UNIVERSIDADE FEDERAL DO MARANHÃO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA PÓS-GRADUAÇÃO EM ENGENHARIA
DE ELETRICIDADE

UM PROCESSO INDEPENDENTE DE DOMÍNIO PARA O POVOAMENTO AUTOMÁTICO DE ONTOLOGIAS A PARTIR DE FONTES TEXTUAIS

Carla Gomes de Faria Alves

2013

Um Processo Independente de Domínio para o Povoamento Automático de Ontologias a partir de Fontes Textuais

Carla Gomes de Faria Alves

Mestre em Engenharia Elétrica na área de Ciência da Computação
Universidade Federal do Maranhão, 2004

Tese apresentada ao curso de Pós-Graduação em Engenharia de Eletricidade da Universidade Federal do Maranhão como parte dos requisitos para a obtenção do título de Doutor em Engenharia Elétrica na área de Ciência da Computação.

Orientadora: Profª. Drª. Rosario Girardi

2013

Alves, Carla Gomes de Faria.

Um processo independente de domínio para o povoamento automático de ontologias a partir de fontes textuais/Carla Gomes de Faria Alves. – São Luís, 2013.

192 f.

Impresso por computador (fotocópia).

Orientadora: Rosario Girardi.

Tese (Doutorado) – Universidade Federal do Maranhão, Programa de Pós-Graduação em Engenharia de Eletricidade, 2013.

1. Ontologias. 2. Povoamento de ontologias. 3. Processamento da linguagem natural. I. Título.

UM PROCESSO INDEPENDENTE DE DOMÍNIO PARA O POVOAMENTO AUTOMÁTICO DE ONTOLOGIAS A PARTIR DE FONTES TEXTUAIS

Carla Gomes de Faria Alves

Tese aprovada em 05 de junho de 2013

Profª Maria del Rosário Girardi Gutierrez, Drª.
(Orientadora)

Prof. Paulo Jorge Freitas de Oliveira Novais, Dr.
(Membro da Banca Examinadora)

Prof. Pedro Manuel Rangel Santos Henriques, Dr.
(Membro da Banca Examinadora)

Prof. Evandro de Barros Costa, Dr.
(Membro da Banca Examinadora)

Prof. Francisco José da Silva e Silva, Dr.
(Membro da Banca Examinadora)

À minha família!

AGRADECIMENTOS

Agradeço a todos que contribuíram direta ou indiretamente para elaboração desta tese, em especial:

À minha mãe, Maria Helena, meu marido, Frederico Alves, e as minhas filhas, Júlia e Aline, pelo o amor, incentivo, dedicação e apoio.

À minha querida irmã, Luciana, pelo apoio e incentivo.

Às minhas amigas, Ilnara, Camila, Mytsa e Lecinha, pelo apoio e incentivo.

À Professora Rosario, pela orientação segura e presente, pelos ensinamentos e dedicação imprescindíveis para a realização deste trabalho.

Aos meus companheiros de laboratório, em especial, Raimundo, Ivo, Luís, Rodrigo, Nailson e Giulia, pela amizade e apoio no decorrer dessa trajetória.

A toda Coordenação da Pós-Graduação, coordenador, funcionários e professores, pelos bons serviços oferecidos e que foram fundamentais para a conclusão do doutorado.

RESUMO

A demanda por sistemas baseado em conhecimento é crescente considerando suas aptidões para a solução de problemas complexos e para a tomada de decisão. As ontologias são formalismos para a representação de conhecimento de um dado domínio, que permitem o processamento semântico das informações e, através de interpretações mais precisas das informações, os sistemas apresentam maior efetividade e usabilidade. O povoamento de ontologias visa a instanciação de propriedades e relacionamentos não taxonômicos de classes de ontologias. Entretanto, o povoamento manual de ontologias por especialistas de domínio e engenheiros do conhecimemto é uma tarefa cara e que consome muito tempo. O povoamento de ontologias rápido e com baixo custo é crucial para o sucesso de aplicações baseadas em conhecimento. Portanto, torna-se fundamental uma semi-automatização ou automatização desse processo. Esta tese propõe um processo genérico para o problema do Povoamento Automático de Ontologias, especificando suas fases e técnicas que podem ser aplicadas em cada uma delas. É também proposto um Processo Independente de Domínio para o Povoamento Automático de Ontologias (DIAOP-Pro) a partir de fontes textuais, que aplica técnicas de processamento da linguagem natural e extração de informação para adquirir e classificar instâncias de ontologias. O DIAOP-Pro se constitui em uma abordagem original uma vez que propõe o povoamento automático de ontologias utilizando uma ontologia para a geração automática de regras para extrair instâncias a partir de textos e classifica-as como instâncias de classes da ontologia. Estas regras podem ser geradas a partir de ontologias específicas de qualquer domínio, tornando o processo independente de domínio. Para avaliar o processo DIAOP-Pro foram conduzidos quatro estudos de caso de modo a demonstrar a sua efetividade e viabilidade. O primeiro estudo de caso foi realizado para avaliar a efetividade da fase “Identificação de Instâncias Candidatas”, no qual foram comparados os resultados obtidos com a aplicação de técnicas estatísticas e de técnicas puramente lingüísticas. O segundo estudo de caso foi realizado para avaliar a viabiliadade da fase “Construção de um Classificador”, através da experimentação com a geração automática do classificador. O terceiro e o quarto estudo de caso foram realizados para avaliar a efetividade do processo proposto em dois domínios distintos, o jurídico e o turístico. Os resultados indicam que o processo DIAOP-Pro povoa ontologias específicas de qualquer domínio com boa efetividade e com a vantagem adicional da independência do domínio.

Palavras-chave: Ontologias, Povoamento de Ontologias, Processamento da Linguagem Natural, Extração de Informação e Engenharia do Conhecimento.

Fonte: Faria, Carla. **Um Processo Independente de Domínio para o Povoamento Automático de Ontologias a partir de Fontes Textuais.** 2013. 192 p. Tese (Doutorado em Engenharia de Eletricidade), Universidade Federal do Maranhão, São Luís.

ABSTRACT

Knowledge systems are a suitable computational approach to solve complex problems and to provide decision support. Ontologies are an approach for knowledge representation about an application domain, allowing the semantic processing of information and, through more precise interpretation of information, turning systems more effective and usable. Ontology Population looks for instantiating the constituent elements of an ontology, like properties and non-taxonomic relationships. Manual population by domain experts and knowledge engineers is an expensive and time consuming task. Fast ontology population is critical for the success of knowledge-based applications. Thus, automatic or semi-automatic approaches are needed. This work proposes a generic process for Automatic Ontology Population by specifying its phases and the techniques used to perform the activities on each phase. It also proposes a domain-independent process for automatic population of ontologies (DIAOP-Pro) from text that applies natural language processing and information extraction techniques to acquire and classify ontology instances. This is a new approach for automatic ontology population that uses an ontology to automatically generate rules to extract instances from text and classify them in ontology classes. These rules can be generated from ontologies of any domain, making the proposed process domain independent. To evaluate DIAOP-Pro four case studies were conducted to demonstrate its effectiveness and feasibility. In the first one we evaluated the effectiveness of phase "Identification of Candidate instances" comparing the results obtained by applying statistical techniques with those of purely linguistic techniques. In the second experiment we evaluated the feasibility of the phase "Construction of a Classifier", through the automatic generation of a classifier. The last two experiments evaluated the effectiveness of DIAOP-Pro into two distinct domains: the legal and the tourism domains. The results indicate that our approach can extract and classify instances with high effectiveness with the additional advantage of domain independence.

Keywords: Ontologies, Ontology Population, Natural Language Processing, Information Extraction and Knowledge Engineering.

Source: Faria, Carla. **A Domain Independent Process for Automatic Ontology Population from Text.** 2013. 192 p. Thesis (Pos-Graduation in Electrical Engineering), Federal University of Maranhão, São Luís**.**

ABREVIATURAS E SÍMBOLOS

AM	Aprendizagem de Máquina
ANNIE	A Nearly-New Information Extraction System
CREOLE	Collection of Reusable Objects for Language Engineering
DIAOP-Pro	Domain Independent Process for Automatic Ontology Population
DIAOP-Tool	Tool for Automatic Ontology Population
EI	Extração de Informação
GATE	General Architecture for Text Engineering
ICEIS	International Conference on Enterprise Information Systems
JAPE	Java Annotation Patterns Engine
LR	Language Resources
NER	Named Entity Recognition
OWL	Web Ontology Language
PLN	Processamento da Linguagem Natural
PO	Povoamento de Ontologias
POS	Part of Speech
PR	Processing Resources
DIAOP-Pro	Processo Independente de Domínio para o Povoamento Automático de Ontologias
REN	Reconhecimento de Entidades Nomeadas
RI	Recuperação de Informação

SIR	Statistics Information Recuperation
TF-IDF	Term Frequency – Inverse Document Frequency
VR	Visual Resources

Sumário

Lista de Figuras

Lista de Tabelas

1. Introdução

1.1. Motivação

A demanda por sistemas baseados em conhecimento é crescente considerando suas aptidões para a solução de problemas complexos e para o suporte à tomada de decisão. Estes sistemas têm como principais componentes uma base de conhecimento e um mecanismo de raciocínio capaz de realizar inferências sobre esta base e obter conclusões a partir desse conhecimento. As ontologias são formalismos para a representação de conhecimento, usadas pelos modernos sistemas baseados em conhecimento para representar e compartilhar a informação de um determinado domínio de aplicação. Estes formalismos permitem expressar um conjunto de entidades e seus relacionamentos, restrições, axiomas e o vocabulário de um dado domínio. Através do processamento semântico das informações contidas em uma ontologia, os sistemas baseados em conhecimento podem realizar interpretações mais precisas das informações e, assim, demonstrar maior efetividade e usabilidade que os sistemas de informação tradicionais.

A aquisição de conhecimento é um processo de alto custo, sujeito a erros. Tradicionalmente, as ontologias são povoadas por especialistas de domínio e engenheiros de conhecimento, em um trabalho complexo, lento e caro. Esta dificuldade na captura do conhecimento requerido pelos sistemas baseados em conhecimento é conhecida como gargalo da aquisição de conhecimento. Por isso, torna-se fundamental uma semi-automatização ou automatização desse processo.

O povoamento de ontologias constitui uma abordagem para automatizar ou semi-automatizar a instanciação de propriedades e relacionamentos não taxonômicos de classes de ontologias com conhecimento descoberto em diferentes fontes de dados, como documentos textuais. O povoamento de ontologias ocorre tanto em ontologias que não possuem instâncias, quanto em ontologias já

povoadas. Quando o povoamento de ontologias é realizado em ontologias que já possuem instâncias o processo é chamado de enriquecimento de ontologias.

O povoamento de ontologias com rapidez e baixo custo é crucial para o sucesso de aplicações baseadas em conhecimento.

Nas seções seguintes descreve-se de forma mais detalhada a problemática e os objetivos que a pesquisa se propõe a investigar.

1.2. Caracterização do problema

O problema do Povoamento de Ontologias (PO) a partir de fontes textuais envolve a realização de três tarefas: identificação de instâncias candidatas, construção de um classificador e classificação de instâncias.

Na identificação de instâncias candidatas o principal desafio é trabalhar com textos em linguagem natural, que são desestruturados. Os sistemas de computação estão aptos a compreender instruções escritas em linguagens de programação, porém possuem dificuldade em compreender instruções escritas em linguagem natural. Isso se deve ao fato das linguagens de programação serem extremamente precisas, contendo regras fixas e estruturas lógicas bem definidas que permitem ao sistema de computação saber exatamente como deve proceder a cada comando. Já na linguagem natural, uma simples frase normalmente contém ambigüidades, nuances e interpretações que dependem do contexto, do conhecimento de mundo, de regras gramaticais, culturais e de conceitos abstratos. Então é necessário que os textos passem por um processo de estruturação, com a finalidade de alcançar uma maior precisão na identificação de instâncias candidatas. Outro desafio é a identificação de entidades nomeadas, para fins de reconhecer nomes que se referem a objetos únicos, as instâncias.

Na construção de um classificador o principal desafio é a geração do classificador de forma automática, que realize a classificação de instâncias de ontologias específicas de um domínio. A geração do classificador de forma

automática pode ser realizada através da aplicação de técnicas de aprendizagem de máquina ou de técnicas de extração de informação. Com a aplicação de técnicas de aprendizagem de máquina supervisionadas é necessário a geração de n-classificadores para cada domínio, para que o classificador gerado realize a classificação de ontologias específicas de uma domínio. O trabalho manual para a construção de diversos classificadores torna inviável a aplicação de técnicas de aprendizagem de máquina supervisionadas. Uma solução é a aplicação de técnicas de aprendizagem de máquina supervisionadas combinadas com a aplicação de técnicas de aprendizagem de máquina não supervisionadas ou a aplicação de técnicas de extração de informação.

Na classificação de instâncias o principal desafio é a associação das instâncias de propriedades e de relacionamentos não taxonômicos as suas respectivas classes da ontologia com boa efetividade. Outro desafio é a representação efetiva das instâncias em uma determinada linguagem de especificação de ontologias.

1.3. Hipótese de pesquisa e Objetivos

1.3.1. Hipótese de Pesquisa

A hipótese de pesquisa que essa tese se propõe a demonstrar é se é possível povoar automaticamente ontologias de domínios específicos a partir de fontes textuais com boa efetividade.

1.3.2. Objetivo Geral

Contribuir com soluções para a aquisição automática de conhecimento de forma a diminuir os custos e esforços no povoamento de bases de conhecimento.

1.3.3. Objetivos Específicos

a) Formalização de técnicas para o povoamento automático de ontologias a partir de fontes textuais.

b) Desenvolvimento de ferramentas computacionais de suporte às técnicas propostas.

c) Avaliação das técnicas propostas através do povoamento de ontologias e desenvolvimento de sistemas baseados no conhecimento para o acesso à informação e suporte as decisões na área jurídica.

1.4. Aspectos Metodológicos

Esta pesquisa, de natureza qualitativa e de caráter exploratório, bibliográfico e experimental, faz parte do projeto de pesquisa HERMES, em execução desde janeiro de 2010. O HERMES é executado em parceria entre pesquisadores das universidades UFMA, UMinho, UFPE e UFAL e é financiado pela CAPES e FCT.

Para o alcance das metas deste trabalho, primeiro será realizada uma análise do estado da arte da aplicação do processamento da linguagem natural, da aprendizagem de máquina e da extração de informação no povoamento automático de ontologias a partir de fontes textuais. Para tanto, serão desenvolvidas duas disciplinas de estudo orientado nesses tópicos. A partir do conhecimento adquirido serão especificadas técnicas para o povoamento automático de ontologias centradas no processamento da linguagem natural, aprendizagem de máquina e/ou extração de informação. A partir das técnicas propostas será especificado um processo para o povoamento automático de ontologias a partir de fontes textuais.

Para o suporte computacional do processo proposto para o povoamento automático de ontologias a partir de fontes textuais será desenvolvida uma

ferramenta que permita reduzir os esforços no povoamento automático de ontologias e viabilizar o funcionamento de sistemas baseado em conhecimento.

Através da utilização da ferramenta computacional desenvolvida, o processo proposto será avaliado através de experimentos no povoamento automático de ontologias nos domínios jurídicos e turísticos e da utilização dessas ontologias na implantação de sistemas baseado em conhecimento para o acesso à informação e suporte a tomada de decisão nos domínios jurídico e turístico.

1.5. Justificativa

A maioria das técnicas propostas para o Povoamento Automático de Ontologias [13] [14] [28] [29] [30] [31] [38] [42] [43] [51] [69] realizam esta tarefa em apenas um domínio específico, através da aplicação de técnicas de Processamento da Linguagem Natural (PLN), Extração de Informação (EI) e/ou Aprendizagem de Máquina (AM).

Uma das principais contribuições desta tese é a formalização de um processo independente de domínio para o povoamento automático de ontologias a partir de fontes textuais. O processo propõe uma nova abordagem para o povoamento automático de ontologias que usa uma ontologia para a geração automática de regras para extrair instâncias a partir de textos e classifica-as como instâncias de classes da ontologia. Estas regras podem ser geradas a partir de ontologias específicas em qualquer domínio, tornando o processo proposto completamente independente de domínio.

1.6. Visão geral da solução proposta

Esta tese propõe um Processo Independente de Domínio para o Povoamento Automático de Ontologias (DIAOP-Pro). O DIAOP-Pro combina as

principais vantagens de diferentes técnicas do estado da arte para o Povoamento de Ontologias (PO) e supera as limitações identificadas.

A idéia central do DIAOP-Pro proposto é a utilização de uma ontologia para a geração automática do classificador utilizado no povoamento de ontologias específicas de um domínio a partir de fontes textuais.

Para a Identificação de Instâncias Candidatas é realizado um processamento lingüístico no corpus, através da aplicação de técnicas de PLN, principalmente, “Análise Morfo-Lexical”, “Reconhecimento de Entidades Nomeadas” e “Identificação de Co-Referências”. A “Análise Morfo-Lexical” tem como objetivo identificar as categorias gramaticais de cada token na sentença. O “Reconhecimento de Entidades Nomeadas” identifica nomes que se referem a objetos únicos no mundo, tais como, nomes de pessoas, organizações dentre outros. A “Identificação de Co-Referências” tem como objetivo identificar co-referências nominais e pronominais. A co-referência pronominal consiste de pronomes que se referem a entidades descritas previamente, enquanto que a co-referência nominal consiste de nomes que se referem a uma mesma entidade. Esta fase tem como objetivo identificar as instâncias de propriedades e de relacionamentos não taxonômicos de classes de ontologias.

Para a Construção de um Classificador são aplicadas técnicas de EI para a geração automática de regras lingüísticas a partir de uma ontologia e consultas a uma base de dados léxica. Estas regras lingüísticas podem ser geradas a partir de ontologias específicas de qualquer domínio, gerando classificadores para qualquer domínio. Esta fase tem como objetivo a geração automática de um classificador, que classifica ontologias específicas de um domínio.

Para a Classificação de Instâncias é utilizado o classificador gerado de forma automática na fase “Construção de um Classificador”, onde o classificador associa as instâncias de propriedades e de relacionamentos não taxonômicos as suas respectivas classes da ontologia e representa as instâncias em uma linguagem de representação de ontologias, OWL. É realizada uma consulta na

ontologia antes de fazer a instanciação propriamente dita, onde é verificado se a instância já existe ou não, através de uma comparação de todas as propriedades e relacionamentos não taxonômicos das instâncias. Esta fase tem como objetivo a geração da ontologia povoada.

Para avaliar o DIAOP-Pro proposto foram conduzidos quatro estudos de caso de modo a demonstrar a sua efetividade e viabilidade. O primeiro estudo de caso foi realizado para avaliar a efetividade da fase “Identificação de Instâncias Candidatas”, no qual foram comparados os resultados obtidos com a aplicação de técnicas estatísticas e de técnicas puramente lingüísticas. O segundo estudo de caso foi realizado para avaliar a viabilidade da fase “Construção de um Classificador”, através da experimentação da geração automática do classificador. O terceiro e o quarto estudo de caso foram realizados para avaliar a efetividade do DIAOP-Pro proposto em dois domínios distintos, o jurídico e o turístico. Foi desenvolvida a DIAOP-Tool – uma ferramenta para o Povoamento Automático de Ontologias – que provê suporte automatizado a aplicação do DIAOP-Pro. A DIAOP-Tool foi desenvolvida na linguagem JAVA e utiliza o GATE para a aplicação de técnicas de Processamento da Linguagem Natural (PLN) e de técnicas de Extração de Informação (EI). As ontologias povoadas automaticamente podem ser utilizadas na execução de sistemas baseados em conhecimento.

1.7. Organização da Tese

Esta tese, incluindo esta introdução, está organizada em 6 capítulos (Figura 01).

O segundo capítulo apresenta a formalização da definição de ontologia utilizada nesta tese e descreve as áreas de conhecimento: Processamento da Linguagem, Aprendizagem de Máquina e Extração de Informação enfatizando suas fases de particular importância nesta pesquisa.

O terceiro capítulo define o problema do Povoamento Automático de Ontologias com um estudo comparativo do estado da arte.

O quarto capítulo apresenta um Processo Independente de Domínio para o Povoamento Automático de Ontologias e descreve uma ferramenta de software, criada com o propósito de suportar o processo proposto.

O quinto capítulo descreve as avaliações do processo proposto com a aplicação da ferramenta criada. O primeiro estudo de caso avalia a efetividade da fase “Identificação de Instâncias Candidatas”. O segundo estudo de caso avalia a viabilidade da fase “Construção de um Classificador”. O terceiro e o quarto estudo de caso avaliam a efetividade do processo proposto em dois domínios distintos, nas áreas jurídica e turística.

No sexto e último capítulo são apresentadas as conclusões da tese, destacando os resultados obtidos e sugestões de trabalhos futuros.

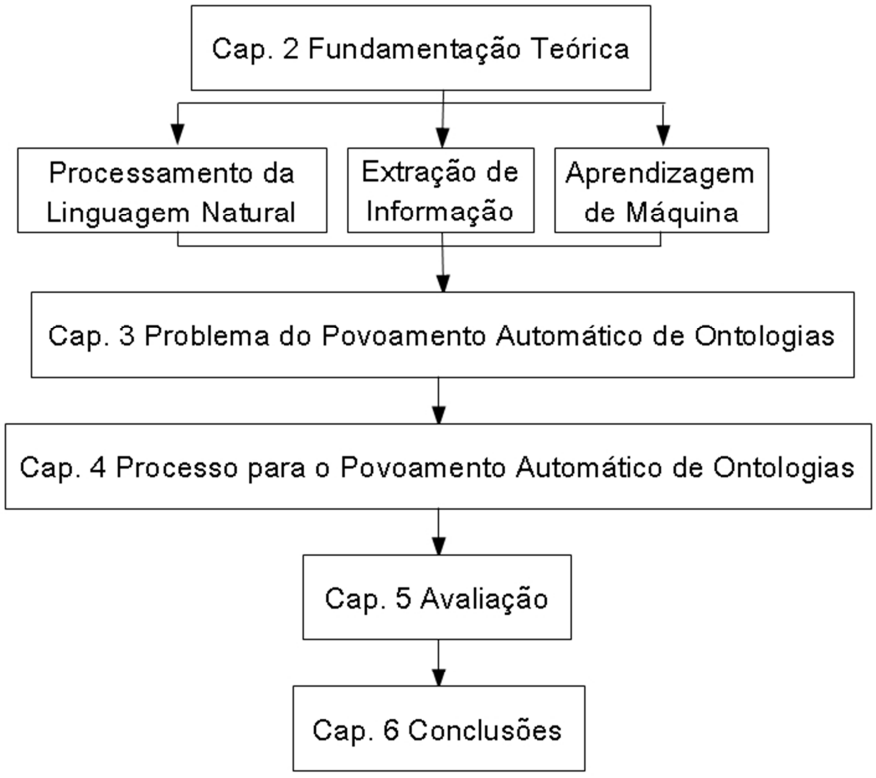


Figura 01: Organização da tese

2. Fundamentação Teórica

2.1. Introdução

Este capítulo apresenta a definição formal de ontologia utilizada nesta tese. Descreve uma visão geral das principais áreas de conhecimento envolvidas no povoamento de ontologias: Processamento da Linguagem Natural (PLN), Extração de Informação (EI) e Aprendizagem de Máquina (AM). E por fim apresenta as ferramentas utilizadas na aplicação de técnicas de PLN e EI.

Este capítulo está organizado como segue. A seção 2.2 formaliza a definição de ontologia. A seção 2.3 apresenta as principais áreas de conhecimento. A seção 2.4 descreve as ferramentas para a aplicação de técnicas de PLN e EI e finalmente a seção 2.5 apresenta as considerações finais do capítulo.

2.2. Ontologias

As ontologias como estruturas de representação de conhecimento ganharam importância na última década. Atualmente, elas são aplicadas na comunicação de agentes de software [37], na integração da informação [1] [80], na composição de Web Services [73], na descrição do conteúdo para facilitar a sua recuperação [46] [80], no processamento de linguagem natural [59], na Web Semântica [46], na construção de sistemas baseados em conhecimento [70] e nas aplicações de gerenciamento de conhecimento.

Uma ontologia é uma especificação formal explícita de uma conceituação compartilhada de um domínio de interesse [45]. Conceituação refere-se a um modelo abstrato de algum fenômeno do mundo. Explícito significa que o tipo de conceitos utilizados e as limitações do seu uso, são explicitamente definidos. Formal refere-se ao fato de que a ontologia deve ser legível por máquina. Compartilhada reflete a noção de que uma ontologia captura o conhecimento consensual, isto é, não é privada de algum indivíduo, mas aceita por um grupo.

Em [65] define-se ontologia como os termos básicos e os relacionamentos que constituem o vocabulário de uma área temática, bem como as regras para combinar termos e relacionamentos para definir extensões ao vocabulário.

Uma ontologia pode ser definida como uma tupla:

$$O := (C, H, I, R, P, A)$$

onde:

- $C = C_C \cup C_I$ é o conjunto de entidades do domínio sendo modelado. O conjunto C_C é formado por classes, ou seja, conceitos que representam entidades que descrevem um conjunto de objetos (por exemplo, “Person” $\in C_C$) enquanto o conjunto C_I é formado por instâncias, ou seja, entidades únicas no domínio (por exemplo, “Erik Brow” $\in C_I$);

- $H = \{tipo\_de\ (c_1,c_2) \mid c_1 \in C_C \wedge c_2 \in C_C\}$ é o conjunto das relações taxonômicas que definem a hierarquia de classes da ontologia e são denotadas por "tipo_de(c_1,c_2)” indicando que c_1 é uma subclasse de c_2. Um exemplo desse relacionamento é "kind_of(Lawyer,Person)”;

- $I = \{é\_um\ (c_1,c_2) \mid c_1 \in C_I \wedge c_2 \in C_C\} \cup \{prop_K\ (c_i,valor) \mid c_i \in C_I\} \cup \{rel_k\ (c_1,c_2,...,\ c_n) \mid \forall i,\ c_i \in C_I\}$ é o conjunto de propriedades e relacionamentos entre os elementos da ontologia e suas instâncias, por exemplo "is_a(Anne Smith, Client)”, “subject(Case12, adoption)” e “represents(Erik Brow, Anne Smith)”.

- $R = \{rel_k\ (c_1,c_2,...,\ c_n) \mid \forall i,\ c_i \in C_C\}$ é o conjunto de relacionamentos não taxonômicos de uma ontologia. Por exemplo, “represents(Lawyer, Client)”;

- $P = \{prop_K\ (c_i,tipo) \mid c_i \in C_C\}$ é o conjunto de propriedades das

classes de uma ontologia e seu tipo de dados básico. Por exemplo, “subject(Case, string)”;

- A = {$condition_x \Rightarrow conclusion_y$ (c_1,c_2,..., c_n) $\forall j$, $c_j \in C_C$} é um conjunto de axiomas, regras que permitem checar a consistência da ontologia e deduzir novos conhecimentos através de algum mecanismo de inferência. O termo $condition_x$ é dado por: $condition_x$ = { ($cond_1$,$cond_2$,…,$cond_n$) | $\forall_z$, $cond_z \in H \cup I \cup R$}. Por exemplo, “∀Defense_Argument, OldCase, NewCase, applied_to(Defense_Argument, OldCase), similar_to (OldCase, NewCase) $\Rightarrow$ applied_to(Defense_Argument, NewCase)” é uma regra que indica que se dois casos legais são similares, então o argumento de defesa que foi usado em um caso pode ser usado no outro.

Tomemos como exemplo uma ontologia simples que descreve o domínio de um escritório de advocacia, onde os advogados são responsáveis pelos casos de clientes (Figura 02).

Considerando a definição e a ontologia da Figura 02, os seguintes conjuntos podem ser identificados:

- C_C = {person, lawyer, client, case}
- C_I = {Erik Brown, Anne Smith, Case12, Case13, DefenseArgument22}
- H = {kind_of(Person, Lawyer), kind_of(Person, Client)}
- I = {is_a(Erik Brown, Lawyer), is_a(Anne Smith, Client), is_a(DefenseArgument22, DefenseArgument), is_a(Case12, Case), is_a(Case13, Case), subject(Case12, “adoption”), subject(Case13, “adoption”)}
- R = {represents(Lawyer, Client), applied_to(DefenseArgument, Case), develops (Lawyer, Defense_Argument), involved_in(Client, Case)}

- P = {subject(Case, String)}
- A = {∀"Defense_Argument, OldCase,NewCase, applied_to (Defense_Argument, OldCase), similar_to (OldCase, NewCase) ⇒ applied_to (Defense_Argument, NewCase)}

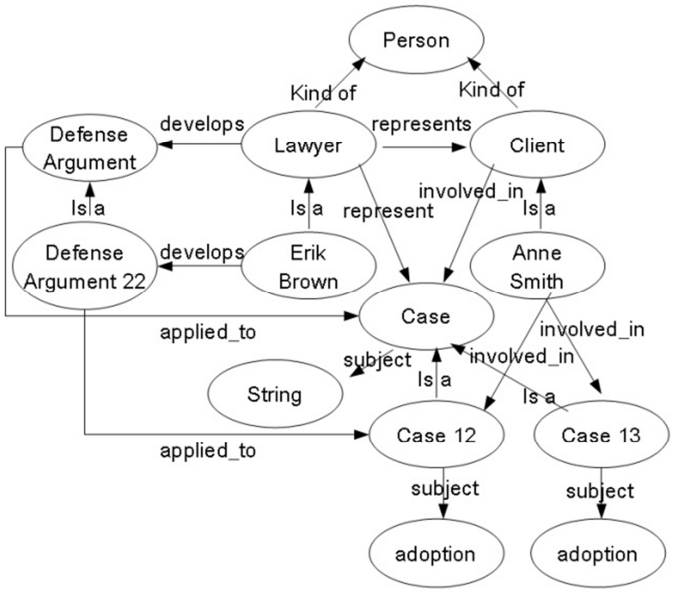


Figura 02: Parte de uma ontologia que descreve alguns elementos pessoais constituintes de uma família

As ontologias podem ser classificadas segundo o seu nível de generalidade [45] ou segundo o seu grau de formalidade [77].

