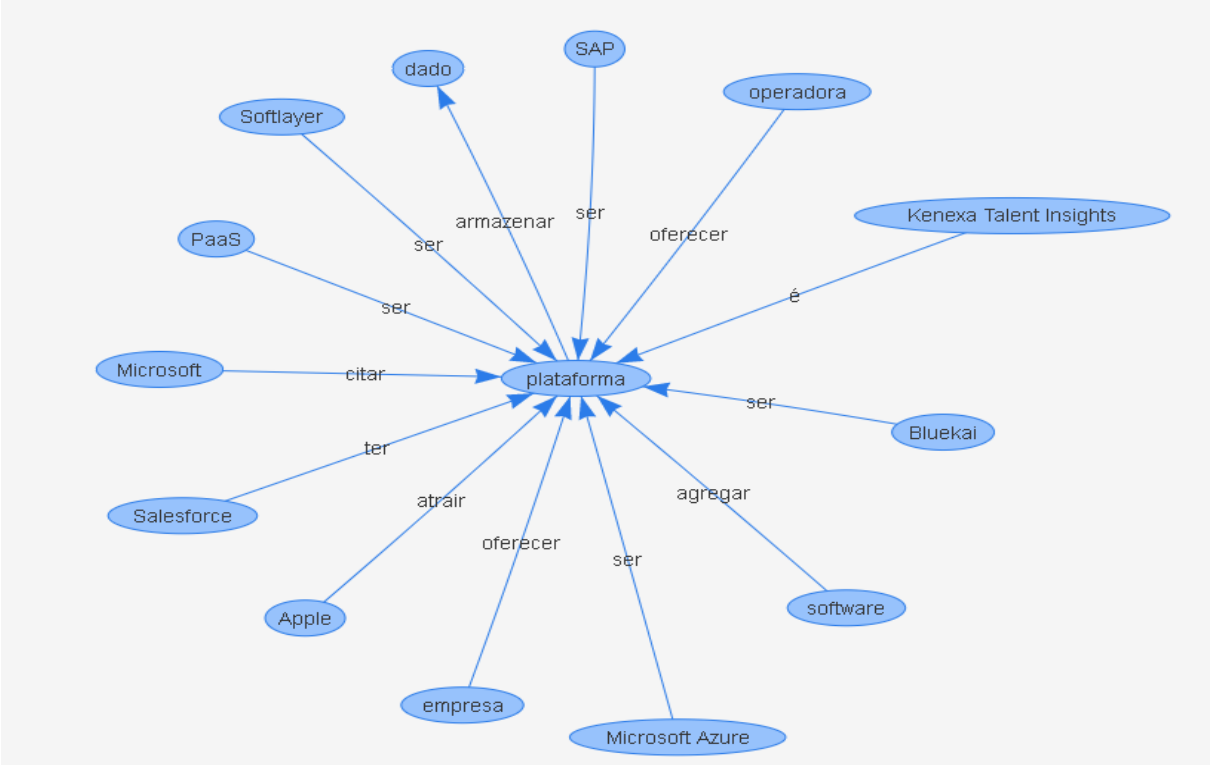
Fonte: Autor.

Figura 38– Relações encontradas para o termo “projeto”