Quanto a seu nível de generalidade as ontologias podem ser: genéricas, de domínio, de tarefa e de aplicação (Figura 03).

Uma ontologia genérica descreve conceitos gerais, tais como, espaço, tempo, matéria, objeto, evento, ação, sendo independentes de domínio.

Uma ontologia de domínio reúne conceitos de um domínio particular e seus relacionamentos, definindo restrições na estrutura e conteúdo do conhecimento desse domínio, por exemplo, o domínio jurídico.

Uma ontologia de tarefa expressa conceitos sobre a resolução de problemas, independentemente do domínio em que ocorram, isto é, descreve o vocabulário relacionado a uma atividade ou tarefa genérica, por exemplo, as técnicas para o acesso à informação.

Uma ontologia de aplicação descreve conceitos dependentes ao mesmo tempo de um domínio particular e de um conjunto de tarefas específicas. Estes conceitos freqüentemente correspondem a papéis desempenhados por entidades do domínio enquanto realizam certas atividades, por exemplo, a aplicação de técnicas para o acesso à informação na área jurídica.

Os conceitos de uma ontologia de domínio ou de uma ontologia de tarefa devem ser especializados dos termos introduzidos por uma ontologia genérica. Os conceitos de uma ontologia de aplicação, por sua vez, devem ser especializações dos termos das ontologias de tarefas e das ontologias de domínio, como mostra a Figura 03 (as setas expressam relacionamentos de especialização).

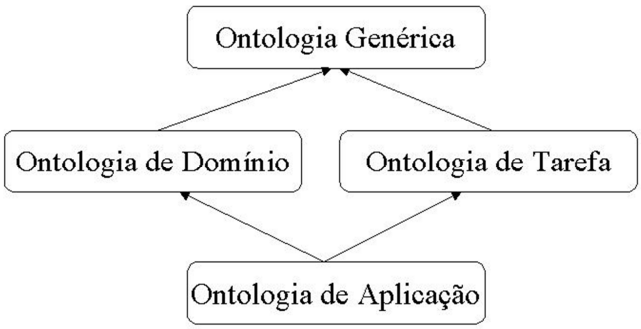


Figura 03: Classificação das ontologias segundo seu nível de generalidade.

Quanto a seu grau de formalidade, uma ontologia pode ser: informal ou formal. Uma ontologia informal é expressa em linguagem natural. Uma ontologia formal é expressa em uma linguagem formal, por exemplo, OWL.

Em particular, o uso de ontologias provê os seguintes benefícios: entendimento, comunicação, representação de conhecimento e reuso.

Quanto ao entendimento, a ontologia pode servir como uma documentação, que as pessoas usam para entender a conceituação de um domínio de interesse; quanto a comunicação, ela pode auxiliar as pessoas na comunicação sobre um domínio de interesse livre de ambigüidades; quanto a representação de conhecimento e reuso, representa um vocabulário de consenso e especifica conhecimento de domínio de forma explícita no seu mais alto nível de abstração com enorme potencial para o reuso. Na próxima subseção será descrito uma visão geral das principais áreas de conhecimento.

2.3. Uma visão geral das principais áreas de conhecimento

2.3.1. Processamento da Linguagem Natural

O PLN é um subcampo da Inteligência Artificial e da Lingüística, que estuda os problemas de processamento e geração das linguagens naturais humanas [50].

Em [24] define-se PLN como um conjunto de técnicas computacionais para analisar e representar textos, em um ou mais níveis de análise lingüística com o objetivo de atingir a maneira humana de processar a linguagem para uso em diversas tarefas ou aplicações.

Dessa forma, o objetivo do PLN é realizar o processamento da linguagem de modo humano. Há mais objetivos práticos para o PLN, muitos relacionados com aplicações específicas para as quais ele está sendo usado. Por exemplo, um sistema de recuperação de informação baseado em PLN tem o objetivo de fornecer a informação mais precisa e completa em resposta a real necessidade de informação dos usuários.

O PLN consiste no desenvolvimento de modelos computacionais para a realização de tarefas que dependem de informações expressas em linguagem natural (e.g. tradução e interpretação de textos, busca de informações em

documentos e interface homem-máquina) [78]. O PLN se torna difícil, pois a linguagem é complexa e ambígua. Por exemplo, considere o seguinte diálogo:

Lúcia: Quero comprar um computador de R$ 1.200,00.

Pedro: Você já tem o dinheiro?

Lúcia: Eu tenho R$ 537,00 no banco e minha mãe me disse que ia dar R$ 680,00. Será que dá?

O raciocínio utilizado para responder esta pergunta ainda não consegue ser reproduzido no computador.

Qualquer abordagem para o PLN requer um maior ou menor grau de conhecimento lingüístico. Os níveis de conhecimento lingüístico são: fonético e fonológico, léxico, morfológico, sintático, semântico, pragmático ou de mundo e de discurso. Pensa-se que os seres humanos normalmente utilizam todos esses níveis, quando produzem ou compreendem uma linguagem, pois cada nível transmite diferentes tipos de significado. Os sistemas de PLN usam diferentes níveis ou uma combinação de níveis de conhecimento lingüístico de acordo com a tarefa que desempenham.

A seguir são apresentados cada um desses níveis acompanhados de alguns exemplos em língua inglesa para facilitar a compreensão.

2.3.1.1. Conhecimento Lingüístico

Nível fonético e fonológico

O nível fonético e fonológico estuda o sistema de sons de uma língua. A fonética está relacionada ao estudo da produção da fala humana, considerando as questões fisiológicas envolvidas, tais como a estrutura do aparelho fonador: mandíbula, laringe, boca, dentes e língua. Essa é uma estrutura bastante complexa, mais de 100 músculos estão envolvidos no controle direto e contínuo da produção das ondas sonoras da fala. Esse é o campo de estudo conhecido como

fonética articulatória. Quando o estudo é mais voltado para as propriedades físicas das ondas sonoras da fala, entramos no campo da fonética acústica [78].

A fonologia é o estudo das regras abstratas e princípios envolvidos na organização, estrutura e distribuição dos sistemas de sons de uma determinada língua. Para se falar sobre os sons da língua é necessário um conjunto de símbolos que representem esses sons, pois a ortografia convencional apresenta problemas do tipo: diferentes sons são associados a uma mesma grafia e, por outro lado, dferentes grafias podem representar um mesmo som [78].

Esse nível de conhecimento lingüístico é a base para o funcionamento de sistemas de reconhecimento e síntese de voz. Em um sistema de PLN de reconhecimento de voz, ou seja, que aceita a fala como entrada, as ondas de som são analisadas e codificadas em sinais digitais para posterior interpretação por meio de regras ou de comparação utilizando um modelo particular de linguagem [24]. Já no processo da síntese da fala em sistemas de PLN ocorre o inverso: uma representação digital é convertida em fala.

Nível lexical

Segundo Gonzalez e Lima [44] “o termo léxico significa uma relação de palavras com suas categorias gramaticais e seus significados”.

Diversos tipos de processamento contribuem para a compreensão ao nível lexical (de palavra) [24]. O primeiro passo geralmente é atribuir um rótulo de parte do discurso (classe gramatical) para cada palavra (nível morfológico). Quando aparece uma palavra que pode ser classificada com mais de um rótulo, é escolhido o rótulo com a maior probabilidade calculada com base nas características do documento em que a palavra ocorre.

Além disso, no nível lexical, as palavras que tem somente um possível sentido ou significado podem ser substituídas pela representação semântica do seu significado.

Nível morfológico

O nível morfológico estuda a estrutura e a formação das palavras. A peculiaridade da morfologia é estudar as palavras olhando para elas isoladamente e não dentro da sua participação na frase.

O nível morfológico consiste em analisar como as palavras são construídas a partir de unidades básicas de significado chamadas morfemas [3]. Algumas palavras, como árvore, não podem ser quebradas em unidades menores, mas isso pode ocorrer com palavras como árvores ou arvorezinhas, por exemplo. Ou ainda palavras como impossível, ou sobremesa. Os morfemas podem ser independentes, como em árvore ou dependentes como no caso dos sufixos (s em árvores) e prefixos (im em impossível) [78].

Por exemplo, a palavra “pré-registration” pode ser morfologicamente analisada em três morfemas separados: o prefixo “pré”, a raiz “registra”, e o sufixo “tion”. Podemos dividir uma palavra desconhecida em seus morfemas constituintes para compreender o seu significado [24]. Por exemplo, adicionar o sufixo “ed” ao verbo em língua inglesa, faz saber que a ação do verbo teve lugar no passado.

Além de estudar a estrutura das palavras, em morfologia estuda-se a classificação das palavras em diferentes categorias, ou, conforme o termo popularmente conhecido na área, as palavras são classificadas em partes do discurso (part-of-speech ou POS). Entre tais categorias encontramos os substantivos (cachorro), verbos (correr), adjetivos (grande), preposições (em), e advérbios (rapidamente). As palavras de uma mesma categoria compartilham várias propriedades em comum como, por exemplo, o tipo de plural (+ s) ou o tipo de diminutivo (+ inho). Os verbos e suas conjugações podem apresentar modificações regulares em vários casos. Na língua inglesa, os adjetivos podem ser acompanhados dos sufixos er e est, como em big, bigger, biggest, significando uma troca de adjetivo comum para um adjetivo comparativo ou superlativo. As categorias de palavras podem ainda ser divididas em classes abertas ou fechadas. As classes abertas são compostas por categorias que abrangem um grande

número de palavras e podem, ainda, abrigar o surgimento de novas palavras. Classes dessas naturezas são os substantivos, verbos e adjetivos. As classes fechadas são aquelas que têm funções gramaticais bem definidas, tais como artigos, demonstrativos, quantificadores, conjunções e preposições [78].

Outra característica compartilhada entre as palavras de uma mesma categoria é a contribuição da palavra para o significado da frase que a contém. Por exemplo, substantivos podem ser usados para identificar um objeto ou conceito determinado, e adjetivos são usados para qualificar esse objeto ou conceito. Ainda a categoria pode dizer algo sobre a posição que as palavras podem ocupar nas frases. As palavras de determinada categoria podem ser usadas como base de um determinado grupo (ou sintagma). Tais palavras são chamadas de núcleo e identificam o tipo de objeto ou conceito que o sintagma descreve. Por exemplo, os sintagmas nominais possuem por núcleo um substantivo (ou nome); em o cachorro, o cachorro raivoso ou em o cachorroraivoso do canil, temos sintagmas nominais que descrevem o mesmo tipo de objeto. Da mesma forma, os sintagmas adjetivais faminto, muito faminto, faminto como um cavalo, descrevem um mesmo tipo de qualidade [78].

Nível sintático

O nível sintático estuda como as palavras podem ser colocadas juntas para formar frases corretas e determina o papel estrutural que cada palavra desempenha na frase [3]. Assim, esse nível foca na análise das palavras em uma sentença de maneira a descobrir a estrutura gramatical da sentença. Isso requer tanto uma gramática, que é um conjunto finito de regras e princípios, quanto um analisador sintático (comumente conhecidos por sua denominação em inglês, “parser”) e cada sentença pode ser armazenada em uma árvore sintática.

Uma gramática define regras válidas para organização das palavras em frases. A ordem em que as palavras aparecem em uma frase é usada para

identificar a composição de constituintes que têm funções bem definidas na frase, como, por exemplo, sujeito e predicado.

As árvores sintáticas são usadas para representar a constituição das frases de acordo com as regras estabelecidas pela gramática.

Por exemplo, na Figura 04 é apresentada a árvore sintática para a frase “The dog chased the cat” (O cão perseguiu o gato). A sentença S é formada pelo sintagma nominal NP (“The dog”) e pelo sintagma verbal VP (“chased the cat”). Os sintagmas nominal e verbal, por sua vez, são formados por outros elementos como artigo (ART), verbo(V), nome(N), etc.

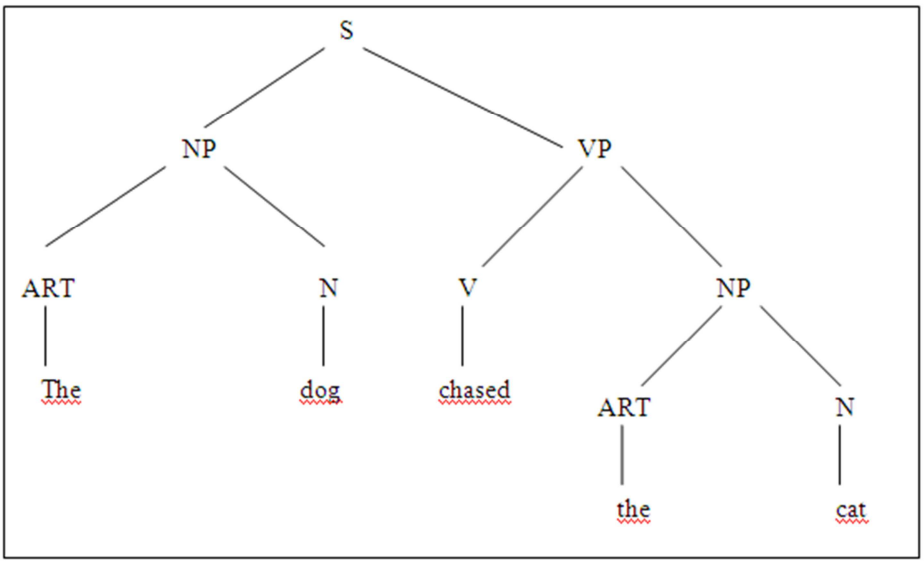


Figura 04: Árvore Sintática

O uso de sintagmas é fundamental para descrição de cadeias permitidas da linguagem, e serve, por exemplo, para identificar objetos do mundo, que em geral são representados por sintagmas nominais. Exemplos: “the dog” e “the cat”.

Nem todas as aplicações de PLN requerem uma análise sintática completa das sentenças, algumas utilizam apenas uma análise superficial ou parcial.

A sintaxe transmite significado [24] já que a mudança na ordem dos elementos de uma sentença pode alterar o sentido da frase. As duas sentenças abaixo diferem somente em termos de sintaxe e transmitem significados completamente diferentes.

“The dog chased the cat” (o cão perseguiu o gato).

“The cat chased the dog” (o gato perseguiu o cão).

Nesse exemplo, percebe-se claramente que a troca na ordem dos termos “dog” e “cat” provocou uma alteração na determinação do agente que sofreu e no que executou a ação. Na primeira frase o cão realiza a ação, enquanto que na segunda o gato é o executor.

Nível semântico

O nível semântico estuda o significado, não só das palavras, mas também do conjunto resultante delas. Assim, o processamento semântico determina os significados possíveis de uma sentença com base nas interações entre os significados das palavras nela contidas [44]. Por exemplo, tendo como entrada para um analisador semântico a seguinte frase: “Edward is dead” (Edward está morto), o analisador semântico poderia apresentar os significados tanto de que Edward está sem vida como que ele está cansado.

Nível pragmático

O nível pragmático estuda o uso intencional da linguagem. Procura obter o significado não literal da frase fazendo uso do conteúdo e contexto do texto e de outros tipos de níveis de conhecimentos lingüísticos mais amplos para a compreensão da mensagem que está no texto. Este nível analisa o uso de sentenças em diferentes situações e como isso afeta a interpretação da sentença [3]. Tenta aproximar o modo como as pessoas interpretam as entrelinhas do que lêem e escutam. O objetivo é explicar como o significado extra é lido em textos sem realmente estar codificados neles. Isto requer muito conhecimento de mundo,

incluindo a compreensão de intenções, planos e objetivos. Algumas aplicações de PLN podem usar bases de conhecimento e módulos de inferência que tentam simular esse comportamento.

Por exemplo, a sentença seguinte requer a resolução do termo anafórico “they”, mas essa resolução exige conhecimento de nível pragmático ou de nível de mundo para ser feita caso o texto de onde foi extraída a sentença não ofereça pistas para a resolução por meio de análise de discurso.

“The city councilors refused the demonstrators a permit because they advocated revolution.” (Os vereadores recusaram aos manifestantes uma autorização, porque eles defendiam a revolução).

No exemplo não se sabe ao certo se o pronome “they” se refere a “city councilors” ou a “demonstrators”. Somente as pessoas que tenham o conhecimento da situação em que a frase foi formulada e usada podem determinar a que termo o pronome se refere.

Nível de discurso

A sintaxe e a semântica trabalham com unidades da sentença, enquanto que o nível de discurso do PLN trabalha com o texto como um todo, ou seja, não interpreta sentenças isoladamente. O nível de discurso faz conexões entre as frases componentes do texto. O nível de discurso trata do fato de que sentenças precedentes afetam a interpretação da próxima sentença [3].

Existem diversos tipos de processamento do nível de discurso: dois dos mais comuns são a resolução de anáfora e o reconhecimento da estrutura do discurso/texto [24].

A resolução de anáfora consiste na substituição de palavras tais como pronomes, que são semanticamente vagos, pela entidade apropriada a que eles se referem. Considere o seguinte exemplo, “The boy sees the girl with your ball. (O

menino vê a menina com sua bola). No exemplo não se sabe ao certo de quem é a bola, ou seja, não se pode afirmar se o pronome possessivo “your” faz referencia ao termo “girl” ou “boy” como o proprietário da bola.

Seria fácil identificar a quem pertence a bola se a sentença estivesse precedida por uma frase esclarecedora, como por exemplo: “Boy looking for his Ball”. Dessa forma, pode-se saber que a bola é do menino (“boy”).

O reconhecimento da estrutura do discurso/texto determina as funções das sentenças no texto, que por sua vez, adicionam ao texto representações significativas. Por exemplo, artigos de jornais podem ser desconstruídos em componentes do discurso tais como história principal, eventos anteriores, avaliações por parte do escritor, citações, etc.

2.3.1.2. Fases do Processamento da Linguagem Natural

A definição de etapas ou fases do PLN se baseia nos níveis de conhecimento lingüístico necessário à compreensão da linguagem natural que foram apresentados acima (fonético e fonológico, léxico, morfológico, sintático, semântico, pragmático e de discurso). Em geral o PLN é composto por etapas de tokenização, análise morfo-lexical, análise sintática, análise semântica e análise contextual.

Dependendo do autor, algumas dessas etapas são subdivididas, omitidas ou agrupadas em uma única etapa. Além disso, podem ser incorporadas novas etapas de acordo com a saída que se pretende obter no final do processamento. Dale, Moisl e Somers [20] definem cinco etapas para o PLN: tokenização, análise léxica, análise sintática, análise semântica e análise pragmática. Já para Cimiano [12], o PLN consiste nas fases (Figura 05) de: pré-processamento, análise sintática, análise semântica e interpretação contextual. A fase de pré-processamento consiste de cinco etapas: tokenização e normalização, POS tagging, análise morfológica, reconhecimento de entidades nomeadas e

resolução de co-referencias. A seguir é apresentada cada uma das etapas de PLN segundo Cimiano.

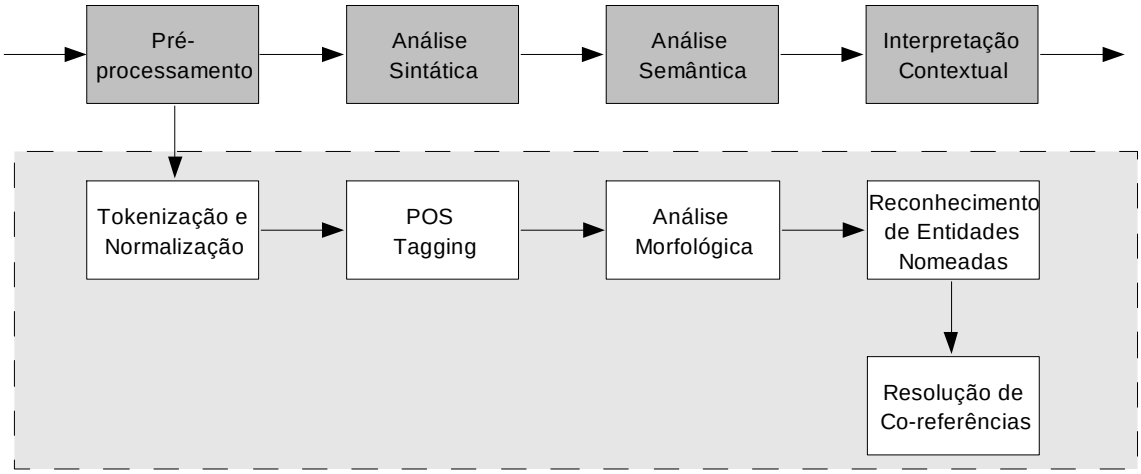


Figura 05: Fases do Processamento da Linguagem Natural [12]

Tokenização e Normalização

A tokenização e a normalização compõem uma única etapa. A tokenização divide o texto em tokens, que são unidades mais simples, como números, pontuação e palavras [18]. Geralmente espaços em branco e sinais de pontuação são usados como delimitadores de palavras. Mas, em se tratando de expressões, que são compostas por duas ou mais palavras, isso não é tão simples. Observe que no nome “São Luís” o espaço não determina o limite da expressão. Por isso, pode ser útil aplicar o reconhecimento de entidades nomeadas (que será discutido mais adiante) antes da tokenização [12].

“São Luís is a city.” (São Luís é uma cidade.)

A frase acima, por exemplo, é composta pelos seguintes tokens: “São Luís”, “is”, “a”, “city” e “.”

Alguns autores como Cimiano [12] incluem a tokenização na atividade de divisão de sentenças, o que particularmente não é considerado conveniente por outros autores [9] [18] [64], sendo preferível ter uma única etapa dedicada à separação de sentenças do texto.

Já a normalização tipicamente consiste em encontrar datas, horários, endereços eletrônicos, números de telefone, abreviaturas, etc. e transformá-las em um formato padrão [12]. É uma tarefa complementar à tokenização, visto que pontos nem sempre delimitam sentenças. Por exemplo: o endereço web “gesec.deinf.ufma.br”.

POS Tagging

O POS Tagging (Part Of Speech Tagging) é responsável por fixar em cada token uma marcação (tag) de sua respectiva participação como componente do discurso, ou seja, sua categoria gramatical como substantivo, adjetivo, verbo, etc [12]. Existe uma variedade de conjuntos de marcação que podem ser utilizadas no POS Tagging. As marcações mais utilizadas são as do conjunto *PennTreebank* [55], mostradas no Anexo A. Por exemplo, a frase: “Bela wrote a sad letter to Edward.” (Bela escreveu uma triste carta a Edward). A marcação desta frase é baseada no conjunto de marcação PennTreebank como mostra a Tabela 1.

Análise Morfológica

A análise morfológica engloba tarefas como a lematização e redução ao radical (“stemming”). A Lematização é um tipo de normalização morfológica que deriva o lema (forma lematizada) da palavra original. Assim, o lema de um verbo é sua forma infinitiva. O lema de uma palavra variável (que não seja verbo) é sua forma singular e quando existir a forma masculina [12]. Por exemplo, a forma verbal do passado “constructed” possui como lema a forma verbal infinitiva “construct”; já o lema das palavras “cat” e “cats” é “cat”.

Um “*stem”* é a parte de uma palavra que é deixada após a retirada de seus afixos (prefixos e sufixos) [4]. Assim, a redução ao radical consiste na extração dos radicais das palavras. Por exemplo, as palavras "fishing", "fished", "fish", and "fisher”, quando aplicadas ao processo de redução ao radical serão reduzidas a um mesmo radical, que no caso é "fish". É importante destacar que a classificação das palavras em categorias gramaticais é perdida após a aplicação da redução ao radical.

Token	**POS Tagging**
Bela	NNP
Wrote	VBD
A	DT
Sad	JJ
Letter	NN
To	TO
Edward	NNP

Tabela 1: Exemplo de POS Tagging com o conjunto de marcação Penn Treebank [55]

Reconhecimento de Entidades Nomeadas

O Reconhecimento de Entidades Nomeadas (“*Named Entity Recognition* – NER”) consiste no reconhecimento de nomes que se referem a objetos exclusivos do mundo [12]. O reconhecimento de entidades nomeadas identifica objetos pertencentes a classes como pessoa, organização, localização, data, etc.

Os sistemas de reconhecimento de entidades nomeadas geralmente utilizam “*gazetteer lists”*, que consistem em listas como cidades, organizações, dias da semana, etc. As listas não são compostas somente por entidades, mas também por nomes de indicadores usuais, tais como designadores de empresa (por exemplo, Ltd.), títulos, etc [57]. Estas listas contêm nomes e seu correspondente tipo, classe ou rótulo. O reconhecimento de entidades nomeadas é feito por meio

de comparações entre as listas e os textos analisados. São exemplos de entidades nomeadas: “Brasil”, “Rio Anil” e “Luís Inácio Lula da Silva”.

Resolução de Co-Referências

A resolução de co-referências abrange problemas em nível de discurso. Os tipos de resolução de co-referências são: nominal [12] e pronominal [18].

A resolução de co-referências nominal tem como objetivo eliminar a diversidade que os nomes apresentam no texto. Por exemplo, quando encontramos em um documento as seguintes expressões “Barack Obama”, “Mr. Barack Obama” e “President Barack Obama”. Tais expressões referem-se à uma mesma entidade do mundo real. A co-referência nominal também é conhecida como co-referencia de entidades nomeadas.

A co-referência pronominal visa descobrir quais são os nomes que são referenciados por pronomes em um documento. Por exemplo, no trecho abaixo a resolução de co-referências pronominal deverá identificar que o pronome “her” está se referindo a Anne, ou seja, que o livro que tem cerca de 50 páginas é de Anne.

“Anne has a book. Her book should be about 50 pages long.” (Anne tem um livro. Seu livro deve ter cerca de 50 páginas.)

Análise Sintática

A análise sintática pode ser do tipo superficial ou parcial (“*chunking*”) ou do tipo completa (“*parsing*”).

A análise sintática superficial ou parcial tem como objetivo descobrir os conjuntos de palavras que, juntas, formam uma unidade sintática (“*chunk*”) [12].

Esta tarefa é importante porque muitos conceitos são expressos em linguagem natural por mais de uma palavra, como ocorre com as frases nominais. A análise sintática superficial também identifica o termo principal da unidade

sintática, que é modificado pelos outros. Numa frase verbal o termo principal é o verbo e em uma nominal é o substantivo. Por exemplo, em “the high school”, “school” é o termo principal.

Resumidamente, a análise sintática superficial visa separar palavras de uma sentença em frases básicas (sintagmas), ou seja, frases nominais ou frases verbais simples.

A análise sintática completa visa revelar a estrutura sintática completa de uma determinada sentença de entrada [12]. Na análise sintática completa é construída uma árvore sintática para cada sentença, de modo a identificar as dependências sintáticas entre as palavras constituintes da sentença. As dependências sintáticas são os relacionamentos de nível sintático entre as palavras. Elas indicam, por exemplo, qual o sujeito e o objeto de um determinado verbo ou qual substantivo é modificado por dado adjetivo.

Análise Semântica

Conforme já citado, a semântica se ocupa com a identificação do significado em nível de sentença. É feito o levantamento dos possíveis significados de uma frase com base na relação entre as palavras que a constituem.

“A área da semântica é uma área de estudo mais complexa que a área da sintaxe, por apresentar questões que são difíceis de tratar de maneira exata e completa” [78]. Isso se deve ao fato de que a mesma frase pode ter significados diversos, o que somente pode ser resolvido por meio da análise situacional feita na etapa de interpretação contextual.

Interpretação Contextual

A Interpretação Contextual leva em conta o fato de que as mesmas palavras podem ter significados diferentes em situações diferentes, ou seja, leva em consideração o contexto no qual as palavras estão inseridas para descobrir seu real significado [12].

Vieira e Lima [78] expressam a dificuldade de se trabalhar com interpretação contextual fazendo o seguinte comentário entre o processamento de textos em nível de discurso (contexto lingüístico) e em nível pragmático (contexto situacional): “O contexto lingüístico é o mais fácil de tratar computacionalmente, pois refere-se ao que é explicitado no texto. [...] É mais difícil tratar computacionalmente o contexto imediato, ou contexto situacional de uma expressão, devido à dificuldade de se chegar a uma representação adequada do conhecimento compartilhado entre os participantes de uma conversação ou comunicação”.

2.3.2. Aprendizagem de Máquina

Os computadores foram criados com o objetivo de realizar tarefas de forma automática. Mas no início, sua programação era estruturada de forma que o modo de sua realização fosse sempre previsível. Desde então, o homem tem se perguntado se os computadores seriam capazes de aprender.

O ato de aprender pode ser definido como a capacidade de aprimorar-se em determinada tarefa por meio da experiência. Embora ainda não seja possível afirmar que os computadores podem aprender como os seres humanos, algumas experiências bem sucedidas em determinadas tarefas de aprendizado, como o reconhecimento de voz [79], têm despertado inúmeras pesquisas sobre o assunto, levando à formalização dos conceitos, métodos e técnicas de aprendizado de máquina.

A AM é uma área da Inteligência Artificial cujo objetivo é o desenvolvimento de técnicas computacionais sobre o processo do aprendizado [7]. Em outras palavras, pode-se dizer que a AM cuida do desenvolvimento de técnicas que possibilitem ao computador melhorar o seu desempenho em determinada tarefa. Além disso, é objetivo da AM a construção de sistemas que sejam capazes de adquirir conhecimento de maneira automática [61].

Cabe ressaltar que aquisição de conhecimento e aprendizado de máquina são processos distintos e que uma não é condição suficiente para que a outra ocorra. Para que haja aprendizado é necessário que o conhecimento adquirido por um sistema interfira positivamente no seu desempenho frente à execução de um determinado conjunto de tarefas. Referente a isto, Mitchell [59] afirma que “um programa aprende, a partir da experiência E, em relação a uma classe de tarefas T, com medida de desempenho P, se seu desempenho em T, medido por P, melhora com E”.

A aquisição do conhecimento pode se dar por meio de três mecanismos: o dedutivo, o indutivo e o analógico. No mecanismo dedutivo o conhecimento é adquirido a partir do que já existe, mas que estava implícito, enquanto que na indução o conhecimento adquirido é intrinsecamente novo, obtido através de generalizações sobre um conjunto particular de exemplos. O mecanismo analógico caracteriza-se pela apropriação de novo conhecimento a partir de uma analogia entre dois problemas e suas respectivas soluções.

As técnicas de AM utilizam o mecanismo indutivo para atualização de sua base de conhecimento. Monard e Baranauskas [60] definem o processo de indução como sendo uma “forma de inferência lógica que permite obter conclusões genéricas sobre um conjunto particular de exemplos”. Com base no exposto, é válido afirmar que na AM um novo conceito é aprendido por meio de inferências indutivas sobre uma coleção de exemplos.

A Figura 06 ilustra um modelo simplificado da aprendizagem de máquina apresentado por Haykin [48], no qual se percebe que o ambiente excita um elemento de aprendizagem fornecendo-lhe alguma informação. O elemento de informação, por sua vez, utiliza a informação recebida do ambiente para atualizar positivamente a base de conhecimento, que será utilizada pelo elemento de desempenho para a execução de sua tarefa.

Neste sentido Russel e Norvig [70] afirmam que “a idéia por trás da aprendizagem é que as percepções não devem ser usadas apenas para agir, mas

também para melhorar a habilidade do agente". A Figura 07 ilustra esta situação, explicitando os processos de percepções e de atuação no ambiente. Diz-se que houve aprendizado se a atuação for melhorada a cada nova situação.

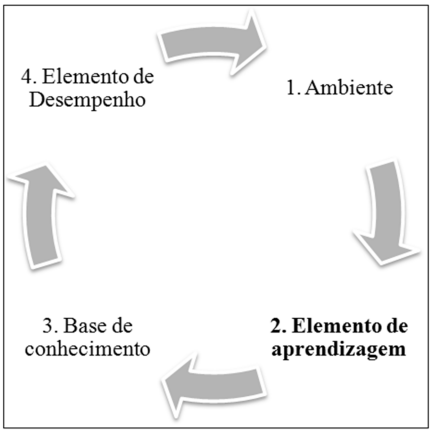


Figura 06: Modelo simplificado da Aprendizagem de Máquina [48]

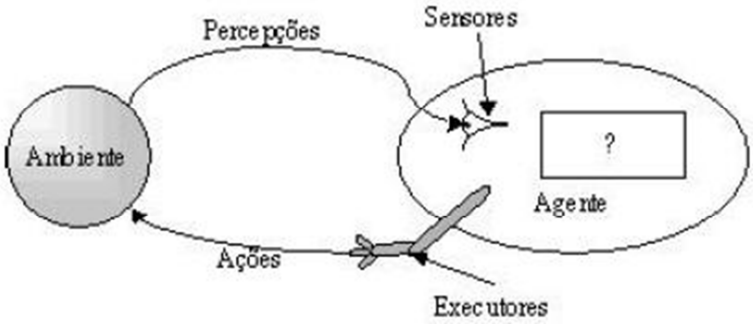


Figura 07: Agente interagindo com o ambiente [70]

O aprendizado indutivo é classificado em supervisionado e não supervisionado. Há, ainda, na literatura uma terceira categoria: o aprendizado por reforço, que é classificado como uma técnica intermediária entre o aprendizado supervisionado e o não supervisionado. A Tabela 2 resume as principais características de cada um desses paradigmas.

Tipo de Aprendizagem	*Caracterização*	*Aplicação*
Supervisionado	Envolve a aprendizagem de uma função a partir de um conjunto de exemplos de suas entradas e saídas.	É utilizado para predizer categorias apropriadas de um exemplo a partir de um conjunto de categorias pré-definidas.
Não supervisionado	Envolve a aprendizagem a partir de vários padrões de entrada da descoberta de semelhanças entre eles e agrupá-los.	É utilizado para encontrar aglomerados de conjuntos de dados semelhantes entre si (*clusters*).
Por Reforço	Envolve a aprendizagem a partir da interação contínua com o ambiente, a realizar uma tarefa pré-determinada com base apenas nos resultados de sua experiência.	É utilizado em tarefas de controle e robótica.

Tabela 2: Classificação das Técnicas de Aprendizagem de Máquina

2.3.2.1. Aprendizagem de Máquina Supervisionada - AMS

A Aprendizagem de Máquina Supervisionada (AMS) envolve a aprendizagem de uma função a partir de exemplos de suas entradas e suas saídas. Um exemplo é um par (x, f(x)), onde x é a entrada e f(x) é a saída da função aplicada a x. O objetivo é aprender uma função matemática que aproxime x de f(x). Sempre é fornecida uma referência do objetivo a ser alcançado, ou seja, o algoritmo de aprendizagem recebe o valor de saída desejado para cada conjunto de dados de entrada apresentado como mostrado na Figura 08.

Segundo [60], “o aprendizado indutivo é efetuado a partir de raciocínio sobre exemplos fornecidos por processo externo ao sistema de aprendizado”. No caso da aprendizagem supervisionada, este conjunto de exemplos é composto geralmente por um vetor de características e pelo rótulo de uma classe associada.

Desta forma, pode-se afirmar que o objetivo do algoritmo de aprendizagem é gerar um classificador capaz de definir corretamente a classe de novos exemplos ainda não rotulados [60].

Pode-se falar em duas categorias para aprendizagem supervisionada: a classificação e a regressão. Chama-se classificação o processo de aprendizado que envolve rótulos de classes discretos. Quando os rótulos de classe são

contínuos, o processo é chamado de regressão. Neste trabalho é dado ênfase ao processo de classificação, que é ilustrado na Figura 09.

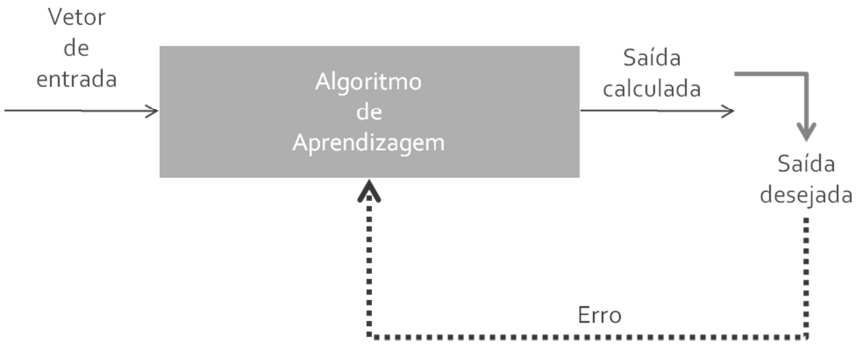


Figura 08: Aprendizagem Supervisionada

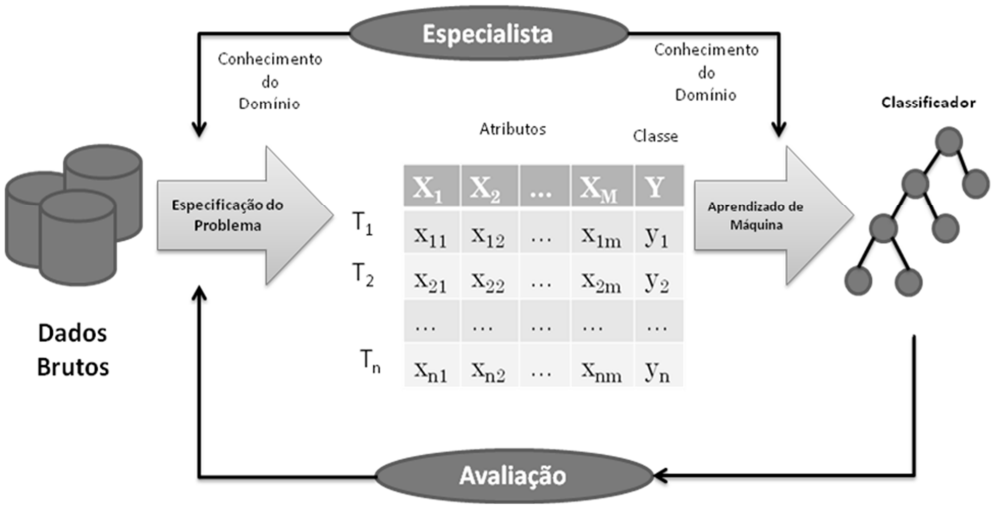


Figura 09: Processo de Classificação [61]

O processo de classificação inicia-se com a especificação do problema e a escolha de um conjunto de exemplos, que posteriormente são divididos em exemplos de treinamento e exemplos de teste. Os exemplos de treinamento (conjunto de valores de atributos e uma classe associada) são submetidos ao algoritmo de aprendizado, que, por sua vez, constrói um modelo para este conjunto de treinamento. Tal modelo representa uma função de aproximação que será capaz

de rotular novos exemplos com uma das classes aprendidas. O classificador induzido, também chamado de hipótese, passa por um processo de avaliação em que é averiguado seu desempenho frente ao conjunto de exemplos de teste. De acordo com o resultado dos testes, é possível realizar modificações na especificação do problema, passando novamente pelo mesmo processo com um novo conjunto de treinamento, gerando um novo classificador.

Segundo [61], uma medida de desempenho utilizada com freqüência é a taxa de erro de um classificador h, que também pode ser chamada de taxa de classificação incorreta e é denotada por err(h). Normalmente, a taxa de erro é calculada através da equação (1), que realiza a comparação entre a classe de cada exemplo com o rótulo atribuído pelo classificador.

$$err(h) = \frac{1}{n}\sum_{i=1}^{n} \| \, y_i \neq h(x_i) \, \| \qquad (1)$$

Um classificador pode também ser avaliado quanto á sua completude e consistência. Completude é a medida da capacidade de uma hipótese classificar todos os exemplos submetidos a teste. Consistência é a proporção de exemplos classificados corretamente. Quanto mais completo é um classificador, mais exemplos ele é capaz de classificar. Da mesma forma, pode-se dizer que quanto mais consistente é um classificador, maior é a porcentagem de exemplos classificados corretamente. Nesta ótica, afirma-se que um classificador completo e consistente (Figura 10.a) classifica todos os exemplos submetidos em sua maioria corretamente. Um classificador incompleto e consistente (Figura 10.b) não classifica todos os exemplos submetidos mas a maioria daqueles que são classificados o são corretamente. Um classificador completo e inconsistente (Figura 10.c) classifica todos os exemplos submetidos, mas alguns incorretamente. Um classificador incompleto e inconsistente (Figura 10.d) não classifica todos os exemplos submetidos e além disso, os classifica incorretamente.

A AMS apresenta dois problemas típicos: “*underfitting*” e “*overffiting*”. O “*underfitting*” ocorre quando um classificador é genérico demais. Neste caso, diz-se que “o classificador ajusta-se muito pouco ao conjunto de treinamento” [61]. Já o

“*overffiting*” ocorre quando o algoritmo de aprendizado induz um classificador muito específico, ou seja, “o classificador ajusta-se em excesso ao conjunto de treinamento” [60].

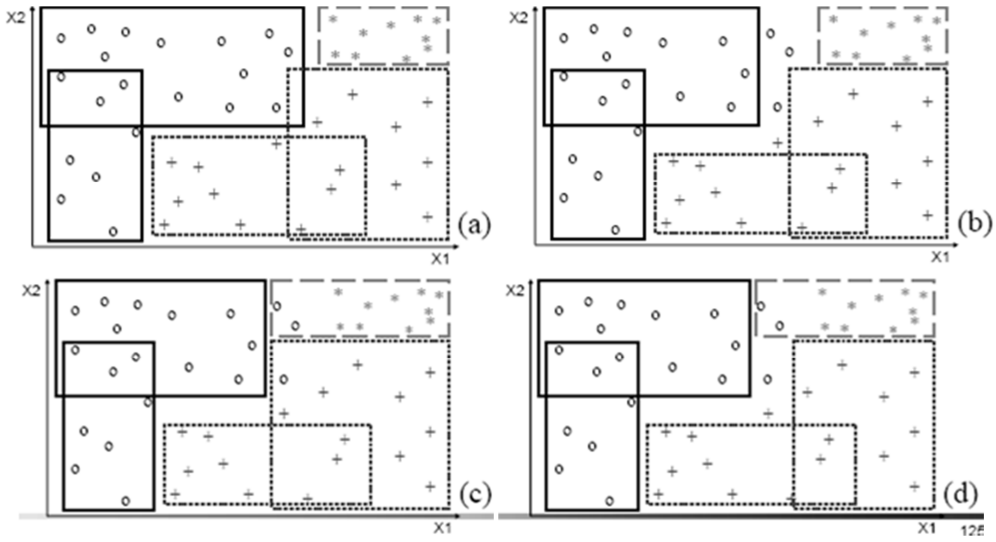


Figura 10: Completude e Consistência de um classificador [61]

O “*underfitting*” pode ser resolvido pela escolha de um conjunto de exemplos mais significativo para o domínio do problema, enquanto que a solução para o “*overfitting*” é a adoção de técnicas de poda, que consistem em aprender um classificador mais genérico a partir do conjunto de treinamento.

Há várias técnicas de AMS como árvores de decisão, redes bayesianas, redes neurais e baseada em instância. A Tabela 3 mostra um comparativo das técnicas de AMS com relação a tempo de treinamento, tempo de classificação, representação de conhecimento, formas de aprendizado e paradigma de aprendizado.

Abordagens	Tempo de Treinamento	Tempo de Classificação	Representação de Conhecimento	Formas de Aprendizado	Paradigma de Aprendizado
Árvores de Decisão	Rápido	Rápido	Simbólico (árvore)	Incremental	Simbólico
Bayesiano	Rápido	Rápido	Numérico (probabilidade)	Incremental	Estatístico
Baseada em Instância	Rápido	Lento	Numérico (distância)	Incremental	Baseado em Exemplos
Redes Neurais	Lento	Rápido	Numérico (pesos)	Incremental	Conexionista

Tabela 3: Quadro comparativo das técnicas supervisionadas

2.3.2.2. Aprendizagem de Máquina Não Supervisionada - AMNS

A Aprendizagem de Máquina Não Supervisionada - AMNS é tipicamente aplicada na exploração de dados. O problema da AMNS envolve a aprendizagem a partir de vários padrões de entrada onde ocorre o agrupamento através da descoberta das semelhanças entre eles. O conjunto de dados de entrada é utilizado para encontrar aglomerados de conjuntos de dados semelhantes entre si (“clusters”).

Agrupamento é uma tarefa de encontrar grupos ou “clusters” de objetos similares. Um bom método de agrupamento deve produzir “clusters” com qualidade, ou seja, alta similaridade intra-classe e baixa similaridade inter-classe. A qualidade do resultado de um processo de agrupamento depende da medida de similaridade, do método utilizado e sua implementação. Além de ser avaliada também pela sua habilidade de descobrir alguns ou todos os padrões escondidos.

As principais abordagens de agrupamento são: algoritmos de partição, algoritmos hierárquicos, algoritmos fundamentados na densidade, algoritmos baseados em grids e os algoritmos baseados em modelos.

Os algoritmos hierárquicos por sua vez são divididos em: aglomerativo e divisivo. O agrupamento hierárquico aglomerativo utiliza uma abordagem bottom-up. Na inicialização é criado um “cluster” próprio para cada elemento. Nas iterações seguintes, os “clusters” mais similares são fundidos. O cálculo da similaridade é

feito entre os “clusters”, através do método de links, que é calculado através da distância. Possui três tipos: o link mínimo, que é a distância mínima entre os “clusters”; link completo, que é a distância máxima entre os “clusters”; e o link médio, que é a distância média entre os “clusters”. O agrupamento hierárquico divisivo utiliza uma abordagem top-down. Particiona um “cluster” universal contendo todos os elementos. O cálculo da similaridade aborda duas questões: como selecionar o próximo “cluster” a ser dividido e como realizar a divisão. Essas questões são abordadas através da função de coerência, onde os elementos menos coerentes são candidatos a deixarem o “cluster” e a função baseada na variância selecionando o “cluster” com maior valor para ser dividido.

2.3.3. Extração de Informação

A EI objetiva localizar a informação relevante em um documento ou em um conjunto de documentos expressos em linguagem natural [17].

Cowie e Wilks [16] definem a EI como qualquer processo que seletivamente estrutura e combina dados de forma explicita ou implícita declarados em um ou mais textos.

Para Cunningham [19], EI é o processo de obtenção de dados quantificáveis desambiguados a partir da linguagem natural, para servir a alguma necessidade de informação precisa e pré-especificada.

Uma tarefa típica da EI é ilustrada na Figura 11 [39] que mostra exemplos de um “template” de um seminário, de um documento e de um “template” preenchido. A EI reconhece o nome (“John Skvoretz”) e o classifica como nome de pessoa. Reconhece o local (PH223D), o horário (4), o título do seminário (“Embedded commitment”) e cria um evento seminário a partir da informação relevante extraída do documento exemplo.

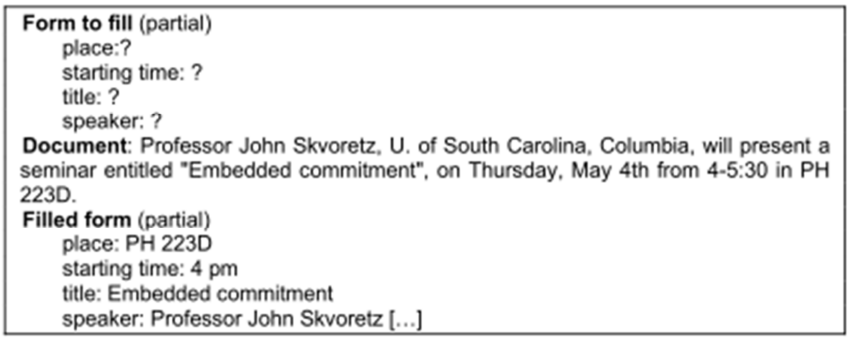


Figura 11: Exemplo de um seminário extraído a partir de um documento

2.3.3.1. Extração de Informação x Recuperação de Informação

A Recuperação de Informação (RI) tem como objetivo selecionar um subconjunto de documentos a partir de uma coleção de documentos baseada em uma consulta [4].

A Extração de Informação tem como objetivo selecionar informação relevante de um documento ou de um conjunto de documentos, através da aplicação de padrões (regras) de extração no documento processado para identificar a informação relevante a ser extraída [19].

A diferença entre a EI e a RI é que a primeira extrai informações específicas e relevantes dos documentos, enquanto que a segunda recupera documentos. Portanto, as duas técnicas são complementares e quando combinadas podem produzir ferramentas interessantes para o processamento de textos [40].

A RI e a EI não diferem somente nos seus objetivos, elas também diferem nas técnicas normalmente utilizadas. As áreas de conhecimento que influenciam a EI são o processamento de linguagem natural e sistemas baseado em regras enquanto que as que influenciam a RI são a teoria da probabilidade e a estatística [40].

2.3.3.2. Abordagens para a Extração de Informação

A escolha da abordagem a ser utilizada em um sistema de extração de informação depende do tipo de texto a ser dado como entrada. Os textos podem ser classificados segundo o seu nível de estruturação: estruturado, semi-estruturado ou desestruturado. O texto estruturado apresenta regularidade no formato de apresentação das informações. Essa regularidade é facilmente compreendida por sistemas de EI, permitindo que cada elemento de interesse seja identificado com base em regras uniformes, que consideram marcadores textuais como delimitadores, e/ou ordem de apresentação dos elementos. Um exemplo de texto estruturado poderia ser um formulário preenchido. Os textos semi-estruturados são aqueles que apresentam alguma regularidade na disposição dos dados. Alguns dados do texto podem apresentar uma formatação, enquanto que outras informações aparecem de forma irregular. É o caso de uma primeira página de um artigo que, em geral, não segue um formato rígido, permitindo variações na ordem e na maneira com que as informações são apresentadas. Por exemplo, quando o artigo tem mais de um autor, os e-mails no mesmo domínio, geralmente são informados de uma vez, separados por vírgula e entre chaves. Os textos desestruturados, por exemplo, textos em linguagem natural, são aqueles que não exibem regularidade na apresentação dos dados. Neste caso, os dados a serem extraídos não são facilmente detectados, a menos que se tenha um conhecimento lingüístico sobre eles.

Tradicionalmente os sistemas de extração de informação utilizam os sistemas baseados em processamento da linguagem natural ou programas extratores (*wrappers*). Os sistemas baseados em processamento da linguagem natural são utilizados quando a entrada são textos semi-estruturados ou desestruturados. Enquanto que os programas extratores são utilizados quando a entrada são textos estruturados ou semi-estruturados. Os sistemas de extração de informação definem regras de extração, que podem ser feitas manualmente, por especialistas de domínio ou com diferentes graus de automação.

Um típico sistema de EI baseado em processamento da linguagem natural (Figura 12) possui três fases: processamento de texto, construção de regras e aplicação de regras [16]. A fase de processamento de texto tem o objetivo de aplicar o PLN em um conjunto de documentos (corpus[1]). A fase de construção de regras tem o objetivo de construir regras de extração a partir da análise de um conjunto de documentos. A fase de aplicação de regras tem o objetivo de extrair informação relevante de um conjunto de documentos processados.

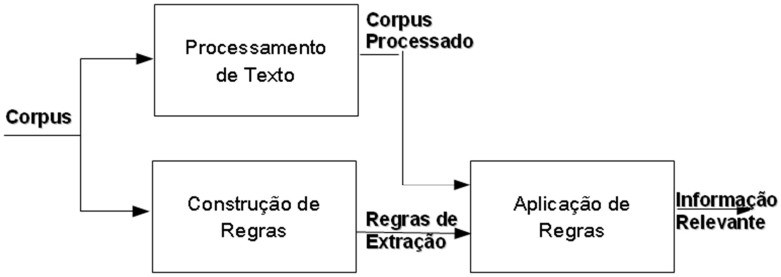


Figura 12: Processo da Extração de Informação

O processamento de texto envolve a aplicação de técnicas de PLN, como tokenização, divisão de sentenças, análise morfo-lexical e resolução de co-referencias já apresentadas na subseção 2.3.1.2.

A fase construção de regras tem o objetivo de desenvolver regras de extração que pode ser feita manualmente por especialistas de domínio ou automaticamente, através de algoritmos de AM.

As regras de extração são usualmente declarativas. A condição é expressa em formalismo baseado em lógica ou na forma de expressões regulares. E a conclusão explora como identificar no texto o valor que preenche o “template”. Por exemplo, a condição é expressa através de uma expressão regular que extrai o que estiver depois da expressão “expression of” e a conclusão explora que o

[1] Corpus (plural corpora) é um conjunto de documentos (textos) em linguagem natural

“Interaction_Target” tem que ser preenchido com o que foi extraído a partir da condição como mostra a Figura 13.

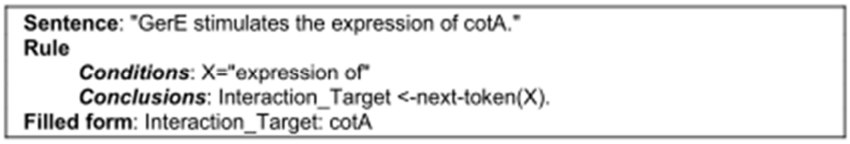


Figura 13: Exemplo de regra de extração

A condição da regra de extração pode checar a presença de um dado item léxico ou a categoria sintática ou a dependência sintática das palavras. Por isso é necessário que a fase processamento de texto seja realizada antes da aplicação das regras de extração.

A fase aplicação de regras tem o objetivo de extrair a informação relevante em um documento ou em um conjunto de documentos através da aplicação das regras construídas na fase anterior.

Existem duas abordagens para a fase construção de regras: a baseada em Treinamento Automático e a baseada na Engenharia de Conhecimento.

A abordagem baseada em Treinamento Automático, como mostra a Figura 14, utiliza técnicas de AM permitindo que o sistema aprenda os padrões (regras) de extração de forma automática. Inicialmente um corpus anotado é submetido a um algoritmo de aprendizagem de máquina para treinamento. Depois do algoritmo treinado, as regras de extração são geradas através de um classificador. Então, o classificador é aplicado em um corpus para que possam ser identificadas as sentenças ou parte delas que casem com as regras de extração geradas. Quando ocorrer o casamento das regras de extração com as sentenças ou parte delas a informação relevante é extraída do corpus. A vantagem dessa abordagem é a geração das regras de forma automática. A desvantagem é o esforço manual na anotação do corpus.

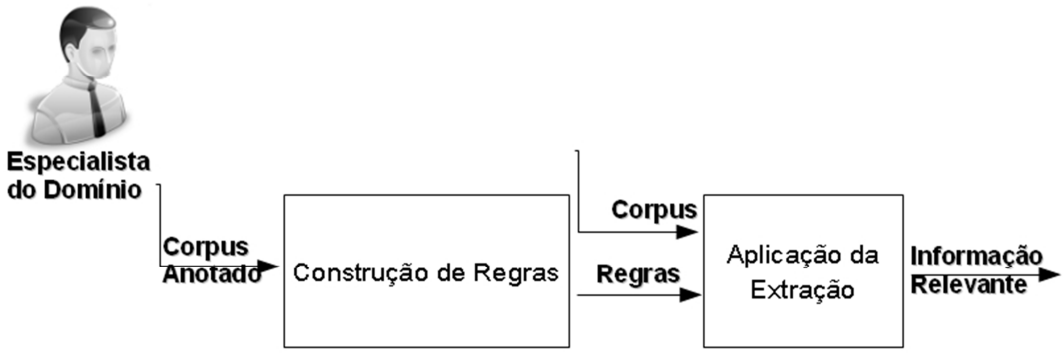


Figura 14: Abordagem baseada em Treinamento Automático

A abordagem baseada na Engenharia de Conhecimento, como mostra a Figura 15, é caracterizada pela construção manual das regras de extração por um especialista de domínio. A construção das regras é realizada através da observação de expressões regulares em um corpus. É um processo iterativo, pois inicialmente o especialista de domínio desenvolve as regras e em seguida ele aplica essas regras no corpus para extrair a informação relevante. Dependendo dos resultados, o especialista de domínio altera as regras e efetua novos testes e esse processo é feito até que o especialista de domínio alcance resultados satisfatórios. Após as regras serem construídas e testadas elas são aplicadas no corpus para identificar as sentenças ou parte delas que casem com as regras de extração. Quando ocorrer o casamento, a informação relevante é extraída do corpus. A vantagem dessa abordagem é a alta precisão das regras de extração, que se deve ao fato delas serem criadas manualmente pelo especialista de domínio. A desvantagem é o esforço manual na construção dessas regras.

A abordagem baseada em Treinamento Automático e a abordagem baseada na Engenharia do Conhecimento exigem um especialista de domínio para a sua aplicação. Na segunda abordagem o especialista de domínio também tem que possuir o conhecimento do formalismo adotado para a representação das regras de extração. Enquanto que na primeira abordagem o trabalho do

especialista de domínio consiste em anotar o corpus de treinamento. É requerido um grande volume de documentos anotados para que seja gerado um classificador com efetividade razoável. A precisão dos resultados obtidos com a segunda abordagem é superior que a precisão dos resultados obtidos com a aplicação da primeira abordagem. Entretanto, o processo de desenvolvimento da segunda abordagem é muito lento, sujeita a erros e com um alto custo.