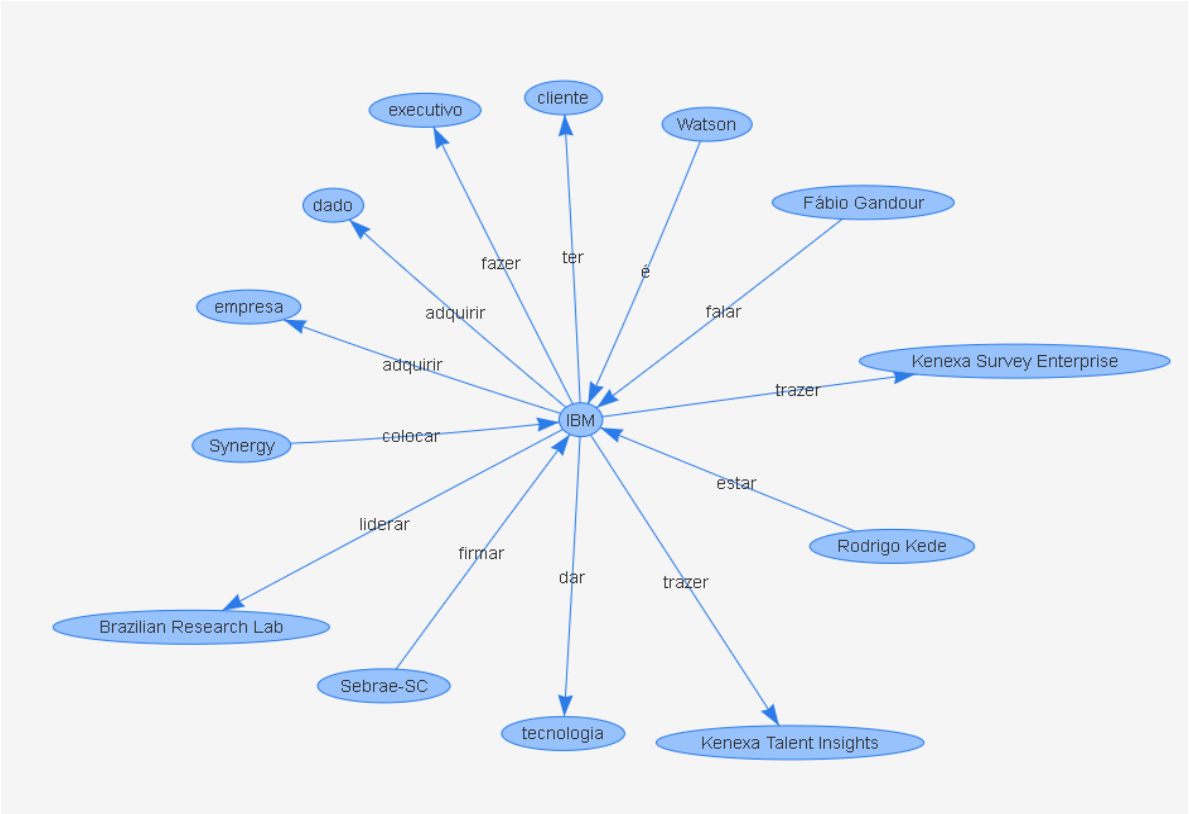
Fonte: Autor.

Figura 39– Relações encontradas para o termo “plataforma”