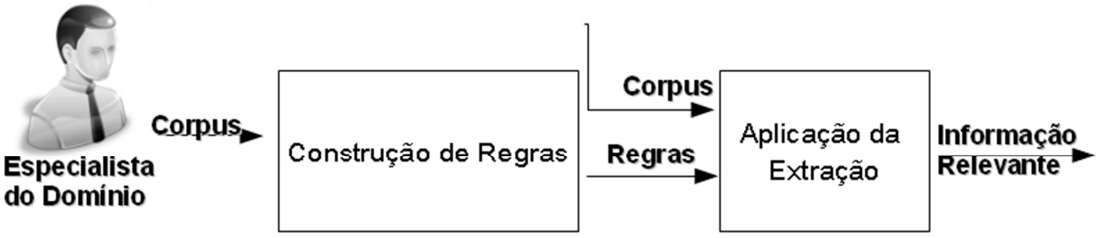


Figura 15: Abordagem baseada na Engenharia do Conhecimento

Os programas extratores (*wrappers*) exploram a regularidade apresentada nos textos estruturados ou semi-estruturados para extrair a informação relevante a partir das regras de extração, previamente definidas. Os wrappers podem ser construídos utilizando a abordagem baseada na Engenharia de Conhecimento ou a abordagem baseada em Treinamento Automático.

2.3.3.3. Extração de Informação baseada em Ontologias

A Extração de Informação baseada em Ontologias é um subcampo da Extração de Informação, que realiza a extração de informação relevante guiada por ontologias a partir de textos desestruturados ou semi-estruturados [81]. As ontologias (seção 2.2) são utilizadas pelos sistemas de extração de informação baseados em processamento da linguagem natural para auxiliar a extração da informação relevante [82]. Estes sistemas utilizam as abordagens baseada em Engenharia de Conhecimento e baseada em Treinamento Automático para a

criação das regras de extração. Na abordagem baseada em Engenharia de Conhecimento (Figura 16) as ontologias são utilizadas pelo especialista de domínio para a criação das regras de extração. Enquanto que na abordagem baseada em Treinamento Automático (Figura 17) as ontologias são utilizadas pelo especialista de domínio para a anotação do corpus.

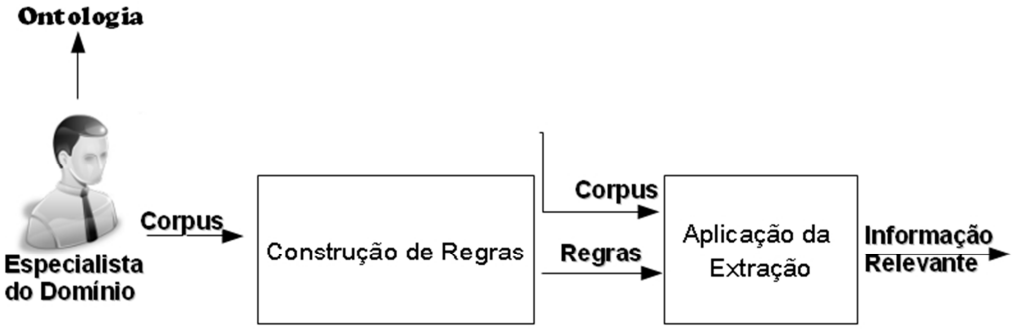


Figura 16: Abordagem baseada na Engenharia do Conhecimento utilizando Ontologia

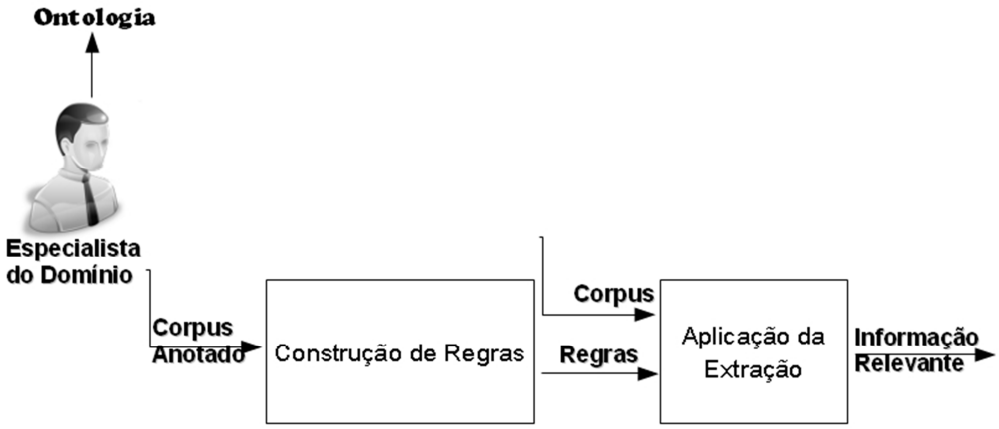


Figura 17: Abordagem baseada em Treinamento Automático utilizando Ontologia

Um sistema de extração de informação baseado em ontologias que utiliza a abordagem da Engenharia do Conhecimento foi proposto por Embley [26]. Este sistema utiliza como entrada textos semi-estruturados e propõe o uso do

Modelo de Sistemas Orientados a Objetos (OSM) [27], onde representa a informação relevante extraída em um banco de dados. O especialista de domínio utiliza uma ontologia para a concepção manual do banco de dados e das regras de extração. Após a aplicação das regras de extração, o banco de dados gerado manualmente é preenchido com a informação relevante extraída. O sistema de extração de informação baseado em ontologia foi avaliado em um estudo de caso que utiliza textos de anúncios de carros como entrada e apresentou um recall de 90% e uma precisão de 98%.

Um sistema de extração de informação baseado em ontologias que utiliza a abordagem de Treinamento Automático foi proposto por Aitken [1]. O especialista de domínio utiliza uma ontologia para anotação do corpus e em seguida são geradas as regras de extração através da Programação em Lógica Indutiva. O sistema foi avaliado e apresentou um recall de 73% e uma precisão de 94%.

O OntoSyphon [58] é um sistema de extração de informação baseado em ontologia. O OntoSyphon recebe como entrada uma ontologia. Cada classe da ontologia é combinada com os padrões de Hearst [49] (seção 2.2.3.5) para gerar consultas em um motor de busca na web. Por exemplo, a classe “mamífero” é combinada com o padrão de Hearst “tais como”, então a consulta “mamífero tais como” é submetida a um motor de busca e para cada par <ocorrência, classe> é calculado a probabilidade através de técnicas estatísticas de co-ocorrência no conjunto de resultados obtidos. Os pares que obtiverem a maior probabilidade são extraídos. O OntoSyphon foi avaliado com ontologias de animal, de comida e de artista e obteve uma precisão de 78%, 93% e 87% respectivamente.

O ontoX [82] é um sistema de extração de informação baseado em ontologia. O ontoX recebe como entrada textos em linguagem natural, a ontologia e palavras-chave associadas às classes e as propriedades da ontologia. As palavras-chave são utilizadas para a geração das regras de extração. Nos textos não é feito nenhum processamento de linguagem natural somente uma exclusão das palavras

vazias. Em seguida, as regras de extração são aplicadas e a informação relevante é extraída. O ontoX foi avaliado com textos de revisões de câmeras fotográficas e obteve uma precisão de 67,8% e um recall de 68,4%. Uma limitação do sistema proposto é a utilização de palavras-chave que influenciam diretamente a efetividade da extração de informação. E a necessidade de um especialista de domínio para definir as palavras-chave associadas as classes e as propriedades da ontologia.

2.3.3.4. Desafios e fatores que influenciam a Extração de Informação

Os fatores que influenciam a precisão da EI são a estruturação e o gênero do corpus. Como apresentado anteriormente o corpus pode ser do tipo estruturado, semi-estruturado ou desestruturado sendo que cada um desses pode apresentar diferentes gêneros textuais requerendo diferentes manipulações.

Os desafios da EI são alcançar altos valores de "*recall*" e *precisão*, escalabilidade e a portabilidade. *Precisão* e "*Recall*" são definidos em termos de um conjunto de informação extraída e o conjunto de informação relevante. *Precisão* é a razão entre o número de informação extraída corretamente (NIEC) e o número de informação extraída (NIE). *"Recall"* é a razão entre o número de informação extraída corretamente (NIEC) e o número de informação no corpus (NIC).

É difícil ter altos valores de *"recall"* e *precisão* simultaneamente, portanto, o que se pretende é ter um equilíbrio entre essas duas medidas.

A escalabilidade pode ser analisada em duas dimensões. Na primeira dimensão, o sistema de EI deve ser escalável na quantidade de dados que é capaz de processar e na segunda dimensão o sistema de EI deve ser escalável em termos do corpus que pode manipular. Geralmente um sistema de EI é projetado para tarefas específicas de domínio.

$$P = \frac{NIEC}{NIE} \qquad (2)$$

$$R = \frac{NIEC}{NIC} \quad (3)$$

2.3.3.5. Padrões Léxicos Sintáticos

Hearst [49] propôs padrões léxicos sintáticos para a descoberta de relações semânticas de hiponímia e de hiperonímia. Hiponímia é relação que denota que um conceito é subclasse de outro conceito, ou seja, um conceito mais específico é chamado de hipônimo. Por exemplo, a relação entre “mãe” e “pessoa” é uma hiponímia em que “mãe” é um hipônimo de “pessoa”. Hiperonímia é a relação inversa a de hiponímia, ou seja, denota que um conceito é uma generalização de outro conceito. O conceito mais genérico é um hiperônimo do mais específico. No exemplo dado “pessoa” é um hiperônimo de “mãe”.

Os padrões léxicos sintáticos identificados por Hearst [49] são utilizados na área da Extração de Informação e podem ser utilizados na área de Povoamento de Ontologias para a identificação de instâncias. A Tabela 4 mostra os padrões propostos por Hearst e exemplos de frases em língua inglesa em que os padrões identificam instâncias da classe “Beach”.

Padrões Léxicos Sintáticos	Exemplos
NP such as {(NP,)*(and – or)} NP	Spend your time at lovely beaches such as Honeymoon Beach, Hawksnest Beach and Coco Beach.
such NP as {(NP,)*{(and – or)} NP	If you just want to relax and have a peaceful day you might such beach as Clovelly, Vaucluse, Coogee, La Perouse and Bronte
NP {,NP}* {,} or other NP	Samara, Nosara or other beach towns
NP, {NP}* {,} and other NP	Samara, Nosara and other beach towns
NP {,} including {NP.}*{and – or} NP	Most tourist visit beaches, including Hermosa Beach, Tamarindo Beach and Coco Beach
NP {,}especially {NP,}*{and – or} NP	Most tourist visit beaches, especially Hermosa Beach, Tamarindo Beach or Coco Beach

Tabela 4: Padrões léxicos sintáticos propostos por Hearst [49]

Por exemplo, na frase do padrão 1 são identificadas as seguintes instâncias: “Honeymoon Beach”, “Hawksnest Beach” e “Coco Beach”; na frase do padrão 2 são identificadas as seguintes instâncias: “Clovelly”, “Vaucluse”, “Coogee, La Perouse” e “Bronte”; nas frases dos padrões 3 e 4 são identificadas as seguintes instâncias: “Samara” e “Nosara”; nas frases dos padrões 5 e 6 são identificadas as seguintes instâncias: “Hermosa Beach”, “Tamarindo Beach” e “Coco Beach”. Na próxima subseção será descrita as ferramentas para a aplicação de técnicas de PLN e EI.

2.4.Ferramentas para a aplicação de técnicas de PLN e EI

2.4.1. General Architecture for Text Engineering - GATE

O GATE[2] (General Architecture for Text Engineering – Arquitetura Genérica para Engenharia de Texto) é uma infra-estrutura para desenvolvimento e implantação de componentes de software que processam a linguagem natural [18]. GATE foi desenvolvido em linguagem Java pela Universidade de Sheffield na Inglaterra.

Essa ferramenta guia os desenvolvedores de três formas:

a) Especificando uma arquitetura genérica para o processamento da linguagem;
b) Fornecendo um framework (conjunto de classes em Java) que implementa a arquitetura;
c) Fornecendo um ambiente gráfico de desenvolvimento (Figura 18).

Arquitetura do GATE é baseada em Componentes conhecidos como Resources (recursos). Os recursos no GATE são diferenciados em três categorias:

a) Recursos de Linguagem (*LanguageResources* – *LR*): são entidades lingüísticas, tais como documentos e corpora.

[2] http://gate.ac.uk/

b) Recursos de Processamento (*ProcessingResources – PR*): são entidades algorítmicas, efetuam alguma forma de processamento. Exemplo: analisadores (parsers), geradores, etc.

c) Recursos Visuais (*VisualResources – VR*): são componentes da Interface Gráfica do Usuário (Graphical User Interface – GUI), que permitem a visualização ou edição de outros componentes.

O conjunto de recursos integrados ao GATE é chamado CREOLE (*Collection of Reusable Objects for Language Engineering – Coleção de Objetos de Engenharia da Linguagem Reusáveis*).

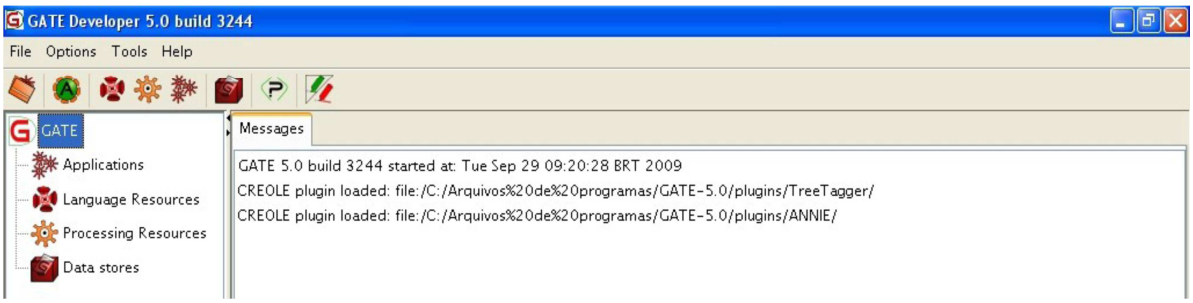


Figura 18: Tela Principal do ambiente gráfico do GATE

Todos os recursos são construídos baseados no modelo de Java Beans – “modelo de componente reutilizável para a linguagem Java e que pode ser manipulado por ferramentas gráficas de forma similar como é feito por outras ferramentas como Delphi e Visual Basic” [62].

Recursos de processamento podem ser combinados em aplicações. Aplicações controlam a execução dos PR através de pipeline. Há duas formas de pipeline:

a) Pipeline Simples: consiste de um grupo de PR que serão executados seqüencialmente em um único documento.

b) Pipeline em corpus: os PR são aplicados a cada um dos documentos do Corpus seqüencialmente.

Existem ainda versões condicionais desses pipelines, onde um PR pode ou não ser executado dependendo de alguma característica do documento.

O GATE suporta documentos nos formatos Plain Text, HTML (Hiper Text Markup Language), XML (eXtensible Markup Language) e RTF (Rich Text Format).

Durante o processamento o GATE realiza anotações nos textos. Essas anotações constituem o resultado do processamento lingüístico realizado pelo framework.

O GATE possui um único modelo de informação que descreve documentos, coleção de documentos e anotações em documentos, baseado em pares de atributos/valores denominados de features. Nomes de atributos são Strings e valores são qualquer objeto Java.

Todos os LR do GATE possuem um conjunto de features que caracteriza o recurso. Com base nisso, [18] apresenta os seguintes conceitos:

a) Corpora: grupo de documentos + features
b) Documento: texto + features + anotações
c) Anotações: consistem de um campo de identificação, um que especifica o seu tipo, um que determina a posição de inicio do token, outro que determina a posição final e um conjunto de características (features) referentes a essa anotação.

Na Figura 19, temos as anotações realizadas sobre a frase *Cyndi savored the soup (Cyndi saboreou a sopa)*. Por exemplo, a anotação com id *1* é do tipo *token*, começa na posição *0* e termina na *5* e possui como feature o atributo *pos* (POS Tagger) com o valor *NP* (nome próprio).

Ao fim de um processamento, o GATE ainda oferece ao usuário as opções de salvar o estado atual dos Processing Resources da aplicação, pela opção Save application state, e de persistir as anotações feitas pelo processamento, o que facilita o trabalho de usuários que necessitam manusear muitas aplicações GATE. Neste último caso, os dados poderão ser armazenados

facilmente no formato de banco de dados (utilizando PostgreSQL ou Oracle) ou em um arquivo de sistema comum, através da opção *Java serialisation*.

Text				
Cyndi savored the soup.				
^0...^5...^10..^15..^20				
Annotations				
Id	Type	SpanStart	Span End	Features
1	token	0	5	pos=NP
2	token	6	13	pos=VBD
3	token	14	17	pos=DT
4	token	18	22	pos=NN
5	token	22	23	
6	name	0	5	name_type=person
7	sentence	0	23	constituents=[1],[2],[3].[4],[5]

Figura 19: Exemplo de resultado das anotações realizado pelo GATE

Fonte: Adaptado de [18]

O GATE dispõe de uma aplicação padrão chamada de ANNIE (a Nearly-New Information Extraction System – Um Quase novo Sistema de Extração de Informação), que é uma aplicação para EI. ANNIE se baseia em autômatos de estados finitos (algoritmo reconhecedor de gramáticas regulares) e na Linguagem JAPE (*Java Annotation Patterns Engine*) assim como os demais recursos do GATE.

O JAPE permite o reconhecimento de expressões regulares sobre o conjunto de anotações, ou seja, o JAPE usa gramáticas restritas Isso agiliza o processamento de grande quantidade de texto.

A aplicação ANNIE padrão (Figura 20) é composta pelos seguintes módulos (recursos):

a) Document Reset: Retorna o documento a seu estado original, apagando todas as anotações. Possui um parâmetro opcional “keepOriginalMarkupsAS”, que permite escolher se as anotações

feitas durante a análise do formato do documento devem ser mantidas.

b) Annie English Tokenizer: Um tokenisador divide o documento em unidades menores chamadas tokens, que podem ser números, pontuações, ou palavras.
c) Gazeteer: São listas contendo um conjunto de nomes, representando entidades como cidades, pessoas, organizações, etc. Um arquivo índice é usado para acessar essas listas, e a cada uma delas é adicionado um tipo e, opcionalmente, um subtipo. Ao processar o documento, o gazeteer cria anotações do tipo lookout para palavras ou expressões que constem em alguma das listas, e a elas adiciona características relativas ao tipo e subtipo.
d) Sentence Splitter: Processa o documento dividindo-o em sentenças. Cada uma delas é anotada com o tipo sentence, bem como cada quebra de sentença é anotada com o tipo Split. Esse processo é necessário para o Tagger. O sentence Splitter é independente de aplicação ou domínio.
e) POS Tagger: Processa o documento, rotulando cada palavra ou símbolo com a respectiva parte do discurso. Utiliza um lexicon e um conjunto de regras. As *tags* utilizadas para rotular são aquelas do conjunto de *tags PennTreebank* [55] e são mostradas no Anexo A.
f) NE Transducer: Contêm regras que atuam em anotações criadas por componentes anteriores, produzindo anotações sobre entidades nomeadas.
g) Orthographic Coreference (OrthoMatcher): Adiciona relações de identidade entre as entidades nomeadas encontradas pelo NE Transducter a fim de realizar co-referência. Não encontra novas entidades nomeadas. Mas, pode classificar corretamente uma entidade não classificada.

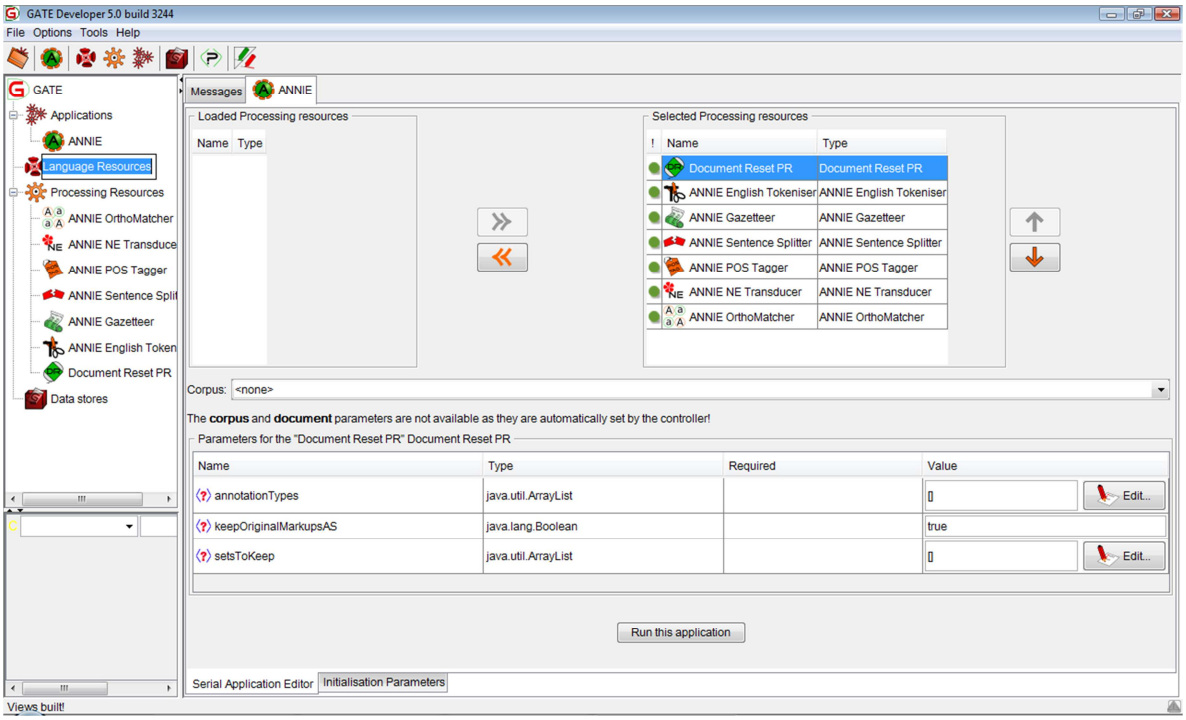


Figura 20: Aplicação padrão ANNIE

Existem outros recursos ANNIE que não são carregados na aplicação padrão e que podem ser adicionados dependendo da necessidade do usuário. Alguns recursos não pertencentes a aplicação padrão são: ANNIE Nominal Coreference, ANNIE Pronominal Coreference e ANNIE VP Chunk.

Além dos recursos da aplicação ANNIE, são disponibilizados pelo GATE muitos outros recursos de processamento, que são descritos em [18]. Abaixo temos uma breve descrição de alguns deles:

a) Pronominal Coreference: executa a resolução da anáfora em termos de co-referencia pronominal usando o formalismo da gramática JAPE. Necessita dos módulos English Tokenizer. Sentence Splitter, NE Transducer e OrthoMatcher para funcionar.
b) GATE Morphological Analyser PR: pode ser encontrado no diretório CREOLE Tools. Ele considera um token e a sua tag da parte do

discurso para identificar o lema e afixo desse token. Estes valores são, então, adicionados como features a anotação de Token.

c) Stanford Parser: É um analisador sintático implementado em Java pelo Grupo de processamento da Linguagem Natural da Universidade de Stanford. Esse parser identifica as dependências sintáticas entre as palavras de acordo com as dependências de Stanford [56], mostradas no Anexo B. Para usá-lo é necessário que os documentos já tenham passado por um tokenizador (ANNIE English Tokeniser) e um divisor de sentenças (ANNIE Sentence Splitter).
d) Stemmer: Esse recurso extrai o radical dos tokens. Cada token é anotado com uma nova feature *"stem"*. Assim, como nos demais PRs do GATE, a aplicação do Stemmer deve ser precedida pela tokenização do documento.
e) Verb Group Chunker: É formado por regras que identificam grupos de verbos em Inglês. É possível descobrir formas verbais no infinitivo, particípio e construções verbais especiais como *"is going to investigate"*. Essas regras são implementadas em JAPE. Essa análise produz anotações do tipo VG com features que representam informações sintáticas sobre o elemento verbal como tipo (type), tempo (tense), voz (voice), etc.
f) Noun Phrase Chunker: Insere marcação de sintagmas nominais no texto analisado. Para funcionar é necessário que os PRs tokeniser, sentence splitter e POS tagger sejam previamente executados.

A relação entre os recursos GATE comentados e as tarefas de PLN é mostrada na Tabela 5.

É importante destacar que os recursos citados na Tabela 05 trabalham sobre o idioma Inglês. Infelizmente, quase não existem recursos no GATE para a Língua Portuguesa.

Existem outras ferramentas semelhantes ao GATE, mas nenhuma apresenta a facilidade de adaptação deste framework.

Tarefas de PLN	***Recurso GATE***
Tokenização	*ANNIE English Tokeniser*
Normalização	*ANNIE Gazetter*
Divisão em Sentenças	*Sentence Splitter*
POS Tagging	*ANNIE POS Tagger*
Lematização	*GATE Morphological Analyser*
Stemming	*Stemmer*
Reconhecimento de Entidades Nomeadas	*NE Transducer*
Co-referência entre Entidades Nomeadas	*ANNIE OrthoMatcher*
Co-referência pronominal	*Pronominal Coreference*
Chunking	*Verb Group Chunker e Noun Phrase Chunker*
Parsing	*StanfordParser*

Tabela 5: Relação entre tarefas de PLN e Plugins do GATE

Pelo fato do GATE ser uma ferramenta que possui um conjunto de módulos bem definidos e reusáveis, capazes de executar tarefas básicas de processamento de linguagem, ele elimina a necessidade do usuário reimplementar algoritmos e módulos inteiros. Além do mais, a ferramenta ainda se responsabiliza pelo processo de armazenamento dos dados e visualização das saídas processadas.

É muito importante observar que o GATE ainda possui uma biblioteca de classes escritas em Java com todos os recursos de processamento presentes na interface gráfica, o que facilita o desenvolvimento de novas aplicações de PLN independentes do ambiente gráfico.

2.4.2. Natural Language ToolKit - NLTK

O NLTK[3] (Natural Language ToolKit – Ferramenta de Linguagem Natural) é uma infra-estrutura para processar a linguagem natural [6]. NLTK foi desenvolvido em linguagem Phyton pela Universidade da Pensilvania.

O NLTK foi desenvolvido com quatro objetivos:

a) **Simplicidade** fornecer um framework intuitivo;
b) **Consistência** fornecer um framework unificado com interfaces e estruturas de dados consistentes;
c) **Extensibilidade** fornecer uma estrutura na qual novos módulos de software possam facilmente incluídos;
d) **Modularidade** fornecer componentes que possam ser utilizados de forma independente.

A relação dos recursos do NLTK e as tarefas de PLN e da AM são mostrados na Tabela 6.

Tarefas de PLN	***Recurso NLTK***	***Tarefas de AM***	***Recurso NLTK***
Tokenização	*NLTK tokeniser*	*Classificação*	*NLTK classify*
POS Tagging	*NLTK tag*		
Stemming	*NLTK stem*	*Clustering*	*NLTK cluster*
Chunking	*NLTK chunk*		
Parsing	*NLTK parse*		

Tabela 6: Relação entre tarefas de PLN e AM e os módulos do NLTK

É importante destacar que os recursos citados na Tabela 06 trabalham sobre o idioma Inglês e Português.

[3] http://www.nltk.org/

Um dos módulos fundamentais do NLTK é utilizado para criar e manipular informações linguísticas estruturadas. Esse módulo inclui árvores - para representação do processamento da análise das expressões; estrutura de traços - para construir e unificar estruturas de valores; gramáticas livres de contexto; e parser - para criar árvores de análise de uma estrutura a partir de uma entrada [6].

Pelo fato do NLTK ser uma ferramenta que possui um conjunto de módulos bem definidos e reusáveis, capazes de executar tarefas básicas de processamento de linguagem, ele elimina a necessidade do usuário reimplementar algoritmos e módulos inteiros.

2.4.3. WordNet

O Wordnet[4] [35] é uma base de dados lexical organizada por significado. Ela foi desenvolvida na Universidade de Princeton por George A. Miller e categoriza conceitos e expressões em língua inglesa em quatro grupos: substantivo, verbo, adjetivo e advérbio. Os itens lexicais são apresentados através de suas diversas definições e suas relações com outros itens lexicais.

A sinonímia é o relacionamento semântico básico do WordNet, ele é expresso por synsets (conjunto de sinônimos). Através de synsets relacionados é formada uma hierarquia lexical entre eles pela hiponímia – relação entre um hiperônimo, mais genérico, e um hipônimo, mais especifico.

No exemplo da Figura 21, os elementos que estão entre chaves são os sinônimos que formam um synset e as setas apontam para os hiperônimos de cada synset.

O WordNet também representa relações parte/todo, denominadas relações de meronímia/holonímia. Por exemplo, bird(pássaro) é holônimo (o todo) de plumage(plumagem) e, conseqüentemente, plumage é merônimo (parte) de bird.

[4] http://wordnet.princeton.edu

O uso de thesaurus como o WordNet vêm fornecendo uma solução para o fenômeno da polissemia, isto é, o fato de que as palavras têm múltiplos significados [43].

O WordNet pode ser usado para apoiar o PAO através do conjunto de hipônimos que são as instâncias disponibilizados por esta base léxica.

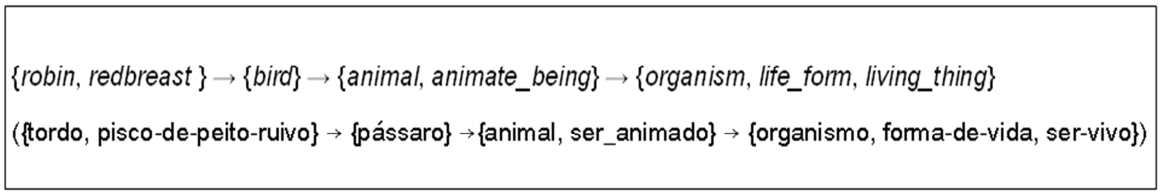


Figura 21: Exemplos de synsets do Wordnet

2.5. Considerações Finais

Neste capítulo foi apresentada a definição formal de ontologia utilizada nesta tese. Foi descrita a visão geral das principais áreas de conhecimento relacionadas ao povoamento automático de ontologias. E finalmente foram também apresentadas as ferramentas utilizadas na aplicação de técnicas de Processamento da Linguagem Natural e de Extração de Informação. Este capítulo forneceu o embasamento teórico para a compreensão desta tese.

No próximo capítulo será discutido o problema do Povoamento Automático de Ontologias e proposto um processo genérico especificando suas fases e quais técnicas podem ser aplicadas para executar as atividades de cada uma das fases. Algumas técnicas do estado da arte do Povoamento Automático de Ontologias são também descritas com as soluções adotadas para cada fase do processo genérico proposto.

3. O Problema do Povoamento de Ontologias

3.1.Introdução

O Povoamento de Ontologias (PO) constitui uma abordagem para automatizar ou semi-automatizar a instanciação de propriedades e de relacionamentos não taxonômicos de classes de ontologias com conhecimento descoberto em diferentes fontes de dados, como documentos textuais. O PO é baseado nas seguintes áreas de conhecimento: Processamento de Linguagem Natural (PLN), Aprendizagem de Máquina (AM) e/ou Extração de Informação (EI) (apresentadas no Capítulo 2). O PO consiste nas seguintes fases: identificação de instâncias candidatas, construção de um classificador e classificação de instâncias. A identificação de instâncias candidatas é realizada através da aplicação de técnicas de PLN e/ou de técnicas estatísticas da área da Recuperação de Informação. A construção de um classificador e a classificação de instâncias são realizadas através da aplicação de técnicas de EI e/ou de técnicas de AM.

Este capítulo descreve o problema do Povoamento Automático de Ontologias (PAO) com um estudo comparativo do estado da arte para situar as contribuições desta tese. São apresentadas também as formas de avaliação do PAO.

Este capítulo está organizado como segue. A seção 3.2 apresenta o problema do PAO. A seção 3.3 descreve um estudo comparativo do estado da arte do PAO. A seção 3.4 apresenta as formas de avaliação do PAO e finalmente a seção 3.5 apresenta as considerações finais do capítulo.

3.2. O Problema do Povoamento de Ontologias

O Povoamento de Ontologias (PO) constitui uma abordagem para automatizar ou semi-automatizar a instanciação de propriedades e de

relacionamentos não taxonômicos de classes de ontologias com conhecimento descoberto em diferentes fontes de dados, como documentos textuais.

As classes em uma ontologia representam os conceitos do domínio e são organizadas em hierarquias. Por exemplo, na área do direito de família, o conceito "*marriage*" é uma classe da ontologia do direito de família, que representa todos os casamentos. A realização desta classe são as instâncias de suas propriedades e relacionamentos não taxonômicos. As propriedades descrevem os atributos das classes e possuem um tipo de dado associado, tais como "string" ou "numérico". Por exemplo, a classe "*marriage*" é descrita através das seguintes propriedades: "*constitutive date*" e "*dissolution date"* ambas com o tipo de dado associado "data". Os relacionamentos não taxonômicos são associações entre as classes. Por exemplo, o relacionamento não taxonômico "*wife member*" ocorre entre a classe "*marriage*" e a classe "*person*".

O problema do PO é extrair um subconjunto I' do conjunto I da definição de ontologia apresentada na seção 2.2. Por exemplo, considere a seguinte frase "*Stuart and Mary married in January 2002 and divorced in November 2010*", as seguintes instâncias são identificadas, extraídas e classificadas: I' = {*wife_member*("*Marriage1", "Mary"*), *husband_member*("*Marriage1", "Stuart*"), *date_constitutive*("*Marriage1",* "01/2002"), *date_dissolution*("*Marriage1"*, "11/2010")}.

A Figura 22 ilustra um processo genérico proposto para o PO com suas entradas, saídas, fases e as técnicas que podem ser aplicadas em cada uma das fases. O PO pode ser realizado através de três fases: identificação de instâncias candidatas, construção de um classificador e classificação de instâncias. Tipicamente, a identificação de instâncias candidatas ocorre através da aplicação de técnicas de PLN [3] e/ou técnicas estatísticas da área da Recuperação de Informação (RI) [71]. A construção de um classificador e a classificação de instâncias são realizadas através da aplicação de técnicas de EI [16] [19] e/ou de técnicas de AM [7]. As entradas do processo são a ontologia a ser povoada e o corpus a ser processado. A saída do processo é a ontologia povoada. A Figura 23

mostra um exemplo simples de PO utilizando um corpus na área do direito de família, onde são ilustrados os resultados das fases do processo proposto na Figura 22.

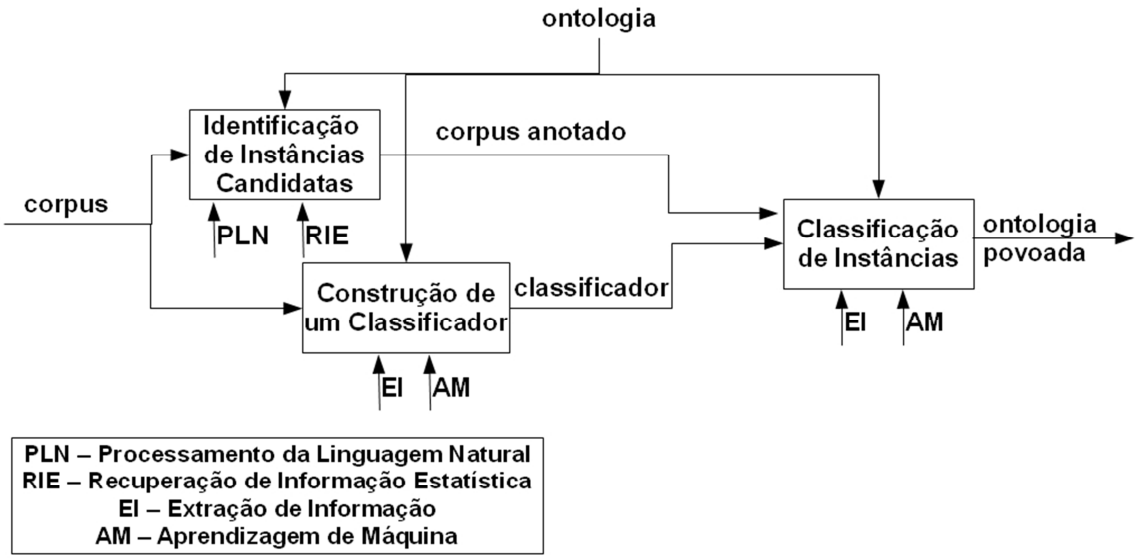


Figura 22: Um processo genérico para o povoamento de ontologias.

3.2.1. Identificação de Instâncias Candidatas

A fase "Identificação de Instâncias Candidatas" é o ponto de partida do PAO, que consiste na detecção de instâncias de propriedades e de relacionamentos não taxonômicos de classes de uma ontologia. Esta fase tem como entrada o corpus e como produto o corpus anotado. Geralmente esta fase está baseada em técnicas de PLN [3], principalmente análise morfo-lexical, reconhecimento de entidades nomeadas e identificação de co-referências. A análise morfo-lexical tem como objetivo identificar a categoria gramatical de cada token na sentença. O reconhecimento de entidades nomeadas consiste na identificação de nomes que se referem a objetos exclusivos do mundo, tais como nomes de pessoas, lugares e organizações dentre outros. A identificação de co-referências identifica tanto as co-referências nominais quanto as co-referências

pronominais. A co-referência nominal consiste de nomes que se referem a uma mesma entidade descrita previamente no texto, enquanto que a co-referência pronominal consiste de pronomes que se referem a uma mesma entidade descrita previamente no texto.

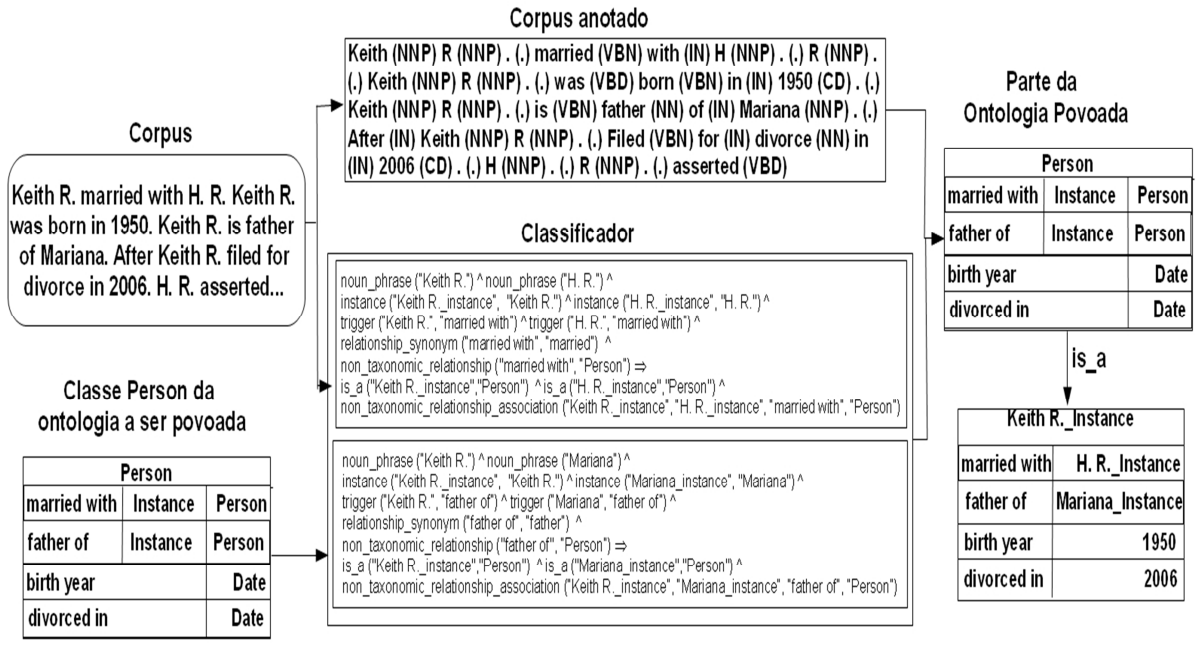


Figura 23: Um exemplo ilustrando a aplicação do processo genérico para o povoamento de ontologias.

Por exemplo, a Figura 23 mostra um exemplo de corpus anotado da área do direito de família produto da fase “Identificação de Instâncias Candidatas” com cinco instâncias detectadas: “Keith R.”, “H. R.”, “Mariana”, “1950” e “2006”.

As técnicas estatísticas da área da Recuperação de Informação (RI) podem ser utilizadas para identificar as instâncias candidatas, através da ponderação dos termos de um documento utilizando as medidas TF-IDF, TF e/ou IDF [71]. Os termos identificados são aqueles cujo peso está dentro de um determinado intervalo. O intervalo e as medidas estatísticas são definidos experimentalmente. Nos experimentos conduzidos são calculados os valores de

recall e de precisão das instâncias extraídas e agrupadas pelas medidas de TF-IDF, TF e IDF, para escolher o intervalo com o maior número de instâncias e o melhor balanço entre as medidas de recall e precisão.

3.2.2. Construção de um Classificador

A fase “Construção de um Classificador” tem o corpus e a ontologia como entrada e como saída o classificador que associa as instâncias identificadas às suas respectivas classes da ontologia. Quando são aplicadas técnicas de EI [19] o classificador gerado é baseado em regras, com a seguinte forma [75]:

$$r_i: (\text{Condition}) \Rightarrow y_r$$

O lado esquerdo da regra é a condição composta de um conjunto de atributos:

$$(A_1 \text{ op } v_1) \wedge (A_2 \text{ op } v_2) \wedge (A_3 \text{ op } v_3) \wedge \ldots (A_k \text{ op } v_k)$$

onde, (A_j, v_j) é um par atributo-valor e op é um operador relacional {=, ≠, <, >, ≤, ≥}. Por exemplo, “noun_phrase (“Keith R.”) ∧ noun_phrase (“H. R.”) ∧ instance (“Keith R._instance”, “Keith R.”) ∧ instance (“H. R._instance”, “H. R.”) ∧ trigger (“Keith R.”, “married with”) ∧ trigger (“H. R.”, “married with”) ∧ relationship_synonym (“married with”, “married”) ∧ non_taxonomic_relationship (“married with”, “Person”)” é a condição que indica que frases nominais precedidas ou seguidas por um trigger são instâncias.

O lado direito da regra é a classe (y_i) à qual a instância pertence. Por exemplo, “is_a (“Keith R._instance”, “Person”) ∧ is_a (“H. R._instance”, “Person”) ∧ non_taxonomic_relationship_association (“Keith R._instance”, “H. R._instance”, “married with”, “Person”)” é a regra que indica a qual classe o relacionamento não taxonômico pertence.

A Figura 23 mostra o classificador baseado em regras gerado a partir do corpus e da classe “Person” da ontologia a ser povoada.

Quando são aplicadas técnicas de AM [7] o classificador é gerado através de um algoritmo de aprendizagem. A geração do classificador inicia-se com a escolha de um conjunto de exemplos, que são divididos em exemplos de treinamento e exemplos de teste. Os exemplos de treinamento (conjunto de valores de atributos e uma classe associada) são submetidos ao algoritmo de aprendizado, que, por sua vez, constrói um modelo para este conjunto de treinamento. Tal modelo representa uma função de aproximação que será capaz de rotular novos exemplos. O classificador é induzido e avaliado. De acordo com o resultado da avaliação é gerado um novo classificador.

3.2.3. Classificação de Instâncias

A fase "Classificação de Instâncias" consiste na aplicação do classificador construído, que associa as instâncias de propriedades e relacionamentos não taxonômicos às suas respectivas classes da ontologia. As entradas desta fase são o corpus anotado, o classificador e a ontologia e como produto a ontologia povoada. Por exemplo, a Figura 23 mostra a classe "Person" povoada como as seguintes instâncias: {married_with("Keith R.", "H. R."), father_of("Keith R.", "Mariana"), birth_date("Keith R.", "1950"), divorce_in ("Keith R.", "2006")}

Na próxima subseção é apresentado um estudo comparativo das principais abordagens do estado da arte do Povoamento Automático de Ontologias.

3.3. Abordagens para o Povoamento Automático de Ontologias

Para cada abordagem do estado arte é apresentado uma descrição geral, incluindo seus objetivos e as soluções particulares adotadas por cada abordagem em cada uma das fases do processo genérico proposto na Figura 22. É também discutido os aspectos positivos e as limitações de cada abordagem.

A abordagem de Fleischman e Hovy [38] tem como objetivo classificar nomes de pessoas em 8 subclasses: atleta, político, artista, advogado, clero, médico, cientista e policial. A fase “Identificação de Instâncias Candidatas” ocorre através da aplicação de técnicas de PLN, como a tokenização, o stemming, POS Tagging e o reconhecimento de entidades nomeadas em um corpus de notícias de jornal. A fase “Construção de um Classificador” é realizada através da aplicação de técnicas de AMS, para a geração do classificador. O recurso utilizado para a indução do classificador é a freqüência dos unigramas, bigramas e trigramas (grupo de uma, duas ou três palavras, respectivamente) que ocorrem em uma janela de três palavras da entidade nomeada. A fase “Classificação de Instâncias” ocorre através da aplicação do classificador induzido, que classifica as instâncias de pessoas. Os autores testaram os seguintes classificadores: árvore de decisão e rede neural. Nos experimentos o classificador que obteve maior precisão (70,4%) foi a árvore de decisão com o algoritmo C4.5, segundo seus autores.

A abordagem de Evans [31] classifica entidades nomeadas. A fase “Identificação de Instâncias Candidatas” ocorre através da aplicação de técnicas de PLN, como a tokenização, o stemming, POS Tagging e o reconhecimento de entidades nomeadas em um corpus com documentos nas áreas do direito, da psicologia, das artes e da literatura. Após a identificação das entidades nomeadas são submetidas consultas a um motor de busca, como Google, usando os padrões de Hearst [49] para identificação dos hiperônimos. Hiperônimo é uma palavra que apresenta um significado mais abrangente do que o do seu hipônimo (termo com sentido mais específico). A fase “Construção de um Classificador” aplica técnicas de EI e técnicas de AMNS, o agrupamento hierárquico, onde os hiperônimos são agrupados para a obtenção dos “clusters”. O Wordnet [35] é utilizado para a escolha do rótulo de cada “cluster”. Na fase “Classificação de Instâncias” cada instância é representada por um vetor de característica que são comparados com os centroids correspondentes de cada cluster para determinar qual classe a instância será classificada. Nos experimentos obteve-se uma precisão de 92,25%, segundo os seus autores.

A abordagem de Tanev e Magnini [76] tem como objetivo o povoamento de ontologia com entidades nomeadas de pessoas e de localizações geográficas. È proposta uma abordagem chamada classe-exemplo. Na fase “Identificação de Instâncias Candidatas” são aplicadas técnicas de PLN, para identificar as instâncias candidatas no corpus. Na fase “Construção de um Classificador” são aplicadas técnicas de AM fracamente supervisionadas, para a geração do classificador. Para a indução do classificador são utilizadas as características sintáticas (por exemplo, são as funções sintáticas da palavra em uma frase: sujeito, objeto e etc) extraídas de um corpus. O conjunto de treinamento é composto de listas de instâncias que podem ser adquiridas de uma ontologia. O algoritmo aprende a partir de um conjunto de treinamento com um único vetor de característica chamado de modelo sintático da classe. Na fase “Classificação de Instâncias” é aplicado o classificador induzido, que identifica a qual classe a instância pertence através da comparação do vetor de característica sintática da classe e da instância. Para avaliar a abordagem baseada em classe-exemplo, ela foi comparada com as abordagens baseada em padrões e baseada em classe palavra. Nos experimentos as três abordagens obtiveram 65%, 18% e 32% para a precisão respectivamente, segundo os seus autores.

A abordagem de Cimiano e Volker [13] classifica entidades nomeadas. A fase “Identificação de Instâncias Candidatas” ocorre através da aplicação de técnicas de PLN, para identificar as instâncias candidatas em um corpus no domínio turístico. A fase “Construção de um Classificador” é realizada através da aplicação de técnicas de AMNS baseada na Hipótese Distribucional de Harry’s [47] e baseada no modelo do espaço vetorial [71], para a geração do classificador, através da construção de um vetor de contexto para as classes e para as instâncias. O contexto é representado por características sintáticas (por exemplo, são as funções sintáticas da palavra em uma frase: sujeito, objeto e etc). A fase “Classificação de Instâncias” ocorre através da aplicação do classificador induzido, que identifica a qual classe que a instância pertence, através da comparação do

vetor de contexto da classe e da instância. Nos experimentos obteve-se uma precisão de 36,82%, segundo os seus autores.

A abordagem de Etizioni et. al. [28] [29] [30] tem como objetivo povoar uma ontologia. A fase “Identificação de Instâncias Candidatas” ocorre através da aplicação de técnicas de PLN, para a identificação de instâncias candidatas no corpus. A fase “Construção de um Classificador” é realizada através da aplicação de técnicas de EI combinadas com técnicas de AMS, para a geração do classificador. Consultas a motores de busca, como Google, são formuladas automaticamente usando padrões de Hearst [49], para calcular o PMI (“Pointwise Mutual Information”) entre palavras e frases. O PMI é número de vezes que a instância aparece na sentença dividida pelo número de vezes que a instância aparece sozinha. Por exemplo, o termo “St. Louis” obteve o valor 4 de PMI, porque ela aparece 20 vezes na sentença “St. Louis is a city” e aparece 5 vezes sozinha. Pares de instâncias e suas respectivas classes com grandes valores de PMI são usados no conjunto de treinamento para gerar o classificador Bayesiano [59]. A fase “Classificação de Instâncias” ocorre através da aplicação do classificador Bayesiano induzido, que classifica as instâncias em classes da ontologia. Etzioni et. al. [28] realizaram vários experimentos medindo os valores de “recall” e de precisão. Nos experimentos reportados o melhor resultado apresentou 90% de precisão e 65% de recall.

A abordagem de Cimiano et. al. [14] classifica entidades nomeadas. A fase “Identificação de Instâncias Candidatas” ocorre através da aplicação de técnicas de PLN, para identificar instâncias candidatas no corpus. A fase “Construção de um Classificador” é realizada através da aplicação de técnicas de EI, que utilizam os padrões de Hearst [49] e outros padrões a partir da observação do especialista de domínio no corpus, para a geração do classificador. A instância é, então, classificada na fase “Classificação de Instâncias” de acordo com o princípio da máxima evidência. Por exemplo, “Nilo” aparece 10 vezes como um país, “Nilo” aparece 50 vezes como um rio, então a partir do princípio da máxima

evidência, será classificado como um rio. Nos experimentos obteve-se uma precisão de 74,37%, segundo seus autores.

A abordagem de Karkaletsis et. al. [51] tem como objetivo o povoamento de ontologias. A abordagem é semi-automática com a intervenção do especialista de domínio. A fase “Identificação de Instâncias Candidatas” ocorre através da aplicação de técnicas de PLN, em particular, o reconhecimento de entidades nomeadas e parsing, para a identificação de instâncias candidatas em um corpus no domínio da Biomedicina. A fase “Construção de um Classificador” é realizada através da aplicação de técnicas de EI, onde o especialista de domínio identifica os padrões léxicos sintáticos, combinadas com técnicas de AMNS, para a geração do classificador. A fase “Classificação de Instâncias” ocorre através da aplicação do classificador e em seguida o especialista de domínio avalia as instâncias classificadas e gera a ontologia povoada.

O sistema ROSA [42] [43] tem como objetivo a instanciação automática de frames (estruturas de representação de conhecimento precursoras das ontologias) para a construção automática das representações internas de elementos de informação no domínio do software. A fase “Identificação de Instâncias Candidatas” ocorre através da aplicação de técnicas de PLN em descrições de software em linguagem natural. O sistema ROSA consiste basicamente de um mecanismo de classificação e um mecanismo de recuperação. O mecanismo de classificação cataloga os componentes de software em uma base de conhecimento de frames através da funcionalidade do software descrita em linguagem natural, ou seja, instancia os frames com a representação interna da funcionalidade dos componentes de software. O mecanismo de recuperação faz a análise de similaridade entre a consulta do usuário e a representação interna dos componentes de software armazenados nos frames. A fase “Construção de um Classificador” é realizada através da aplicação de técnicas de EI com o uso de casos semânticos, para a geração do classificador. Os marcadores de casos semânticos são palavras em linguagem natural que identificam as possíveis instâncias dos slots dos frames. Por exemplo, um frame com o slot “location”, as

palavras que auxiliam a identificação de “location” são “in”, “at”, “into”, “on”, “onto”, “over” e “within”, que são os marcadores do caso semântico “location”. A fase “Classificação de Instâncias” ocorre através da aplicação do mecanismo de classificação, que associa as instâncias a seus respectivos frames.

A abordagem utilizada por Ruiz-Martinez et. al. [69] tem como objetivo o povoamento de ontologias. A fase “Identificação de Instâncias Candidatas” ocorre através da aplicação de técnicas de PLN, em particular, o reconhecimento de entidades nomeadas e “gazetteer lists” (listas de instâncias), para a identificação de instâncias candidatas no corpus no domínio turístico. A fase “Construção de um Classificador” é realizada através da aplicação de técnicas de EI com a construção manual de padrões léxicos sintáticos, para a geração do classificador. A fase “Classificação de Instâncias” ocorre através da aplicação dos padrões léxicos sintáticos, que associa as instâncias as suas respectivas classes. Nos experimentos obteve-se uma precisão de 93,2%, segundo seus autores.

A Tabela 7 mostra uma comparação das abordagens atuais com relação as técnicas/abordagens, as ferramentas, a efetividade, a fonte, o domínio e a automação.

Conforme a análise realizada verifica-se que as abordagens aplicam o PAO através das mesmas fases: Identificação de Instâncias Candidatas, Construção de um Classificador e Classificação de Instâncias. A fase “Identificação de Instâncias Candidatas” é realizada pelas abordagens através da aplicação de técnicas de PLN. A fase “Construção de um Classificador” é realizada pelas abordagens [13] [14] [28] [29] [30] [31] [38] [51] [76] através da aplicação de técnicas de AM, enquanto que as outras abordagens [42] [43] [69] aplicam a EI. A fase “Classificação de Instâncias” é realizada segundo as abordagens [13] [14] [28] [29] [30] [31] [38] [51] [76] através da aplicação do classificador induzido, enquanto que outras abordagens [42] [43] [69] aplicam padrões léxicos sintáticos.

As ferramentas utilizadas pelas abordagens analisadas são ferramentas proprietárias e que não se encontram disponíveis para testes. Impossibilitando assim, uma análise aprofundada de cada ferramenta.

Abordagens / Ano	Técnicas / Abordagens	Ferramentas	Efetividade	Fonte	Domínio	Automação
Fleischman e Hovy / 2002	PLN e AM	MenRun	70,4%	Corpus	Independente de Domínio	Automática
Evans / 2003	PLN, EI e AM	NERO	92,25%	Corpus	Independente de Domínio	Automática
Tanev e Magnini / 2006	PLN, EI e AM	MiniPar	65%	Corpus	-	Automática
Cimiano e Volker / 2005	PLN, EI e AM	Pankow	36,82%	Corpus	Turístico	Automática
Etzioni et. al. / 2005	PLN, EI e AM	KnowItAll	90%	Web	Independente de Domínio	Automática
Cimiano et. al. / 2005	PLN, EI e AM	C-Pankow	74,37%	Web	Independente de Domínio	Automática
Karkaletsis et. al. / 2006	PLN, EI e AM	M-PIRO NLG	-	Corpus	Biomedicina	Semi-Automática
Macedo 2010	PLN e Estatístico	NLPDumper	64,6%	Corpus	Independente de Domínio	Automática
Girardi / 1995	PLN e EI	ROSA	-	Corpus	Software	Automática
Ruiz-Martinez et. al. / 2008	PLN e EI	GATE	93,2%	Corpus	Turístico	Automática

Tabela 7: Quadro comparativo de abordagens para o Povoamento Automático de Ontologias

A efetividade, em termos de precisão, apresentada na Tabela 07 não pode ser considerada para propósitos comparativos entre as abordagens, pois a avaliação de cada uma foi conduzida utilizando diferentes corpora e ontologias. Entretanto os valores apresentados fornecem uma aproximação à efetividade de cada uma das abordagens.

A fonte de informação utilizada pelas abordagens [13] [14] [28] [29] [30] é a Web e as demais abordagens [31] [38] [51] [69] [76] utilizam um corpus em um domínio de aplicação específico.

Das abordagens citadas somente a do Karkaletsis et. al. [51] faz o Povoamento de Ontologias de forma semi-automática. As demais o fazem de forma automática, segundo os seus respectivos autores.

Uma das principais limitações identificadas nas abordagens [13] [43] [51] [69] é o povoamento automático de ontologias apenas em um domínio específico. Uma outra limitação identificada nas abordagens [14] [28] [29] [30] [31] [38] [76] é que o povoamento automático de ontologias é realizado somente com entidades nomeadas.

3.4. Avaliação

Existem três formas de avaliação. Na primeira forma de avaliação a efetividade da classificação de instâncias é medida. Na segunda forma de avaliação a ontologia povoada é comparada com uma ontologia de referência. Na terceira forma de avaliação a ontologia povoada é utilizada em sistemas baseado em conhecimento.

Para avaliar a efetividade do processo de classificação de instâncias é necessária uma adaptação das medidas de precisão e *recall* da área de RI [21] considerado o número de instâncias classificadas corretamente (apresentadas na seção 2.3.3.4.).

Normalmente mecanismos que melhoram o *recall* reduzem a precisão e vice-versa. É desejável ter bons valores de *recall* e precisão logo se torna necessário uma medida que reflita essa combinação. A medida freqüentemente utilizada é a Medida-F que é uma medida harmônica entre o *recall* e a precisão:

$$Medida - F = \frac{(2 * P * R)}{(P + R)} \quad (6)$$

A segunda forma de avaliação é realizada através da comparação da ontologia povoada automaticamente com uma ontologia de referência povoada

manualmente por especialistas de domínio ou engenheiros do conhecimento. A avaliação das instâncias é realizada por especialistas de domínio ou engenheiros do conhecimento de forma manual. Uma vantagem é que os especialistas de domínio e engenheiros de conhecimento são capazes de identificar se a ontologia povoada automaticamente é boa ou não, enquanto que uma desvantagem é que a avaliação é lenta e subjetiva.

Na terceira forma de avaliação a ontologia povoada é utilizada em sistemas baseado em conhecimento. As ontologias povoadas automaticamente são úteis na medida em que melhoram a efetividade dos sistemas nos quais elas são empregadas. Assim, a avaliação da ontologia povoada em uma aplicação executável visa medir a efetividade de um sistema que utiliza as ontologias que estão sendo avaliadas. Por exemplo, experimentos iniciais são reportados em [68]. Uma desvantagem dessa avaliação é que os resultados da avaliação dependem da dependência do sistema sobre a ontologia povoada.