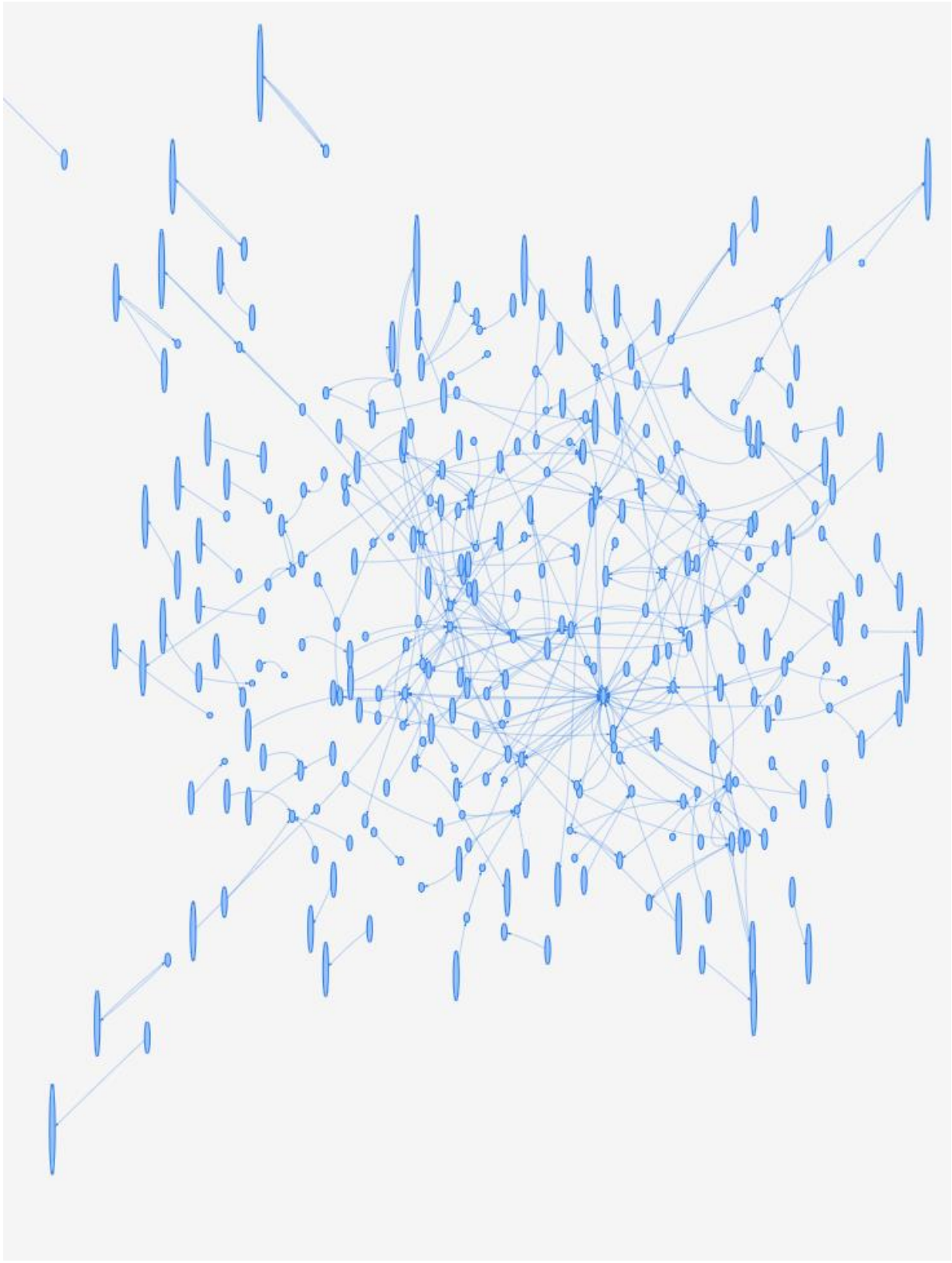
Fonte: Autor.

Figura 40– Relações encontradas para o termo “IBM”



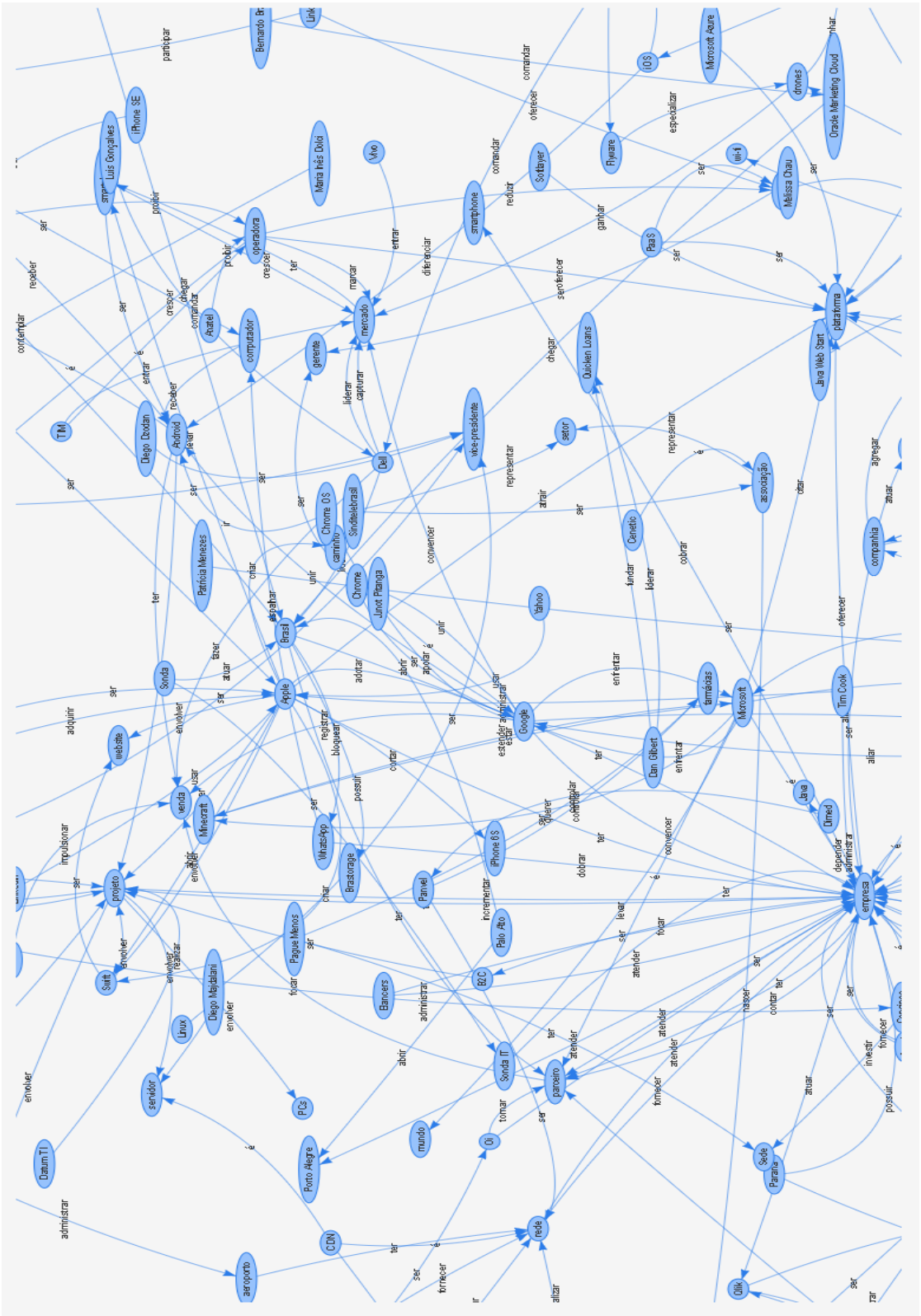
Fonte: Autor.

Figura 41 – Visão geral da ontologia com as relações encontradas.



Fonte: Autor

Figura 42 – Visão geral da ontologia com um pouco de zoom



Fonte: Autor.

5 CONCLUSÕES E TRABALHOS FUTUROS

O trabalho aqui desenvolvido consiste em um sistema que auxilia o usuário na identificação de relações entre termos de uma ontologia. O sistema aplica uma série de regras, em cima de textos que se encontram no banco de dados, inseridos pelo usuário ou previamente importados, para uma lista de sugestões de novas relações entre os termos. No final cabe ao usuário aceitar ou não essas sugestões e assim expandir a quantidade de relações da ontologia que se encontra no sistema.

A área de processamento de linguagem natural é bem complexa, a linguagem humana não é “binária” e está sempre sujeita a interpretações diferentes devido ao contexto em que ela está sendo usada, logo para obter bons resultados geralmente é viável a intermediação de um usuário nos sistemas que buscam a interpretação de textos, sendo no treinamento de aprendizado de linguagem do sistema ou nas etapas finais de validação dos resultados.

Quanto à sugestão de relações entre entidades, o sistema utiliza a ferramenta TreeTagger para o tratamento léxico dos textos em conjunto com as regras de sugestões de relações, que são um conjunto de regras parametrizáveis desenvolvidas a partir da análise de alguns trabalhos relacionados e de padrões encontrados durante a leitura de textos em geral. O TreeTagger foi escolhido por já ter sido utilizado e aprovado previamente por Fetzer(2013), apesar de ocasionalmente a análise não ser 100% correta. Nestes casos cabe a interpretação do usuário em conjunto a parametrização das regras.