3.5. Considerações Finais

Neste capítulo foi analisado o problema e as principais abordagens para o Povoamento de Ontologias, sendo discutidos os aspectos positivos e as limitações. Foi proposto um processo genérico para o Povoamento de Ontologias com suas entradas, saídas, fases e as técnicas que podem ser aplicadas em cada uma das fases.

Nesta pesquisa o estudo da fundamentação teórica e a definição do problema do povoamento automático de ontologias foram de suma importância visto que a partir disto foram elaboradas e experimentadas técnicas para o povoamento automático de ontologias. A primeira técnica elaborada foi centrada na combinação de técnicas lingüísticas e estatísticas para a atividade de Identificação de Instâncias Candidatas e centrada na AMS para as atividades de Construção de um Classificador e Classificação de Instâncias. Uma avaliação realizada na primeira atividade de Identificação de Instâncias Candidatas apresentou uma

efetividade de 54%. Em busca de um aumento na efetividade foi utilizado consultas ao Wordnet, uma base de dados léxica. Uma segunda avaliação foi realizada e apresentou uma efetividade de 64%. Na atividade Construção de um Classificador e Classificação de Instâncias foram utilizadas como recurso para a construção do conjunto de treinamento as dependências sintáticas. Uma avaliação foi realizada e não apresentou resultados satisfatórios. Em busca de uma melhoria nos resultados obtidos foi utilizada as vantagens da área de conhecimento de EI.

Com o conhecimento adquirido com o estudo da EI elaboramos e experimentamos uma técnica para o PAO centrada no PLN para a atividade Identificação de Instâncias Candidatas e centrada na EI para as atividades Construção de um Classificador e Classificação de Instâncias. Uma avaliação foi realizada e apresentou uma efetividade de 90%. Os resultados obtidos foram satisfatórios.

A análise comparativa das principais abordagens do estado da arte foi fundamental para a identificação das vantagens e limitações de cada abordagem. A abordagem de Fleischman e Hovy [38], de Evans [31] e de Tanev e Magnini [76] realizam somente o povoamento de ontologias com entidades nomeadas, ou seja, uma ontologia que não é específica de um domínio. A abordagem de Cimiano e Volker [13] realiza o povoamento de ontologias no domínio turístico com o reconhecimento de entidades nomeadas, ou seja, é dependente de um domínio específico. As abordagens de Karkaletsis et. al. [51] e Ruiz-Martinez et. al. [69] realizam o povoamento de ontologias de forma semi-automática e dependente de um domínio específico, biomedicina e turístico respectivamente. O sistema ROSA [42] [43] realiza o povoamento automático de frames, mas é dependente de um domínio específico, o software.

A partir dessas limitações constatou-se uma necessidade da formalização de um Processo Independente de Domínio para o Povoamento Automático de Ontologias. O processo proposto será apresentado no próximo capítulo.

4. DIAOP-Pro - Um Processo Independente de Domínio para o Povoamento Automático de Ontologias a partir de texto

4.1. Introdução

Este capítulo descreve um Processo Independente de Domínio para o Povoamento Automático de Ontologias (DIAOP-Pro) que utiliza técnicas de PLN [3] [20] e de EI [16] [19]. O DIAOP-Pro, principal contribuição desta pesquisa, se constitui em uma nova abordagem uma vez que propõe o povoamento automático de ontologias utilizando uma ontologia para a geração automática de regras para extrair instâncias a partir de textos e classifica-as como instâncias de classes da ontologia. Estas regras podem ser geradas a partir de ontologias específicas de qualquer domínio, tornando o DIAOP-Pro independente de domínio.

DIAOP-Tool uma ferramenta para o Povoamento Automático de Ontologias foi desenvolvida para automatizar o processo DIAOP-Pro. A DIAOP-Tool processa corpus em língua inglesa e povoa ontologias OWL.

Uma avaliação do processo DIAOP-Pro (Capítulo 5) foi realizada através da condução de quatro estudos de caso nos domínios jurídico e turístico de modo a demonstrar a sua efetividade e viabilidade.

Este capítulo está organizado como segue. A seção 4.2 descreve o processo proposto com um exemplo no domínio do direito de família. A seção 4.3 apresenta a ferramenta desenvolvida que suporta o processo proposto e a seção 4.4 apresenta as considerações finais do capítulo.

4.2. O Processo

O Processo DIAOP-Pro consiste de três fases (Figura 24): “Identificação de Instâncias Candidatas”, “Construção de um Classificador” e “Classificação de Instâncias”. A Figura 24 ilustra as técnicas aplicadas nas fases do processo: PLN e EI.

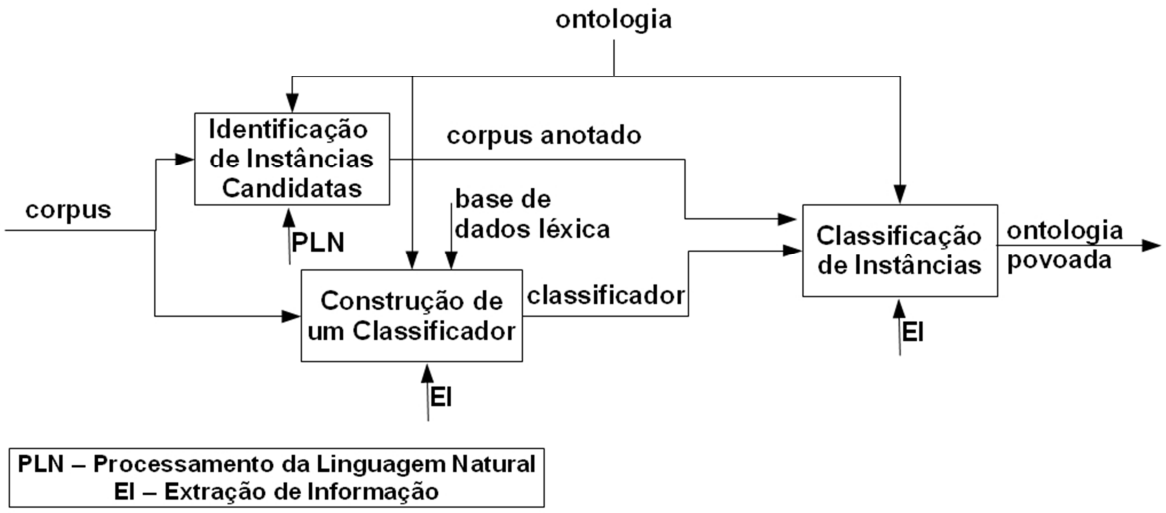


Figura 24: Um processo para o povoamento automático de ontologias

A fase “Identificação de Instâncias Candidatas” aplica técnicas de processamento da linguagem natural para identificar instâncias de relacionamentos não-taxonômicos e propriedades de uma ontologia através da anotação do corpus que é dado de entrada. Nesta fase foi escolhida a aplicação de técnicas de PLN por serem efetivas na utilização de corpus de qualquer tamanho como mostra a avaliação realizada no primeiro estudo de caso no Capítulo 5. A Figura 25 mostra um exemplo de corpus anotado no domínio do Direito de Família produto da fase “Identificação de Instâncias Candidatas” com as seguintes instâncias detectadas: “Keith R.”, “H. R.”, “Mariana”, “1950” e “2006”.

A fase “Construção de um Classificador” aplica técnicas de extração de informação para construir um classificador baseado em um conjunto de regras lingüísticas a partir de uma ontologia e de consultas a uma base de dados léxica. Nesta fase foi escolhida a aplicação de técnicas de EI, pois temos que gerar “n” classificadores específicos de domínio no momento da execução do processo. Se fossem aplicadas técnicas de AM não como teríamos gerar e avaliar “n” classificadores específicos de domínio. Esta fase tem como entrada o corpus e a ontologia e como produto o classificador usado na fase “Classificação de Instâncias”, que associa as instâncias às classes da ontologia. Por exemplo, a

Figura 25 mostra um exemplo de classificador gerado a partir do corpus e da classe “Person” da ontologia a ser povoada.

A fase “Classificação de Instâncias” usa o classificador gerado pela fase “Construção de um Classificador”, que associa as instâncias de relacionamentos não-taxonômicos e propriedades às suas respectivas classes da ontologia. Esta fase tem como entrada o corpus anotado, o classificador e a ontologia e como produto a ontologia povoada. Por exemplo, a Figura 25 mostra a classe “Person” povoada da ontologia do Direito de Família com as seguintes instâncias: {married_with(“Keith R.”, “H. R.”), father_of(“Keith R.”, “Mariana”), birth_year(“Keith R.”, “1950”), divorced_in (“Keith R.”, “2006”)}.

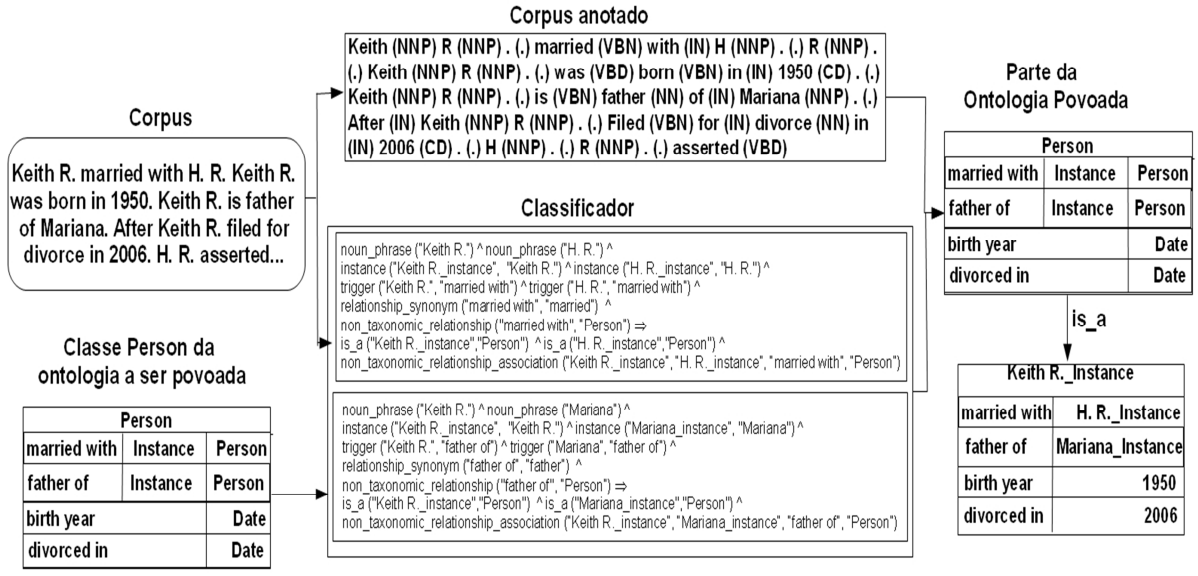


Figura 25: Um exemplo ilustrando a aplicação do processo genérico para o povoamento de ontologias

As próximas seções detalham cada uma das fases, tarefas e atividades do processo, ilustradas através de exemplos utilizando como entrada um corpus no domínio do Direito de Família (Figura 26), que por razão de simplicidade tem apenas um parágrafo em língua inglesa.

Keith R. married with H. R. They married in 2004. Keith R. was born in 1950. Keith R. is father of Mariana. After Keith R. filed for divorce in 2006. H. R. asserted domestic violence allegations against Keith R., and requested sole custody of Daughter. Following an investigation and a hearing, the court (Judge Claudia Silbar) denied Mother's requests. In February 2007, the court (Judge Pollard) entered an order granting both parents joint legal and physical custody, and appointed a child custody evaluator, who recommended maintaining the current custody arrangements based on Daughter's parental attachments.

Figura 26: Fragmento de texto de um documento no domínio do Direito de família [54]

4.2.1. Identificação de Instâncias Candidatas

A fase "Identificação de Instâncias Candidatas" (Figura 27) consiste de três tarefas: "Análise Morfo-Lexical", "Reconhecimento de Entidades Nomeadas" e "Identificação de Co-Referências". Esta fase tem como entrada o corpus e como produto o corpus anotado.

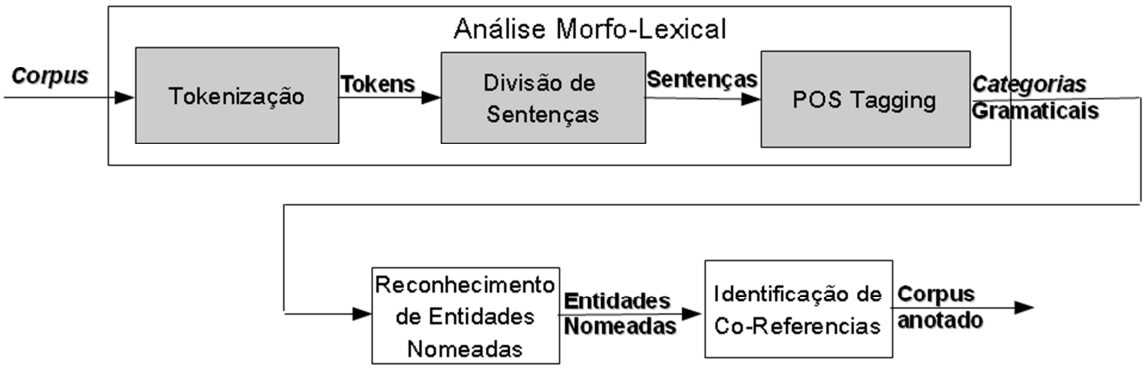


Figura 27: A fase "Identificação de Instâncias Candidatas" do processo proposto

A tarefa "Análise Morfo-Lexical" consiste nas seguintes atividades: "Tokenização", "Divisão de Sentenças" e "POS Tagging". A "Tokenização" divide o texto em tokens, unidades mais simples, como números, pontuação e palavras. A "Divisão de Sentenças" consiste em dividir o texto em parágrafos. O "POS Tagging" é responsável por fixar em cada token uma marcação de sua respectiva participação como componente do discurso, ou seja, sua categoria gramatical, por

exemplo, substantivo, adjetivo, verbo, dentre outros. Por exemplo, o resultado da tarefa “Análise Morfo-Lexical” para o fragmento de texto da Figura 26 é mostrado na Figura 28. As marcações utilizadas neste exemplo são aquelas do conjunto de tags Penn Treebank [55].

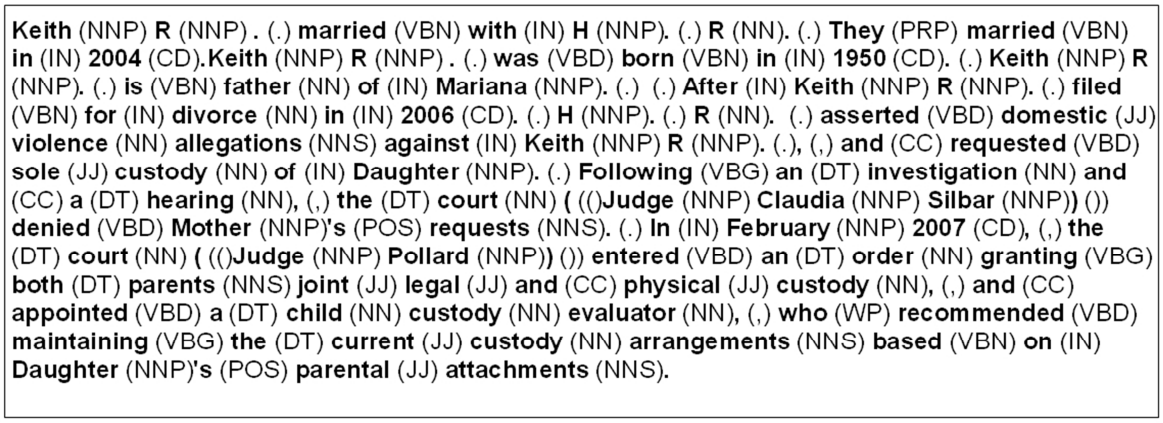


Figura 28: Resultado da tarefa “Análise Morfo-Lexical” para o fragmento de texto da Figura 26.

A tarefa “Reconhecimento de Entidades Nomeadas” tem como objetivo detectar nomes que se referem a objetos exclusivos do mundo, prováveis instâncias de propriedades e relacionamentos não taxonômicos da ontologia. O resultado da tarefa “Reconhecimento de Entidades Nomeadas” para o fragmento de texto da Figura 26 é um conjunto de três entidades do tipo “Pessoa”: “Keith R.”, “Mariana” e “H. R.”.

A tarefa “Identificação de Co-Referências” tem como objetivo a detecção de co-referências pronominais e nominais, ou seja, nomes ou pronomes que se referem a uma mesma entidade descrita previamente no texto. O resultado da tarefa “Identificação de Co-referencias” para o fragmento de texto da Figura 26 é o pronome “They” que se referencia aos nomes “Keith R.” e “H. R.”.

4.2.2. Construção de um Classificador

A fase “Construção de um Classificador” (Figura 29) é realizada de forma independente da primeira fase, “Identificação de Instâncias Candidatas”, e consiste de três tarefas: “Seleção de Classes, Propriedades e Relacionamentos”, “Seleção de Triggers” e “Geração de Regras”. Esta fase tem como entradas a ontologia e o corpus e como produto o classificador.

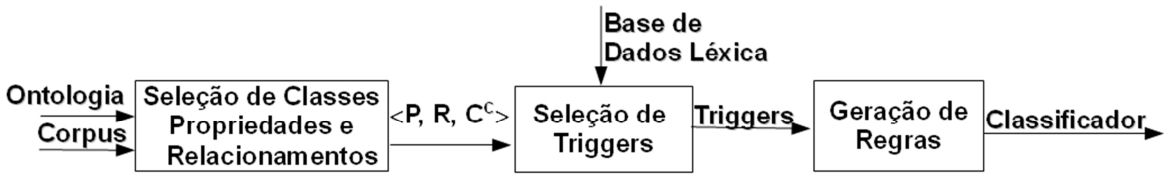


Figura 29: A fase “Construção de um Classificador” do processo proposto

A tarefa “Seleção de Classes, Propriedades e Relacionamentos” é realizada a partir de uma ontologia da qual são selecionados as classes (C^C), propriedades (P) e os relacionamentos não-taxonômicos (R). Por exemplo, a Figura 30 mostra a classe “Person” com as propriedades: “birth_year”, “divorce” e “constitutive” e os relacionamentos não taxonômicos “married” e “father”, como exemplo do produto desta tarefa.

Instâncias são encontradas em frases nominais precedidas ou seguidas por um trigger. Um trigger consiste de uma ou mais palavras que sugerem a presença da instância. Por exemplo, a frase nominal “John” seguida pelo trigger “married with” sugere a presença da instância “John”. Na tarefa “Seleção de Triggers” para cada sinônimo na base de dados léxica de relacionamento não taxonômico (R) ou de propriedade (P) da ontologia um trigger é gerado. Por exemplo, “married”, “marry” e “wed” são sinônimos e correspondem aos triggers do relacionamento não taxonômico “married_with”, da ontologia do direito de família. O

produto da tarefa “Seleção de Triggers” para a classe “Person” da ontologia do direito de família é ilustrada na Figura 31.

Person		
married	Instance	Person
father	Instance	Person
birth_year		Date
divorce		Date
constitutive		Date

Figura 30: Classe “Person” da Ontologia do Direito de Família

Property	Trigger	Non Taxonomic Relationship	Trigger
Birth_Year	Born	Married	Married
	Birth		
Divorce	Divorce		Marry
	Divorcement		
	Disjoint		Spouse
	Disunite		
	Dissociate	Father	Father
	Disassociate		
Constitutive	Constitutive		Male Parent
	Marriage		
	Matrimony		Begetter
	Marriage		

Figura 31: Triggers identificados da classe “Person” da ontologia do direito de família.

A tarefa “Geração de Regras” tem como objetivo a especificação de regras de classificação com a seguinte forma:

se <condição> então <conclusão>

A condição é composta de cinco predicados unidos por uma conjunção lógica:

- Frase_nominal (X), representa uma frase nominal X (por exemplo, frase_nominal (“John”));

- Instância (I, X), representa uma instância I a ser classificada de uma frase nominal X (por exemplo, instância (“John_instância”, “John”) representa a instância “John_instância” da frase nominal “John”);
- Trigger (X, Y), representa um trigger Y de uma frase nominal X (por exemplo, trigger (“John”, “married”) representa o trigger “married” da frase nominal “John”);
- Relacionamento_sinônimo (Y, R) representa um sinônimo Y de um relacionamento não-taxonômico R (por exemplo, relacionamento_sinônimo (“married”, “married”) representa o sinônimo “married” do relacionamento não-taxonômico “married”);
- Propriedade_sinônimo (Y, P) representa um sinônimo Y de uma propriedade P (por exemplo, propriedade_sinônimo (“born”, “birth_year”) representa o sinônimo “born” da propriedade “birth_year”);
- Relacionamento_não_taxonômico (R, C) representa um relacionamento não-taxonômico R de uma classe C (por exemplo, relacionamento_não_taxonômico (“married”, “Person”) representa o relacionamento não-taxonômico “married” da classe “Person”);
- Propriedade (P, C, V) representa uma propriedade P de uma classe C com um valor V (por exemplo, propriedade (“birth_year”, “Person”, “1950”) representa a propriedade “birth_year” da classe “Person” com um valor de “1950”).

A conclusão é composta de dois predicados unidos por uma conjunção lógica:

- É_um (I, C) representa uma classe C onde a instância I pode ser classificada (por exemplo, é_um (“John_instância”, “Person”) representa a classe “Person” onde a instância “John_instância” pode ser classificada);

- Relacionamento_não_taxonômico_associação (I_1, I_2, R, C) representa instâncias I associadas por um relacionamento não-taxonômico R de uma classe C (por exemplo, relacionamento_não_taxonômico_associação (“John_instância”, “H. R._instância”, “married”, “Person”) representa as instâncias “John_instância” e “H. R._instância” associadas pelo relacionamento não-taxonômico “married” da classe “Person”);
- Propriedade_associação (I, P, V, C) representa uma instância I com um valor V para uma propriedade P de uma classe C (por exemplo, propriedade_associação (“John_instância”, “birth_year”, “1950”, “Person”) representa a instância “John_instância” com um valor de “1950” para a propriedade “birth_year” da classe “Person”).

$$\forall X_1, X_2, Y, I_1, I_2, R, C \mid \text{frase\_nominal}(X_1) \wedge \text{frase\_nominal}(X_2) \wedge \text{instância}(I_1, X_1) \wedge \text{instância}(I_2, X_2) \wedge \text{trigger}(X_1, Y) \wedge \text{trigger}(X_2, Y) \wedge \text{relacionamento\_sinônimo}(Y, R) \wedge \text{relacionamento\_não\_taxonômico}(R, C) \Rightarrow \text{é\_um}(I_1, C) \wedge \text{é\_um}(I_2, C) \wedge \text{relacionamento\_não\_taxonômico\_associação}(I_1, I_2, R, C) \quad \text{(I)}$$

$$\forall X, Y, I, P, C, V \mid \text{frase\_nominal}(X) \wedge \text{instância}(I, X) \wedge \text{trigger}(X, Y) \wedge \text{propriedade\_sinônimo}(Y, P) \wedge \text{propriedade}(P, C, V) \Rightarrow \text{é\_um}(I, C) \wedge \text{propriedade\_associação}(I, P, V, C) \quad \text{(II)}$$

Exemplo de regra de classificação gerada para o relacionamento não-taxonômico “married” da classe “Person” é mostrada na Figura 32.

noun_phrase (“John”) ^ noun_phrase (“H. R.”) ^
instance (“John_instance”, "John") ^ instance (“H. R._instance”, "H. R.") ^
trigger (“John”, “married with”) ^ trigger (“H. R.”, “married with”) ^
relationship_synonym (“married with”, “married”) ^
non_taxonomic_relationship ("married with", “Person”) ⇒
is_a ("John_instance","Person") ^ is_a ("H. R._instance","Person") ^
non_taxonomic_relationship_association (“John_instance”, “H. R._instance”, “married with”, “Person”)

Figura 32: Exemplo de regra de classificação gerada para o relacionamento não taxonômico “married”

Exemplo de regra de classificação gerada para a propriedade “birth_year” da classe “Person” é mostrada na Figura 33.

noun_phrase (“John”)
instance (“John_instance”, “John”)
trigger (“John”, “born in”)
property_synonym (“born in”, “birth_year”)
property (“birth_year”, “Person”, “1955”)
is_a (“John_instance”, “Person”)
property_association (“John_instance”, “birth_year”, “1955”, “Person”)

Figura 33: Exemplo de regra de classificação gerada para a propriedade “birth_year”

O produto da tarefa “Geração de Regras” para a classe “Person” da ontologia do direito de família e para o corpus ilustrado na Figura 26:

frase_nominal(“Keith R.”) ∧ frase_nominal(“H. R.”) ∧
instância(“Keith R._instância” “Keith R.”) ∧
instância(“H. R._instância”, “H. R.”) ∧
trigger(“Keith R._instância”, “married”) ∧
trigger(“H. R._instância”, “married”) ∧
relacionamento_sinônimo(“married”, “married”) ∧
relacionamento_não_taxonômico(“married”, “Person”)⇒
é_um(“Keith R._instância”, “Person”) ∧
é_um(“H. R._instância”, “Person”) ∧

relacionamento_não_taxonômico_associação(“Keith R._instância”, “H. R._instância”, “married”, “Person”)

frase_nominal(“Keith R.”) ∧ frase_nominal(“Mariana”) ∧ instância(“Keith R._instância”, “Keith R.”) ∧
instância(“Mariana_ instância”, “Mariana”) ∧
trigger(“Keith R.”, “father of”) ∧
trigger(“Mariana”, “father of”) ∧
relacionamento_sinônimo(“father of”, “father of”) ∧
relacionamento_não_taxonômico(“father of”, “Person”)⇒
é_um(“Keith R._instância”, “Person”) ∧
é_um(“Mariana_ instância”, “Person”) ∧
relacionamento_não_taxonômico_associação(“Keith R._instância”, “Mariana_instância”, “father of”, “Person”)

frase_nominal(“They”) ∧
instância(“They_instância”, “They”) ∧
trigger(“They”, “married in”) ∧
propriedade_sinônimo(“married in”, “constitutive”) ∧
propriedade(“constitutive”, “Person”, “2004”) ⇒
é_um(“They_instância”, “Person”) ∧
propriedade_associação(“They_instância”, “constitutive”, “2004”, “Person”)

frase_nominal(“Keith R.”) ∧
instância(“Keith R._instância”, “Keith R.”) ∧
trigger(“Keith R.”, “born in”) ∧
propriedade_sinônimo(“born in”, “birth_year”) ∧
propriedade(“birth_year”, “Person”, “1950”)⇒
é_um(“Keith R._instância”, “Person”) ∧
propriedade_associação(“Keith R._instância”, “birth_year”, “1950”, “Person”)

frase_nominal(“Keith R.”) ∧
instância(“Keith R._instância”, “Keith R.”) ∧

```
trigger(“Keith R.”, “divorce in”) ∧
propriedade_sinônimo(“divorce in”, “divorce”) ∧
propriedade(“divorce”, “Person”, “2006”)⇒
é_um(“Keith R._instância”, “Person”) ∧
propriedade_associação(“Keith R._instância”, “divorce”, “2006”, “Person”)
```

4.2.3. Classificação de Instâncias

A fase “Classificação de Instâncias” (Figura 34) consiste de duas tarefas: “Associação de Instâncias” e “Instanciação”. Esta fase tem como entradas o classificador, a ontologia e o corpus anotado e como produto a ontologia povoada.

A tarefa “Associação de Instâncias” associa as instâncias as suas respectivas classes, propriedades e relacionamentos não-taxonômicos da ontologia gerando o conjunto I’ $\subseteq$ I. Como explicado na seção 4.2.2. as frases nominais podem ser precedidas ou seguidas por um trigger. Sempre que um trigger casar com o predicado trigger da regra de classificação do tipo I (mostrada na seção 4.2.2.) as instâncias serão classificadas de acordo com o predicado de conclusão é_um(I,C). As instâncias de relacionamentos não-taxonômicos desta classe serão classificados de acordo com o predicado de conclusão relacionamento_não_taxonômico_associação(I_1, I_2, R, C). Da mesma forma que um trigger casar com o predicado trigger da regra de classificação do tipo II (mostrada na seção 4.2.2.) as instâncias serão classificadas de acordo com o predicado de conclusão é_um(I,C). As instâncias de propriedades desta classe serão classificadas de acordo com o predicado de conclusão propriedade_associação(I, P, V, C). Por exemplo, o produto da tarefa “Associação de Instâncias” para o fragmento de texto da Figura 26 e a classe “Person” da ontologia do direito de família é o conjunto I’ = {married(“Keith R.”, “H. R.”), father(“Keith R.”, “Mariana”), constitutive(“2004”, “Date”), birth_year(“1950”, “Date”) e divorce(“2006”, “Date”)}.

A tarefa “Instanciação” visa efetivamente povoar a ontologia com o conjunto I’ gerado pela tarefa “Associação de Instâncias”. As instâncias duplicadas

no conjunto I' são previamente removidas. Para cada instância do conjunto I' uma pesquisa da classe a ser instanciada é realizada. O produto da tarefa "Instanciação" é mostrado na Figura 35, a classe "Person" da ontologia do direito de família povoada.