A fim de validação, o sistema desenvolvido utilizou notícias, importadas manualmente, do website http://www.baguete.com.br/noticias. A partir da leitura destas noticias, também foram adicionados termos na ontologia que foram considerados relevantes para a validação.

Em relação a desenvolvimentos futuros que este trabalho pode receber, novas regras podem ser criadas para serem usadas em conjunto com as existentes, com o objetivo de considerar o contexto do texto e/ou parágrafo, ou seja, criar regras que analisam a questão da semântica dos textos e assim sugerir relações com uma interpretação mais complexa.

Devo admitir que durante os testes e validações, houve uma certa decepção com a pouca quantidade de relações que estavam sendo úteis em relação a todas foram sugeridas. Na maioria das vezes, aproximadamente de uma média geral, de uma a três relações eram aceitas de um total de aproximadamente 20 relações sugeridas. Mas depois de várias notícias terem

sido processadas e analisando a ontologia como um todo, notei que estas poucas relações que eram encontradas por notícia, em conjunto, construíram uma rede de informações bem interessante como pode ser visto nas imagens da seção 4.3.3.

REFERÊNCIAS

ALMEIDA, M.; BAX, M. Uma visão geral sobre ontologias: pesquisa sobre definições, tipos, aplicações, métodos de avaliação e de construção. Dez. 2003. Disponível em: <www.scielo.br/pdf/ci/v32n3/19019>. Acesso em 09 ago. 2015.

ALUÍSIO, S. M.; ALMEIDA G. M. B, O que é e como se constrói um corpus? Lições aprendidas na compilação de vários corpora para pesquisa lingüística. Revistas Unisinos, Universidade do Vale dos Sinos, 2006. Disponível em: < http://revistas.unisinos.br/index.php/calidoscopio/article/view/6002/3178> Acesso em: 10 ago. 2015.

ÁLVAREZ, A. C., *Extração de Informação de Artigos Científicos: uma abordagem baseada em indução de regras de etiquetagem.* 2007. 131 f. Dissertação (Obtenção do título de Mestre em Ciências de Computação e Matemática Computacional)-Universidade de São Paulo.

AMO, S., Curso de Data Mining. [201?]. Disponível em: <http://www.deamo.prof.ufu.br/arquivos/Aula17.pdf> Acesso em: 15 ago. 2015.

BARRETO A. F., Identificação de relações semânticas entre entidades mencionadas, Pontifícia Universidade Católica do Rio de Janeiro, [201]. Disponível em: <http://www.puc-rio.br/pibic/relatorio_resumo2010/relatorios/ctch/let/LET-Andrea%20da%20Fonseca%20Barreto.pdf> Acesso em: 09 ago. 2015.

BATISTA D. S. et al. Extração de Relações Semânticas de Textos em Português Explorando a DBpédia e a Wikipédia. In: linguamatica 5.1 (2013), pp. 41–57. Disponível em: <http://www.linguamatica.com/index.php/linguamatica/article/view/157/241> Acesso em: 20 ago. 2015

BRUCKSCHENET M. et al, Reconhecimento automático de relações entre entidades mencionadas em textos de língua portuguesa. Celsul, Pontifícia Universidade Católica do Rio Grande do Sul, 2008. Disponível em: <http://www.inf.pucrs.br/~linatural/Docs/publicacoes/SeRELeP-FINAL-Mirian-nropp.pdf> Acesso em: 09 ago. 2015.

FETZER, Felipe da Silva. Retroalimentação de estrutura de ontologia com base em lista de dados. 2013. 65 f. Monografia (Graduação) - Universidade de Santa Cruz do Sul, 2013

FILHO F. W.; LÓSCIO B. F. Web Semântica: Conceitos e Tecnologias, Universidade Federal do Piauí , [201?]. Disponível em: <http://www.ufpi.br/subsiteFiles/ercemapi/arquivos/files/minicurso/mc9.pdf> Acesso em: 13 ago. 2015.