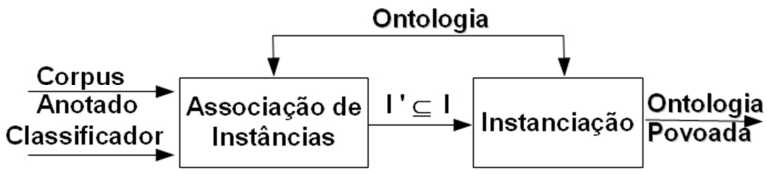


Figura 34: A fase "Classificação de Instâncias" do processo proposto.

Person		
married	Instance	Person
father	Instance	Person
birth_year		Date
divorce		Date
constitutive		Date

is_a

Keith R._Instance	
married	H. R._Instance
father	Mariana_Instance
birth_year	1950
divorce	2006
constitutive	2004

Figura 35: Classe "Person" povoada.

4.3. DIAOP-Tool – Uma Ferramenta para o Povoamento Automático de Ontologias

Considerando o processo proposto e a língua alvo escolhida foi desenvolvida a DIAOP-Tool - uma ferramenta para o Povoamento Automático de Ontologias. A DIAOP-Tool tem como entradas o corpus em língua inglesa e a ontologia OWL e como saída a ontologia OWL povoada.

A ferramenta foi implementada em linguagem Java e utiliza a biblioteca de classes do GATE, do Wordnet e do JENA. A DIAOP-Tool utiliza os seguintes recursos do GATE para a aplicação da fase "Identificação de Instâncias Candidatas": "Document Reset PR", "ANNIE English Tokeniser", "ANNIE Gazetter",

“Sentence Splitter”, “ANNIE POS Tagger”, “GATE Morphological Analyser”, “NE Transducer”, “ANNIE OrthoMatcher” e “Pronominal Coreferences”.

Para a aplicação da fase “Construção de um Classificador” a DIAOP-Tool utiliza a API do “JENA” para ler a ontologia OWL e a API do “Wordnet” para a recuperação dos sinônimos que serão utilizados na geração das regras de classificação que compõe o classificador. As regras de classificação são geradas na linguagem JAPE (Java Annotation Patterns Engine), que é uma linguagem poderosa para definição de regras utilizada pelo GATE (descrito no capítulo 2). JAPE consiste em regras lingüísticas do tipo <condição, ação>. Uma regra JAPE tem dois lados: o Esquerdo e o Direito. O lado esquerdo da gramática contém uma expressão regular a ser detectada no conjunto de documentos. O lado direito descreve a ação a ser tomada sobre a expressão regular detectada. A Figura 36 ilustra a regra gerada em JAPE para a propriedade “birth_in” da classe “Person” da ontologia do Direito de Família. O lado esquerdo contém instâncias da propriedade “birth_in”, enquanto o lado direito estabelece a classificação de “Data” (“CD”) como instância da propriedade “birth_in” da classe “Person” da ontologia do Direito de Família. O conjunto de regras JAPE geradas compõe o classificador.

```
Rule: Person_Birth0
Priority: 50
(
        {Token.string =~ "[Bb]irth" }
        {SpaceToken}
        {Token.string =~ "[io]n" }
        {SpaceToken}
        {Token.category == CD }
):Person_Birth
-->
:Person_Birth.Person_Birth = { rule = "Person_Birth0" ,InterText="True", RuleType="birth [io]n @CD",
owlPropName="Date_of_Birth", owlClassName="Person", owlRanger="http://www.w3.org/2001/XMLSchema#date" }
```

Figura 36: Exemplo de regra em JAPE gerada pela ferramenta para a propriedade “Birth_in” da classe “Person” da ontologia do direito de Família.

Para a aplicação da fase “Classificação de Instâncias” a DIAOP-Tool utiliza a API do “GATE” para aplicar o classificador induzido e a API do “JENA” para realizar a efetiva instanciação da ontologia OWL.

A Figura 37 mostra parte do diagrama de classes da ferramenta DIAOP-Tool, que têm as seguintes classes: “Intertext”, “ExtractInstance”, “TextEngineer”, “Plugin”, “PluginResource”, “OWLtoJAPE”, “JENA”, “Wordnet”, “JAPE”, “Rule”, “Category”, “MInstance” e “MInstanceProperty”. A classe “InterText” é a classe principal sendo responsável por dividir as tarefas e validar as informações capturadas. A classe “ExtractInstance” é responsável pela anotação do corpus. A classe “TextEnginner” é responsável por executar e configurar os plugins do GATE. Os plugins do GATE são configurados usando as classes “Plugin” e “PluginResource”. A classe “OWLtoJAPE” é responsável por gerar as regras de classificação na linguagem JAPE, entretanto esta tarefa é realizada juntamente com as classes “JAPE”, “Wordnet” e “JENA”, porém é de sua responsabilidade organizar as regras de classificação. A classe “Wordnet” é responsável pela consulta dos sinônimos das classes, propriedades e relacionamentos não taxonômicos. A classe “JENA” é responsável pela leitura do arquivo OWL e pela seleção das classes, propriedades e relacionamentos não taxonômicos. A classe “JAPE” é responsável pela geração das regras. Para auxiliar a criação das regras existem duas classes a “Rule” e a “Category”, elas são usadas para facilitar a criação de cada regra na linguagem Jape e especificar a categoria da regra respectivamente. A classe “ExtractInstance” também é responsável pela classificação das instâncias e armazena-as na classe “MInstances”. A classe “MInstanceProperty” é responsável por armazenar as instâncias de uma ontologia OWL já povoada.

A Figura 38 mostra o diagrama de seqüência da ferramenta DIAOP-Tool. Na tela inicial da ferramenta o usuário seleciona o corpus e seleciona a ontologia OWL para a realização do povoamento automático de ontologias. A classe “Intertext” é disparada com a mensagem “Povoar Ontologia” e ela dispara a classe “ExtractInstance” para que realize a anotação do corpus. A classe “TextEngineer”

realiza a anotação do corpus. A classe “Intertext” dispara a mensagem para a classe “OWLtoJAPE” para a geração do classificador. A classe “OWLtoJAPE” envia uma mensagem para a classe “JENA” para que ela realize a seleção das classes, propriedades e relacionamentos não taxonômicos da ontologia OWL. Em seguida a classe “OWLtoJAPE” dispara uma mensagem para a classe “Wordnet” solicitando a seleção dos sinônimos das classes, propriedades e relacionamentos não taxonômicos. Depois a classe “OWLtoJAPE” envia uma mensagem para a classe “JAPE”, para geração das regras de classificação. Após a geração das regras o classificador é induzido. Em seguida a classe “Intertext” dispara uma mensagem para a classificação das instâncias para a classe “ExtractInstance”. A classe “Jena” é disparada para ler o arquivo OWL e comparar as instâncias, para que não haja duplicidade na ontologia. Por último, a classe “JENA” realiza o povoamento da Ontologia enviando-a a classe “Intertext”.

A Figura 39 mostra a tela inicial da ferramenta, que recebe como entrada o corpus, que é selecionado o diretório onde se encontra os documentos a serem anotados. Também recebe como entrada a ontologia OWL, que é selecionado o arquivo da ontologia, para que seja feito o povoamento. Após a seleção do corpus e da ontologia a ferramenta executa as fases: “Identificação de Instâncias Candidatas”, “Construção de um Classificador” e “Classificação de Instâncias” tendo como produto a ontologia OWL povoada (Figura 40).

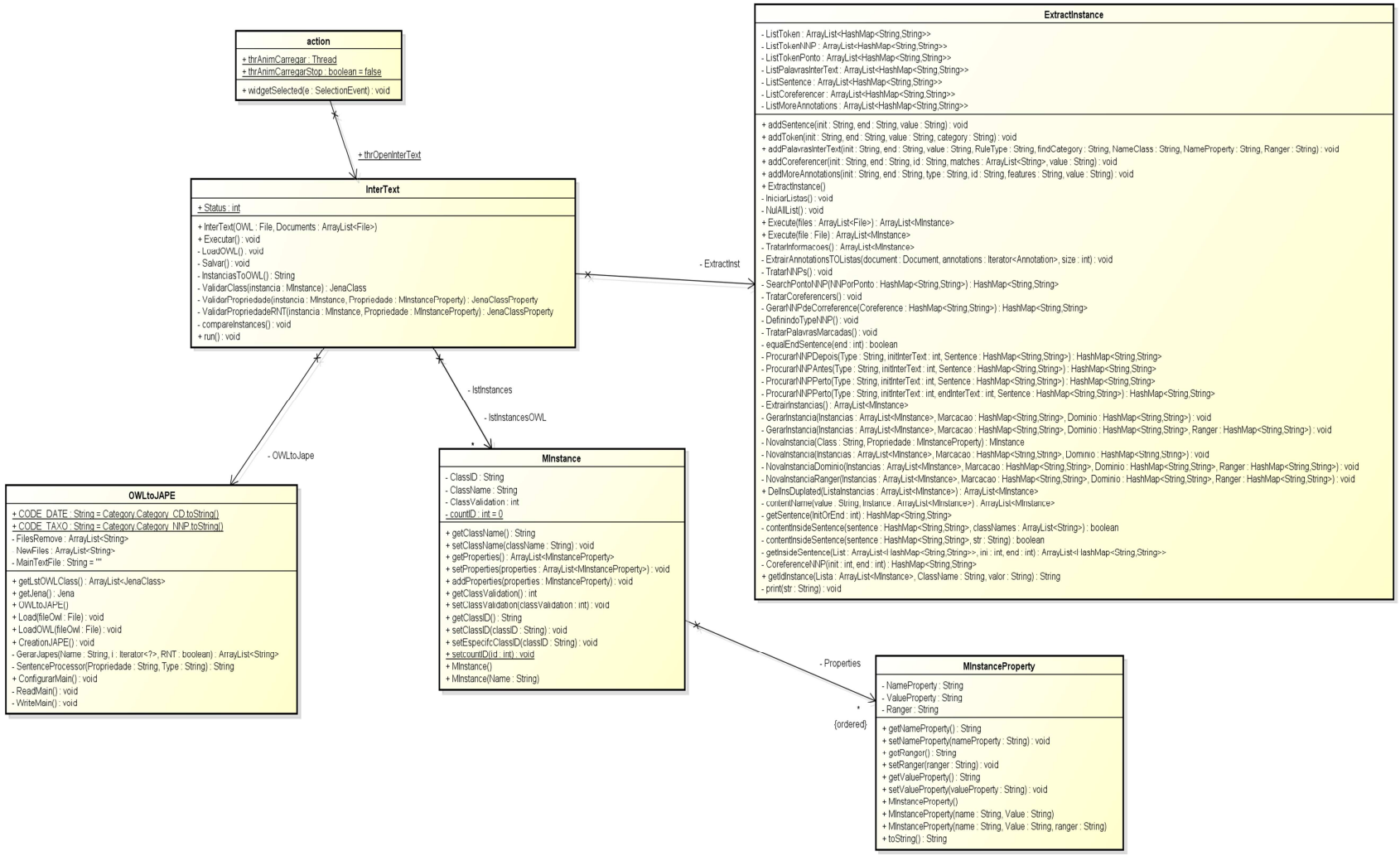


Figura 37: Parte do Diagrama de Classe da ferramenta DIAOP-Tool.

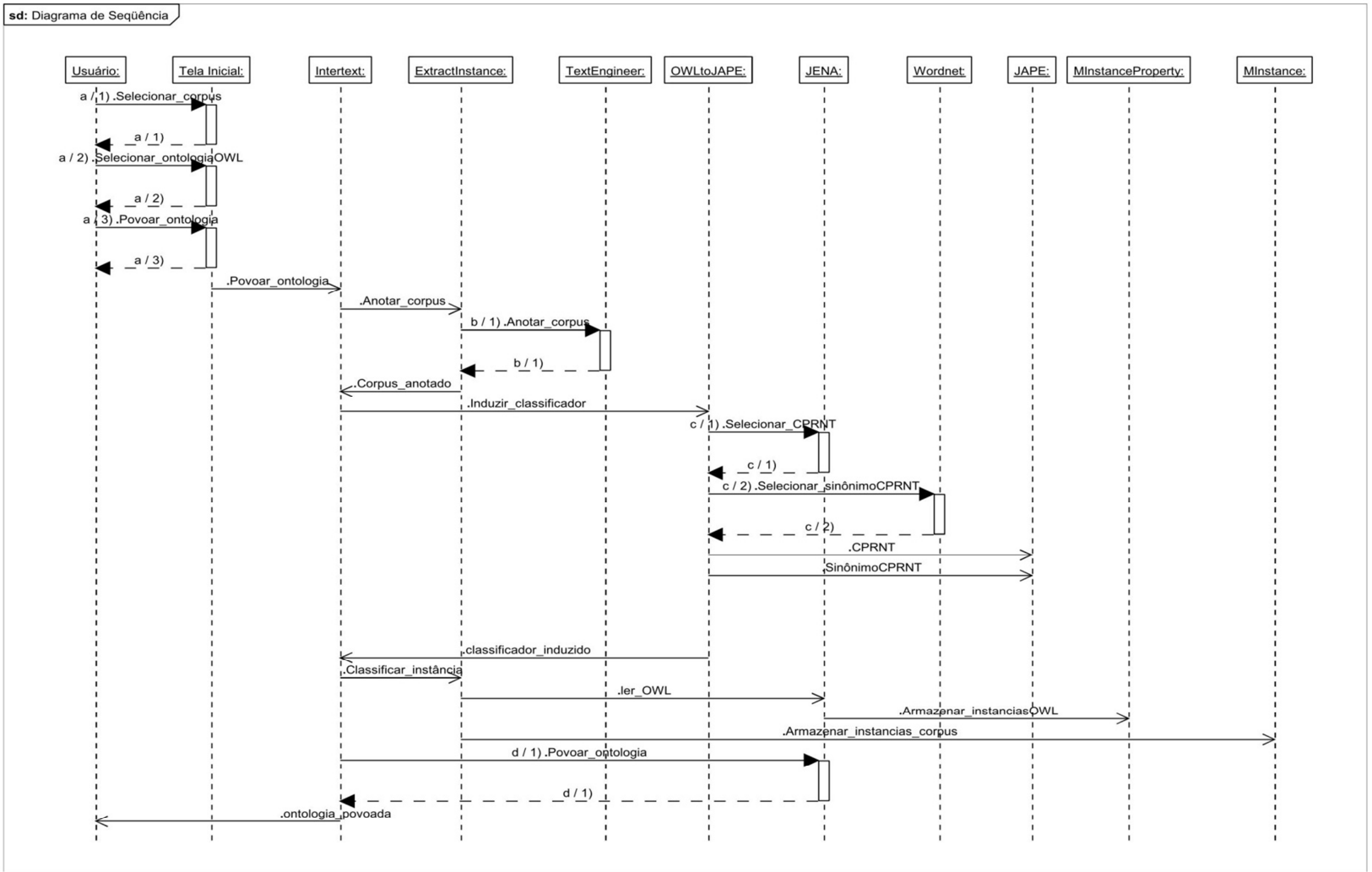


Figura 38: Diagrama de seqüência da ferramenta DIAOP-Tool.

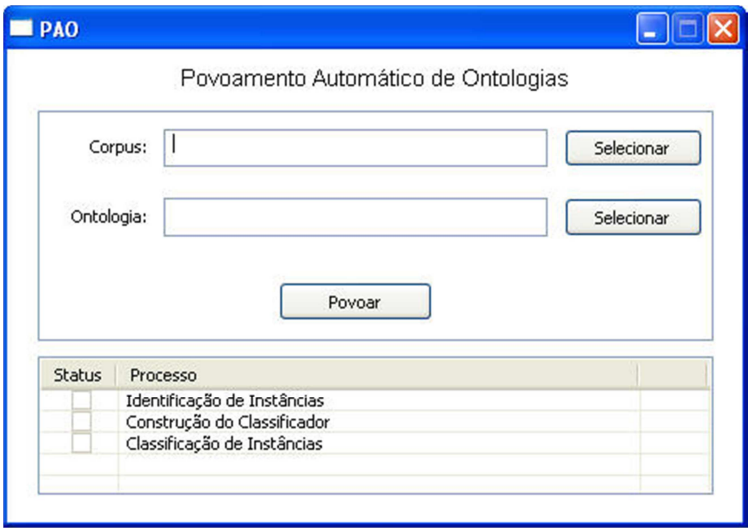


Figura 39: Tela inicial da ferramenta.

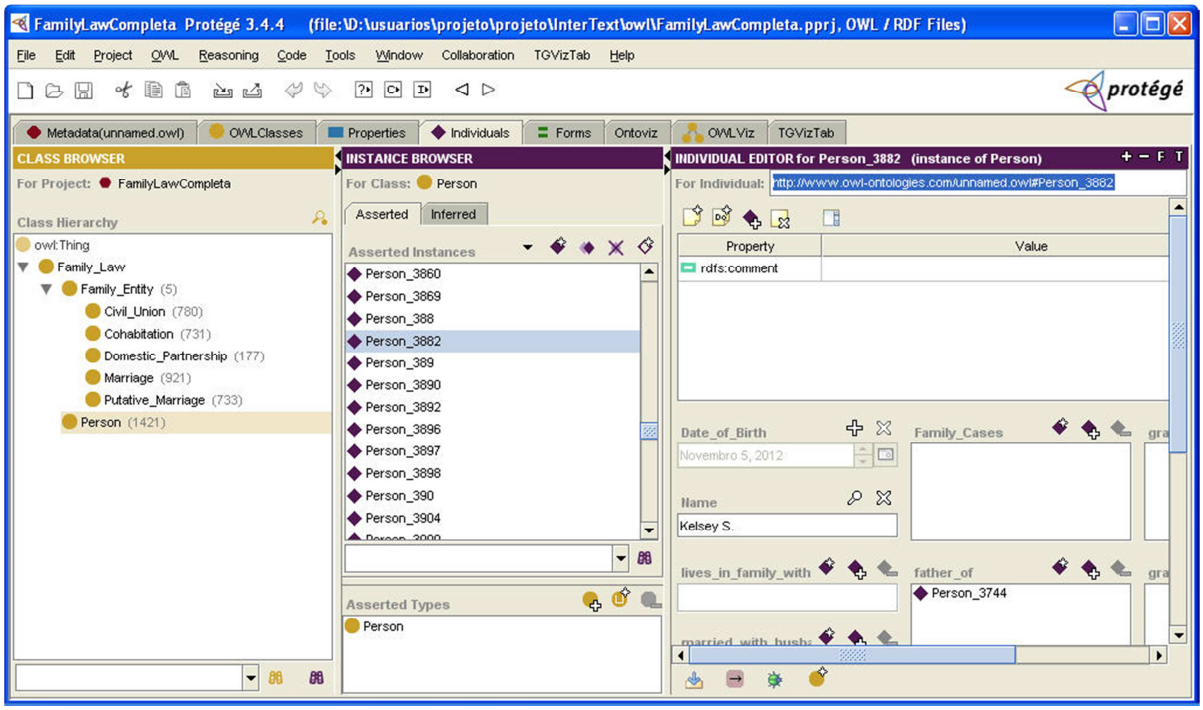


Figura 40: Parte da ontologia do direito de Família povoada

4.4. Considerações Finais

Neste capítulo foi apresentado DIAOP-Pro - Um Processo Independente de Domínio para o Povoamento Automático de Ontologias – principal contribuição desta pesquisa, que se constitui em uma nova abordagem uma vez que propõe o povoamento automático de ontologias utilizando uma ontologia para a geração automática de regras para extrair instâncias a partir de textos e classifica-as como instâncias de classes da ontologia. Estas regras podem ser geradas a partir de ontologias específicas de qualquer domínio, tornando o DIAOP-Pro independente de domínio.

O DIAOP-Pro não leva em consideração os axiomas representados pelo conjunto A da definição de ontologia apresentada na seção 2.3. O produto da aplicação do DIAOP-Pro é uma ontologia povoada, quando os axiomas dessa ontologia povoada forem submetidos a um processo de inferência novas instâncias serão descobertas e acrescentadas a ontologia. Por exemplo, considerando o axioma, “irmão (x,y) $\Rightarrow$ genitor (x,z) $\wedge$ genitor (y,z) ”, que indica que se duas pessoas tem o mesmo genitor, então as pessoas são irmãos e considerando que a ontologia povoada têm a instância de “genitor”: “João”; e instância de “pessoa”: “Maria” e “Josefina”. Sendo genitor (Maria, João) e genitor (Josefina, João), após o axioma ser submetido a um processo de inferência será descoberto que “Maria” e “Josefina” são irmãs, conseqüentemente a nova instância do relacionamento não taxonômico “irmão (Maria, Josefina)” será acrescentada a ontologia.

O DIAOP-Pro não foi concebido para ser aplicado somente na língua inglesa, entretanto a ferramenta que provê suporte automatizado a sua aplicação trabalha com textos em língua inglesa. No entanto, cabe salientar que a aplicação do DIAOP-Pro em outras línguas não inviabiliza nenhuma das contribuições científicas aqui apresentadas. Na verdade, conforme será visto nas conclusões desta tese, um trabalho futuro será avaliar o DIAOP-Pro sendo aplicado em textos de outra língua, por exemplo, língua portuguesa, sendo necessária a seleção ou criação de ferramentas que auxiliem a aplicação de técnicas de Processamento da

Linguagem Natural, Extração de Informação e uma base de dados léxica na língua alvo escolhida.

A Tabela 8 mostra um comparativo entre o processo DIAOP-Pro proposto e as abordagens do estado da arte. O DIAOP-Pro aplica o PAO através das fases: Identificação de Instâncias Candidatas, Construção de um Classificador e Classificação de Instâncias como as abordagens do estado da arte. A fase Identificação de Instâncias Candidatas é realizada através da aplicação de técnicas de PLN. As fases Construção de um Classificador e Classificação de Instâncias são realizadas através da aplicação de técnicas de EI.

Abordagens / Ano	Técnicas / Abordagens	Ferramentas	Efetividade	Fonte	Domínio	Automação
Fleischman e Hovy / 2002	PLN e AM	MenRun	70,4%	Corpus	Independente de Domínio	Automática
Evans / 2003	PLN, EI e AM	NERO	92,25%	Corpus	Independente de Domínio	Automática
Tanev e Magnini / 2006	PLN, EI e AM	MiniPar	65%	Corpus	-	Automática
Cimiano e Volker / 2005	PLN, EI e AM	Pankow	36,82%	Corpus	Turístico	Automática
Etzioni et. al. / 2005	PLN, EI e AM	KnowItAll	90%	Web	Independente de Domínio	Automática
Cimiano et. al. / 2005	PLN, EI e AM	C-Pankow	74,37%	Web	Independente de Domínio	Automática
Faria / 2013	PLN e EI	DIAOP-PROTool	87,3%	Corpus	Independente de Domínio	Automática
Karkaletsis et. al. / 2006	PLN, EI e AM	M-PIRO NLG	-	Corpus	Biomedicina	Semi-Automática
Macedo / 2010	PLN e Estatístico	NLPDumper	64,6%	Corpus	Independente de Domínio	Somente instâncias candidatas
Girardi / 1995	PLN e EI	ROSA	-	Corpus	Software	Automática
Ruiz-Martinez et. al. / 2008	PLN e EI	GATE	93,2%	Corpus	Turístico	Automática

Tabela 8: Quadro comparativo de abordagens para o Povoamento Automático de Ontologias e o processo DIAOP-PRO proposto

A ferramenta DIAOP-Tool foi desenvolvida na linguagem JAVA e utiliza o GATE para a aplicação das técnicas de PLN e EI. A DIAOP-Tool automatiza o processo DIAOP-Pro.

O DIAOP-Pro foi avaliado nos domínios jurídico e turístico e apresentou uma boa efetividade como mostra a Tabela 8.

O DIAOP-Pro utiliza um corpus como fonte de informação como algumas abordagens do estado da arte.

A principal vantagem do processo DIAOP-Pro é a independência de domínio no povoamento automático de ontologias específicas de domínio. Isso se deve ao fato das regras serem geradas a partir de uma ontologia durante a execução do processo proposto. Então, uma vez que a ontologia de entrada é definida, o processo automaticamente povoa a ontologia em um domínio específico com instâncias extraídas a partir de documentos em linguagem natural.

O próximo capítulo apresenta as avaliações realizadas no processo DIAOP-Pro com a aplicação da ferramenta DIAOP-Tool.

5. Avaliação

Este capítulo tem como objetivo apresentar as avaliações realizadas do processo DIAOP-Pro proposto, de modo a demonstrar a efetividade de suas fases, a viabilidade da geração automática do classificador e a independência de domínio com o povoamento automático de ontologias específicas dos domínios jurídico e turístico. Os experimentos foram realizados com a aplicação da ferramenta DIAOP-Tool descrita na seção 4.3 do capítulo 4. A Tabela 9 mostra os objetivos dos quatro estudos de caso desenvolvidos para a avaliação do processo DIAOP-Pro.

Estudo de Caso	Objetivo da Avaliação	Corpus	Ontologia
1	Efetividade da identificação de instâncias	FamilyJuris	FamilyLaw
2	Viabilidade da geração automática do classificador	FamilyJuris	FamilyLaw
		Turístico	OntoTur
3	Efetividade do povoamento automático da FamilyLaw	FamilyJuris	FamilyLaw
4	Efetividade do povoamento automático da OntoTur	Turístico	OntoTur

Tabela 9: Estudos de caso desenvolvidos para avaliação do processo DIAOP-Pro proposto

O primeiro estudo de caso descrito na seção 5.1 foi realizado para avaliar a efetividade da fase “Identificação de Instâncias Candidatas”, no qual foram comparados os resultados obtidos com a aplicação de técnicas estatísticas e de técnicas puramente lingüísticas no corpus FamilyJuris para a identificação de instâncias da ontologia FamilyLaw.

O segundo estudo de caso descrito na seção 5.2 foi realizado para avaliar a viabilidade da fase “Construção de um Classificador”, através da geração automática de classificadores e aplicando-os nos corpora FamilyJuris e Turístico e nas ontologias FamilyLaw e OntoTur para a classificação de instâncias.

O terceiro e o quarto estudos de caso descritos nas seções 5.3 e 5.4, respectivamente, foram realizados para avaliar a efetividade e a independência de

domínio do DIAOP-Pro proposto no povoamento automático de ontologias específicas em dois domínios distintos, jurídico e turístico, utilizando os corpora FamilyJuris e Turístico e as ontologias FamilyLaw e OntoTur respectivamente.

5.1. Estudo de Caso I: Aplicação de técnicas estatísticas e de técnicas puramente lingüísticas na fase "Identificação de Instâncias Candidatas"

O estudo de caso consistiu na utilização do corpus FamilyJuris como entrada da fase "Identificação de Instâncias Candidatas" com a aplicação de técnicas estatísticas e de técnicas puramente lingüísticas, assim como na avaliação da efetividade dos resultados obtidos nessa fase utilizando as métricas precisão, recall e medida-F, descritas na seção 2.7 do capítulo 2.

O corpus FamilyJuris [54] foi construído pelo grupo GESEC, extraído a partir do site FindLaw [36] e contêm 230 documentos que relatam decisões judiciais relativas a ações no Direito de Família. O Direito de família [23] é um ramo do Direito Civil, que constitui o complexo de normas que regulam entre outros fatos, a celebração do casamento, sua validade e os efeitos que dele resultam; as relações pessoais e econômicas da sociedade conjugal; a dissolução desta; as relações entre pais e filhos; os vínculos de parentesco e os institutos complementares da tutela, curatela e da ausência [11]. A Figura 41 mostra parte de um documento do corpus FamilyJuris.

As próximas subseções detalham o estudo de caso que está organizado como segue. A seção 5.1.1 aplica técnicas e estatísticas na fase "Identificação de Instâncias Candidatas" a partir do corpus FamilyJuris. A seção 5.1.2 aplica técnicas puramente lingüísticas na fase "Identificação de Instâncias Candidatas" a partir do corpus FamilyJuris. A seção 5.1.3 descreve uma análise comparativa da aplicação de técnicas estatísticas e de técnicas puramente lingüísticas na fase "Identificação de Instâncias Candidatas" e finalmente a seção 5.1.4 discute os resultados obtidos com os experimentos.

McKINSTER, J.- A father of minor children appeals from an order modifying his child-support obligation. We affirm.

FACTUAL AND PROCEDURAL BACKGROUND

Dennis Scheppers and Nancy Scheppers married in 1974. The marriage produced six children: Micah, Matthew, Amber, Joseph, Amanda, and Jennifer. When their marriage was dissolved in 1987, all six children were minors.

In July of 1998, the mother applied for and obtained an order to show cause seeking, inter alia, a modification of child support for the two minor children living with her. Following an evidentiary hearing, the trial court set the father's child support obligation at $2,991 per month. The father appeals.

CONTENTIONS

In a somewhat different order, the father contends that the trial court erred in four respects: by failing to include in the mother's gross income sums she received as the beneficiary of a life insurance policy; by failing to impute any income to the mother based upon her ability to earn; by basing the father's income upon an unreasonable work schedule; and by depriving the father of due process and equal protection of the law.

ANALYSIS

A. The Trial Court Did Not Err By Excluding The Life Insurance Proceeds From The Mother's Income. Micah, the eldest child, committed suicide in February of 1998, when he was 22 years old. In February or March of that year, the mother received $200,568 as the beneficiary of an insurance policy insuring Micah's life. The father argued that those life insurance proceeds should be counted as income to the mother in 1998. The trial court decided that the insurance proceeds were an asset, not income. However, the court did include as income the interest that could be earned from the investment of that corpus.