FURTADO, Maria Inês Vasconcellos, *Inteligência competitiva para o ensino superior provado: Uma abordagem através da mineração de textos.* 2004. 129f. Dissertação

(Programa de Pós Graduação em Ciências em Engenharia Civil)-Universidade Federal do Rio de Janeiro.

GONZALES, M.; LIMA V. L. S, Recuperação de informação e processamento da linguagem natural. Edipucrs, Universidade Católica do Rio Grande do Sul, 2003. Disponível em: <http://www.inf.pucrs.br/~gonzalez/docs/minicurso-jaia2003.pdf> Acesso em: 10 ago. 2015.

HACK, A. F. et al, Text Mining. Portal de Periódicos UFSC, Universidade Federal de Santa Catarina, 2013. Disponível em: <http://www.inf.ufsc.br/~alvares/INE5644/G2_texto.pdf> Acesso em: 05 ago. 2015.

HAGÈGE C.; BAPTISTA J.; MAMEDE N., Desafios na avaliação conjunta do reconhecimento de entidades mencionadas: O Segundo HAREM, 2008, Capítulo 15, p. 261–274.

JUNIOR, Auto Tavares da Camara, *Processamento de linguagem natural para indexação automática semântico-ontológica.* 2013. 181 f. Dissertação (Programa de Pós Graduação em Ciência da Informação)-Universidade de Brasília.

JUNIOR, Jõao Ribeiro Carrilho, *Desenvolvimento de uma Metodologia para Mineração de Textos.* 2007. 96. Dissertação (Engenharia Elétrica do Departamento de Engenharia Elétrica do Centro Técnico Científico da PUC)- Pontifícia Universidade Católica do Rio de Janeiro.

MACHADO, A. P. et al, Mineração de texto em redes sociais aplicada à educação a distância. In: Revista Digital da CVA, Porto Alegre, v.6 n.26 2010.

MACHADO, Pablo Neves, *Extração de relações hiponímicas em corpora de língua portuguesa.* 2015. 80 f. Dissertação (Programa de Pós Graduação em Ciência da Computação)-Universidade de Católica do Rio Grande do Sul.

MORAIS, E.; AMBRÓSIO, A. P. Ontologias: conceitos, usos, tipos, metodologias, ferramentas e linguagens. Dez. 2007. Disponível em: <http://www.inf.ufg.br/sites/default/files/uploads/relatorios-tecnicos/RT-INF_001-07.pdf>. Acesso em 12 ago. 2015

MÜLLER, Daniel Nehme, *Compreensão da linguagem falada.* 2002. 72f. Dissertação (Programa de Pós Graduação em Computação)-Universidade Federal do Rio Grande do Sul.

NETO J. M. O.; TONIN S. D.; PRIETCH S. S. et al, Processamento de Linguagem Natural e suas Aplicações Computacionais. Portal do INPA, Universidade Federal de Mato Grosso, 2010. Disponível em: <https://www.inpa.gov.br/erin2010/Artigo/Artigo9.pdf> Acesso em: 07 ago. 2015.

SANTOS D.; MAMEDE N.; BAPTISTA J., Extraction of Family Relations between Entities. Proceedings of the INForum. Universidade do Algarve, 2008. Disponível em: <http://www.inesc-id.pt/pt/indicadores/Ficheiros/9070.pdf> Acesso em: 01 ago. 2010.

SILVA, Elcelina Rosa Correia Carvalho, *Técnicas de data e text mining para anotação de um arquivo digital*. 2010. 96 f. Dissertação (Obtenção do grau de Mestre em Engenharia Electrónica e Telecomunicações)-Universidade de Aveiro.

SCHUSTER FILHO, Roberto Antonio. Adaptação temporal e qualitativa sobre mecanismos de clipagem eletrônica. 2013. 77 f. Monografia (Graduação) – Universidade de Santa Cruz do Sul, 2013.