Figura 41: Exemplo de parte um documento do corpus FamilyJuris [54]

5.1.1. Aplicação de técnicas estatísticas na fase "Identificação de Instâncias Candidatas"

Este experimento foi realizado com a aplicação da ferramenta NLPDumper [54], que implementa a identificação de instâncias candidatas com a aplicação de técnicas estatísticas a partir do corpus FamilyJuris.

A fase "Identificação de Instâncias Candidatas" (Figura 42) consiste de nove tarefas: "Tokenização", "Divisão de Sentenças", "POS Tagging", "Lematização", "Reconhecimento de Entidades Nomeadas", "Co-Referência Nominal e Pronominal", "Parsing", "Identificação de Termos Candidatos" e "Extração de Instâncias Candidatas".

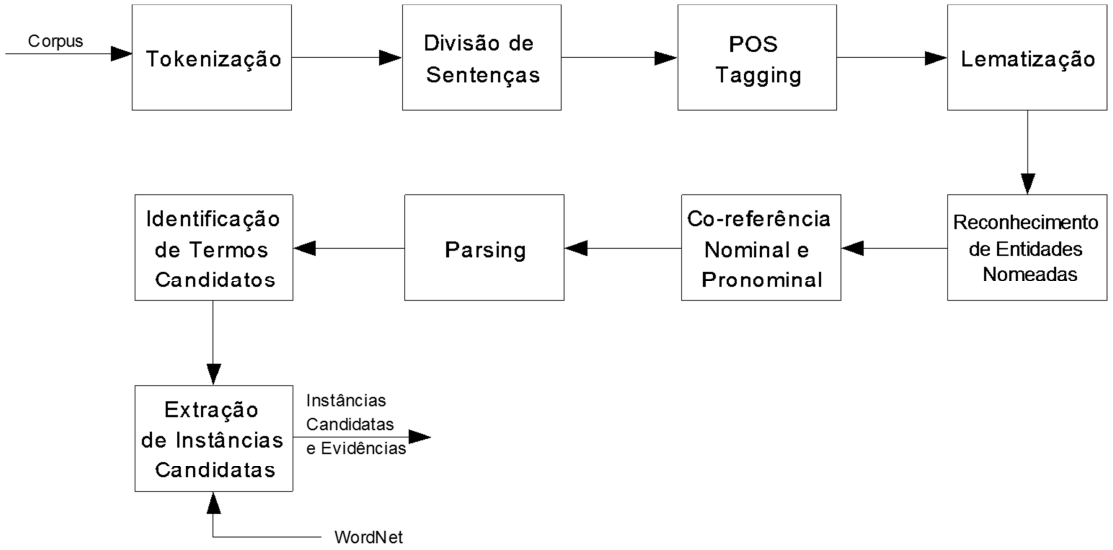


Figura 42: A fase "Identificação de Instâncias Candidatas" com a aplicação de técnicas estatísticas

As tarefas "Tokenização", "Divisão de Sentenças", "POS Tagging", "Reconhecimento de Entidades Nomeadas" e "Co-Referência Nominal e Pronominal" foram descritas no Capítulo 4. O "Parsing" é responsável por extrair a árvore sintática de cada sentença no corpus, identificando assim as dependências

sintáticas entre as palavras. A “Identificação de Termos Candidatos” é responsável por identificar os substantivos próprios e descartar todos os demais termos. A “Extração de Instâncias Candidatas” é responsável por ponderar os termos selecionados na tarefa anterior usando a medida TF-IDF, TF e/ou IDF, sendo selecionados apenas os termos cujo peso está dentro um determinado intervalo. As freqüências dos termos são computadas baseadas nos lemas dos tokens. Além disso, os termos que não têm o valor da métrica adotada dentro do intervalo pré-determinado, mas são instâncias do WordNet (instâncias de hiperônimos) são também selecionados para a formação da lista de instâncias candidatas.

Para se avaliar qual a melhor métrica estatística a ser adotada foram realizados experimentos com um corpus de 200 e 900 documentos. O desejável é que a medida a ser usada apresente intervalos com taxas de precisão e recall elevadas e ao mesmo tempo equilibradas. A partir dos experimentos concluiu-se que a medida IDF com intervalo de 0,56 a 0,93 é a melhor, dentre as outras analisadas.

A fase “Identificação de Instâncias Candidatas” é ilustrada através de exemplos utilizando o corpus FamilyJuris. Para a aplicação da tarefa “Tokenização” considera-se como entrada o texto de um documento do corpus FamilyJuris (Figura 41). O resultado da tarefa “Tokenização” e “POS Tagging” para o texto da Figura 41 é mostrado na Figura 43.

O resultado parcial da tarefa de “Lematização” para o texto da Figura 41 é mostrado na Tabela 10.

O resultado da tarefa “Reconhecimento de Entidades Nomeadas” para o texto da Figura 41 são conjuntos de oito entidades do tipo “Pessoa” e uma entidade do tipo “Localização”. As entidades do tipo “Pessoa” são: “McKINSTER, J.”, “Dennis Scheppers”, “Nancy Scheppers”, “Matthew”, “Amber”, “Joseph”, “Amanda” e “Jennifer”. A entidade do tipo “Localização” é “Nancy”.

Tokens	*Lema*
children	Child
dissolved	dissolve
was	Be
appeals	appeal
modifying	Modify

Tabela 10: Resultado parcial da tarefa “Lematização” para o texto da Figura 41

O resultado da tarefa “Co-Referência Nominal e Pronominal” para o texto da Figura 41 são as co-referências pronominais listadas na Tabela 11.

Tokens	*Co-Referências Pronominais*
McKINSTER, J.	His
Jennifer	her, she
father	He

Tabela 11: Resultado da tarefa “Co-Referência Nominal e Pronominal” para o texto da Figura 41

O resultado parcial da tarefa “Parsing” para o texto da Figura 41 são as dependências de Stanford (Anexo B) listadas na Tabela 12.

Token	*Token*	*Dependências*
McKINSTER	J	appos
McKINSTER J.	A	dep
A	father	det
father	of	prep
of minor	children	probj
minor	children	amod
children	appeals	nn

Tabela 12: Resultado parcial da tarefa “Parsing” para o texto da Figura 41

McKINSTER [NNP], J [NNP].- A [DT] father [NN] of [IN] minor [JJ] children [NNS] appeals [NNS] from [IN] an [DT] order [NN] modifying [VBG] his [PRP$] child-support [JJ] obligation [NN]. We [PRP] affirm [NN]. FACTUAL [NNP] AND [CC] PROCEDURAL [NNP] BACKGROUND [NNP] Dennis [NNP] Scheppers [NNP] and [CC] Nancy [NNP] Scheppers [NNP] married [VBD] in [IN] 1974 [CD]. The [DT] marriage [NN] produced [VBD] six [CD] children [NNS]: Micah [NNP], Matthew [NNP], Amber [NNP], Joseph [NNP], Amanda [NNP], and [CC] Jennifer [NNP]. When [WRB] their [PRP$] marriage [NN] was [VBD] dissolved [VBN] in [IN] 1987 [CD], all [DT] six [CD] children [NNS] were [VBD] minors [NNS]. In [IN] July [NNP] of [IN] 1998 [CD], the [DT] mother [NN] applied [VBN] for [IN] and [CC] obtained [VBD] an [DT] order [NN] to [TO] show [VB] cause [NN] seeking [VBG], inter [NN] alia [NN], a [DT] modification [NN] of [IN] child [NN] support [NN] for [IN] the [DT] two [CD] minor [JJ] children [NNS] living [VBG] with [IN] her [PRP]. Following [VBG] an [DT] evidentiary [JJ] hearing [NN], the [DT] trial [NN] court [NN] set [VBD] the [DT] father [NN]'s [POS] child [NN] support [NN] obligation [NN] at [IN] $2 [CD],991 [CD] per [IN] month [NN]. The [DT] father [NN] appeals [NNS]. CONTENTIONS [NNP] In [IN] a [DT] somewhat [RB] different [JJ] order [NN], the [DT] father [NN] contends [VBZ] that [IN] the [DT] trial [NN] court [NN] erred [VBN] in [IN] four [CD] respects [VBZ]: by [IN] failing [VBG] to [TO] include [VB] in [IN] the [DT] mother [NN]'s [POS] gross [JJ] income [NN] sums [NNS] she [PRP] received [VBD] as [IN] the [DT] beneficiary [NN] of [IN] a [DT] life [NN] insurance [IN] policy [NNP]; by [IN] failing [VBG] to [TO] impute [NN] any [DT] income [NN] to [TO] the [DT] mother [NN] based [VBD] upon [NNP] her [NNP] ability [NNP] to [TO] earn [NNP]; by [IN] basing [VBG] the [DT] father[NN]'s [POS] income [NN] upon [IN] an [DT] unreasonable [JJ] work [NN] schedule [NN]; and [CC] by [IN] depriving [VBG] the [DT] father [NN] of [IN] due [JJ] process [NN] and [CC] equal [JJ] protection [NN] of [IN] the [DT] law [NN]. ANALYSIS [NNP] A. [DT] The [DT] Trial [NNP] Court [NNP] Did [VBD] Not [RB] Err [VB] By [IN] Excluding [VBG] The [DT] Life [NNP] Insurance [NNP] Proceeds [NNS] From [IN] The [DT] Mother [NNP]'s [POS] Income [NNP]. Micah [NNP], the [DT] eldest [JJS] child [NN], committed [VBN] suicide [NN] in [IN] February [NNP] of [IN] 1998 [CD], when [WRB] he [PRP] was [VBD] 22 [CD] years [NNS] old [JJ]. In [IN] February [NNP] or [CC] March [NNP] of [IN] that [DT] year [NN], the [DT] mother [NN] received [VBD] $200,568 [CD] as [IN] the [DT] [NN], committed [VBN] suicide [NN] in [IN] February [NNP] of [IN] 1998 [CD], when [WRB] he [PRP] was [VBD] 22 [CD] years [NNS] old [JJ]. In [IN] February [NNP] or [CC] March [NNP] of [IN] that [DT] year [NN], the [DT] mother [NN] received [VBD] $200,568 [CD] as [IN] the [DT] beneficiary [NN] of [IN] an [DT] insurance [NN] policy [NN] insuring [VBG] Micah [NNP]'s [POS]

Figura 43: Parte do resultado das tarefas “Tokenização” e “POS Tagging” para o texto da Figura 41

O resultado da tarefa “Identificação de Termos Candidatos” para o texto da Figura 41 são as instâncias candidatas listadas na Tabela 13.

Termos Candidatos			
Income	Jennifer	Life	Matthew
Procedural	Joseph	Mother	Contentions
July	March	Insurance	Trial
Nancy	Dennis	Court	Factual
Amber	Analysis	Amanda	Mckinster
Background			

Tabela 13: Resultado da fase “Identificação de Termos Candidatos”

O resultado da tarefa “Extração de Instâncias Candidatas” para o texto da Figura 41 são as instâncias candidatas listadas na Tabela 14.

Instâncias Candidatas			
Income	Jennifer	Life	Matthew
Procedural	Joseph	Mother	Contentions
July	March	Insurance	Trial
Nancy	Dennis	Court	Factual
Amber	Analysis	Amanda	Mckinster
Background			

Tabela 14: Resultado da fase “Extração de Instâncias Candidatas”

5.1.2. Aplicação de técnicas puramente lingüísticas na fase “Identificação de Instâncias Candidatas”

Este experimento foi realizado com a aplicação da ferramenta DIAOP-Tool (descrita na seção 4.3 do capítulo 4), que implementa a aplicação de técnicas lingüísticas na fase “Identificação de Instâncias Candidatas” a partir do corpus FamilyJuris.

A fase “Identificação de Instâncias Candidatas” (Figura 44) consiste de três tarefas: “Análise Morfo-Lexical”, “Reconhecimento de Entidades Nomeadas” e “Identificação de Co-Referências”, como descritas no capítulo 4.

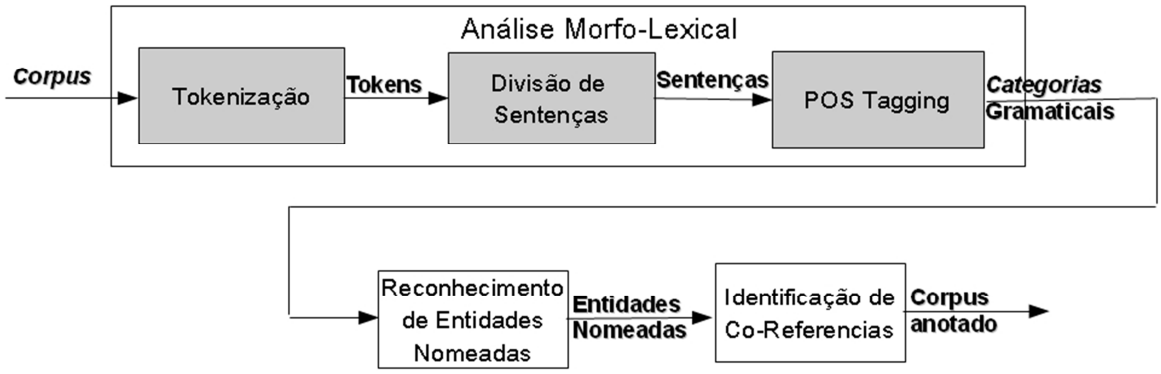


Figura 44: A fase “Identificação de Instâncias Candidatas”

A fase “Identificação de Instâncias Candidatas” é ilustrada através de exemplos utilizando o corpus FamilyJuris. Para a aplicação da tarefa “Análise Morfo-Lexical” considera-se como entrada o texto de um documento do corpus FamilyJuris (Figura 41). O resultado da tarefa “Análise Morfo-Lexical” para o texto da Figura 41 é mostrado na Figura 43.

O resultado da tarefa “Reconhecimento de Entidades Nomeadas” para o texto da Figura 41 são conjuntos de oito entidades do tipo “Pessoa” e uma entidade do tipo “Localização”. As entidades do tipo “Pessoa” são: “McKINSTER, J.”, “Dennis Scheppers”, “Nancy Scheppers”, “Matthew”, “Amber”, “Joseph”, “Amanda” e “Jennifer”. A entidade do tipo “Localização” é “Nancy”.

O resultado da tarefa “Identificação de Co-Referências” para o texto da Figura 41 são as co-referências pronominais listadas na Tabela 11.

As instâncias identificadas para o texto da Figura 41 estão listadas na Tabela 15.

Instâncias Candidatas			
Dennis Scheppers	Nancy Scheppers	Amanda	Micah
Joseph	Jennifer	Matthew	Amber
1974		1987	

Tabela 15: Resultado da fase “Identificação de Instâncias Candidatas”

5.1.3. Análise comparativa da aplicação de técnicas estatísticas e de técnicas puramente lingüísticas na fase “Identificação de Instâncias Candidatas”

Esta seção apresenta uma análise comparativa da efetividade da fase “Identificação de Instâncias Candidatas” com a aplicação de técnicas estatísticas e de técnicas puramente lingüísticas a partir do corpus FamilyJuris utilizando as métricas precisão, *recall* e medida-F.

O corpus FamilyJuris [54] foi utilizado como entrada para a fase “Identificação de Instâncias Candidatas” nos experimentos realizados. Cada documento foi analisado manualmente para saber quais termos seriam realmente instâncias, sendo identificados um total de 794 instâncias em 230 documentos e um total de 2038 instâncias em 900 documentos.

A Tabela 16 resume os resultados da avaliação da fase “Identificação de Instâncias Candidatas”.

Um primeiro experimento foi realizado aplicando as técnicas estatísticas na fase “Identificação de Instâncias Candidatas” com 230 documentos. Os termos identificados como instâncias foram 2324, as instâncias corretamente identificadas foram 332 das 794 instâncias no corpus. Obteve-se, portanto, uma precisão de 14%, um *recall* de 41% e uma medida-F de 20,87%.

Um segundo experimento foi realizado aplicando as técnicas puramente lingüísticas na fase “Identificação de Instâncias Candidatas” com 230 documentos. Os termos identificados como instâncias foram 783, as instâncias corretamente

identificadas foram 711 das 794 instâncias no corpus. Obteve-se, portanto, uma precisão de 90%, um *recall* de 89,5% e uma medida-F de 89,74%.

Um terceiro experimento foi realizado aplicando as técnicas estatísticas na fase “Identificação de Instâncias Candidatas” com 900 documentos. Os termos identificados como instâncias foram 3187, as instâncias corretamente identificadas foram 1609 das 2038 instâncias no corpus. Obteve-se, portanto, uma precisão de 50%, um *recall* de 78% e uma medida-F de 61%.

Um quarto experimento foi realizado aplicando as técnicas puramente lingüísticas na fase “Identificação de Instâncias Candidatas” com 900 documentos. Os termos identificados como instâncias foram 2019, as instâncias corretamente identificadas foram 1853 das 2038 instâncias no corpus. Obteve-se, portanto, uma precisão de 91%, um *recall* de 90% e uma medida-F de 90,49%.

Avaliação da identificação de instâncias	**Estatística**		**Lingüística**	
	Número de documentos no corpus FamilyJuris			
	230	**900**	**230**	**900**
Instâncias no corpus	794	2038	794	2038
Termos identificados como instâncias	2324	3187	783	2019
Instâncias corretamente identificadas	332	1609	711	1853
Precisão	14%	50%	90%	91%
Recall	41%	78%	89,50%	90%
Medida-F	20,87%	61%	89,74%	90,49%

Tabela 16: Resultado dos experimentos da fase “Identificação de Instâncias Candidatas”

Uma desvantagem da aplicação de técnicas estatísticas é que é necessário um corpus com um grande volume de documentos, para que os valores das medidas de precisão, recall e medida-F sejam bons, enquanto que as técnicas puramente lingüísticas não possuem essa restrição, podendo ser aplicadas em

corpus de qualquer tamanho. Como podemos observar quando foi utilizado um corpus com um maior número de documentos obteve-se uma significativa melhora nos valores de precisão, *recall* e medida-F, como mostra a Tabela 16, entretanto esses valores ainda permaneceram abaixo dos apresentados pelas técnicas puramente lingüísticas.

5.1.4. Discussão dos resultados obtidos com a aplicação de técnicas estatísticas e de técnicas puramente lingüísticas na fase "Identificação de Instâncias Candidatas"

Considerando os experimentos realizados (Tabela 16) podemos concluir que as técnicas puramente lingüísticas apresentam melhores resultados que as técnicas estatísticas. Isso se deve ao fato de que as técnicas estatísticas requerem um corpus com um grande volume de documentos, para que os valores de precisão e recall sejam bons. Como podemos observar no primeiro experimento na aplicação de técnicas estatísticas, quando foi utilizado um corpus com 230 documentos obteve-se uma precisão de 14% e um recall de 41%. Já no terceiro experimento na aplicação de técnicas estatísticas, quando foi utilizado um corpus com 900 documentos obteve-se uma precisão de 50% e um recall de 78%. Então, podemos concluir que na aplicação de técnicas estatísticas quanto maior o número de documentos no corpus irá ocorrer um aumento significativo nos valores de precisão e recall. Como podemos observar na aplicação de técnicas puramente lingüísticas o tamanho do corpus não influencia os valores de precisão e recall. No segundo e no quarto experimento na aplicação de técnicas puramente lingüísticas os valores de precisão e recall apresentaram uma variação mínima.

As técnicas estatísticas são adequadas para a identificação de instâncias, quando utilizamos um corpus com um grande volume de documentos, enquanto que as técnicas puramente lingüísticas são adequadas para a identificação de instâncias em um corpus com qualquer número de documentos.

Os termos identificados como instâncias pelas técnicas estatísticas apresentam muito ruído, como exemplo, termos que são claramente classes são identificados como instâncias. O contrário ocorre na aplicação das técnicas puramente lingüísticas aonde a maioria dos termos identificados como instâncias são realmente instâncias.

Considerando os resultados obtidos no primeiro estudo de caso, as técnicas puramente lingüísticas foram escolhidas para serem aplicadas na fase "Identificação de Instâncias Candidatas" do DIAOP-Pro proposto.

5.2. Estudo de Caso II: Construção de um classificador de forma manual e automática na fase "Construção de um Classificador"

O estudo de caso consistiu na utilização dos corpora FamilyJuris e Turístico e na utilização das ontologias FamilyLaw e OntoTur como entradas da fase "Construção de um Classificador" de forma manual e automática, assim como na avaliação da efetividade dos resultados obtidos nessa fase utilizando as métricas precisão, recall e medida-F, descritas na seção 2.7 do capítulo 2. A viabilidade foi avaliada através da geração automática do classificador, pois a geração manual do classificador é uma tarefa custosa e dispendiosa, no entanto é efetiva por causa da intervenção humana.

O corpus FamilyJuris foi descrito na seção 5.1. O corpus Turístico foi construído a partir do site LonelyPlanet[5] e contêm 34 documentos que descrevem pontos turísticos de diversos países. A Figura 45 mostra parte de um documento do corpus Turístico.

As ontologias FamilyLaw e OntoTur serão descritas nas seções 5.2.1 e 5.2.2 respectivamente.

A geração automática dos classificadores (descrita na seção 5.2.4) (Figura 46) foi realizada pela ferramenta DIAOP-Tool (descrita na seção 4.3 do

[5] http://www.loonelyplanet.com/

capítulo 4) a partir do corpus FamilyJuris e da ontologia FamilyLaw e a partir do corpus Turístico e da ontologia OntoTur.

Hotel De Tassche is beautiful step-fronted traditional house. This Hotel has been an Inn since at least 1682 (though the facade is inscribed 1710). It's family-run, beautifully maintained and superbly positioned just a block off Bruges's central square.Behind the historic facade, room decor is modern and gently stylish. Some bed covers are a fascinatingly irridescent blue-black colour. Above on the grey bed-boards in calligraphic script is the phrase Zoete Zachte Dromen (Sweet soft dreams). The bathrooms are gleaming white cubes with decent showers. Even if you're not staying on the top floor it's worth taking the lift to have a look at the roof structure. Like in many Bruges houses the original beams remain (these date to 1450), and are held in place by wooden pegs. The beam-structure was prefabricated on the ground, then taken apart like a jigsaw puzzle to reassemble in situ. You can still see carpenters' marks to show where each piece went. Rather than divide the high ceiling, the hotel has cunningly created boxes for individual upper-floor rooms. From the wiggling corridor between them the whole apex of the roof beams is visible.

Figura 45: Exemplo de parte de um documento do corpus do domínio turístico

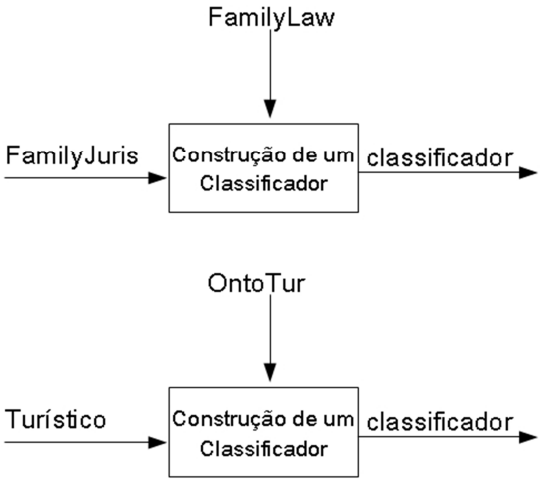


Figura 46: Geração automática do classificador

A construção manual do classificador (descrita na seção 5.2.3) (Figura 47) foi realizada por um especialista de domínio após uma análise do corpus FamilyJuris e da ontologia FamilyLaw e do corpus Turístico e da ontologia OntoTur. Foram gerados dois classificadores de forma manual.

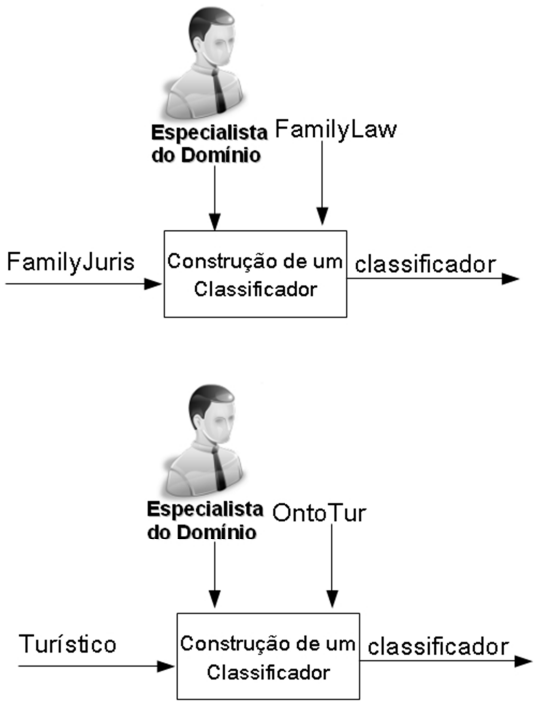


Figura 47: Construção manual do classificador

As próximas seções detalham o estudo de caso que está organizado como segue. A seção 5.2.1 apresenta a ontologia FamilyLaw. A seção 5.2.2 descreve a ontologia OntoTur. A seção 5.2.3 apresenta a construção manual do classificador. A seção 5.2.4 descreve a geração automática do classificador. A seção 5.2.5 apresenta uma análise da aplicação dos classificadores construídos de forma manual e gerados de forma automática nos corpora FamilyJuris e Turístico e nas ontologias FamilyLaw e OntoTur e finalmente a seção 5.2.6 discute os resultados obtidos com os experimentos.

5.2.1. Ontologia FamilyLaw

A ontologia FamilyLaw foi desenvolvida pelo grupo GESEC com o uso da ferramenta Protégé. A FamilyLaw descreve conhecimento acerca do ramo do Direito de Família brasileiro [23] e consiste de uma classe raiz “Family_Law”, da qual derivam duas grandes classes “Family_Entity” e “Person”. A primeira descreve as principais entidades familiares legalmente consideradas e a segunda discrimina os elementos pessoais constituintes de uma família. A estrutura de classes da ontologia é mostrada na Figura 48.

Uma entidade familiar, descrita pela classe “Family_Entity” , consiste de um vínculo jurídico entre duas ou mais pessoas, caracterizado obrigatoriamente por uma propriedade “married_date”, que indica a data de formação do vínculo e, opcionalmente, por uma propriedade “dissolution_date”, que indica a data de finalização do vínculo.

Há diferentes tipos de entidades familiares, conforme apresenta a hierarquia de classes da ontologia (Figura 48): “Marriage”, “Cohabitation”, “Civil_Union”, “Domestic Patnership” e “Putative_Marriage”, subclasses da superclasse “Family_Entity”. Todas as subclasses herdam as propriedades “constitutive_date” e “dissolution_date” da superclasse “Family_Entity”.

Um casamento, representado pela classe “Marriage”, é uma entidade familiar formada por esposo, esposa e, em alguns casos, um ou mais filhos. Convém ressaltar que neste trabalho é considerado casamento a união entre duas pessoas de sexos diferentes. Os atributos específicos da classe “Marriage” são os relacionamentos não taxonômicos: “child_members” (indica os filhos), “husband_member” (indica o esposo) e “wife_member” (indica a esposa).

A união estável representada pela classe “Cohabitation” é um acordo firmado entre duas pessoas, no qual ambas decidem viver juntas como se casados fossem. Os atributos específicos da classe “Cohabitation” são os relacionamentos não taxonômicos: “child_members” (indica os filhos), “femal_partner” (indica o parceiro feminino) e “male_partner” (indica o parceiro masculino).

A união civil representada pela classe “Civil_Union” é todo contrato civil entre duas pessoas do mesmo sexo com fins de constituir família. Os atributos específicos da classe “Civil_Union” são os relacionamentos não taxonômicos: “child_members” (indica os filhos) e “same_sex_couple” (indica as duas pessoas do mesmo sexo).

A parceria doméstica representada pela classe “Domestic_Partnership” é uma relação jurídica entre dois indivíduos que vivem juntos, mas não são unidos por casamento ou união civil, como por exemplo, um avô com seus netos. Os atributos específicos da classe “Domestic_Partnership” são os relacionamentos não taxonômicos: “chief_member” (indica membro chefe) e “dependents_members” (indica os membros dependentes).

O casamento putativo representado pela classe “Putative_Marriage” é um casamento aparentemente válido, mas legalmente inválido por um impedimento técnico, como, por exemplo, um casamento pré-existente desconhecido por uma das partes. Os atributos específicos da classe “Putative_Marriage” são os os relacionamentos não taxonômicos: “child_members” (indica os filhos), “femal_partner” (indica o parceiro feminino), “male_partner” (indica o parceiro masculino) e “technical_impediment” (indica o impedimento técnico).

Uma pessoa, descrita pela classe “Person”, é caracterizada pelas propriedades “name” e “date_of_birth”. Os relacionamentos não taxonômicos desta classe são os membros de uma família: “father_of”, “mother_of”, “brother_of”, “sister_of”, “daugter_of”, “son_of”, “husband_of”, “wife_of”, “grandparent_of”, “grandmother_of”, “grandson_of”, “granddaughter_of”, “lives_in_family_with_female _family_partner” e “lives_in_family_with_male_family_partner”.

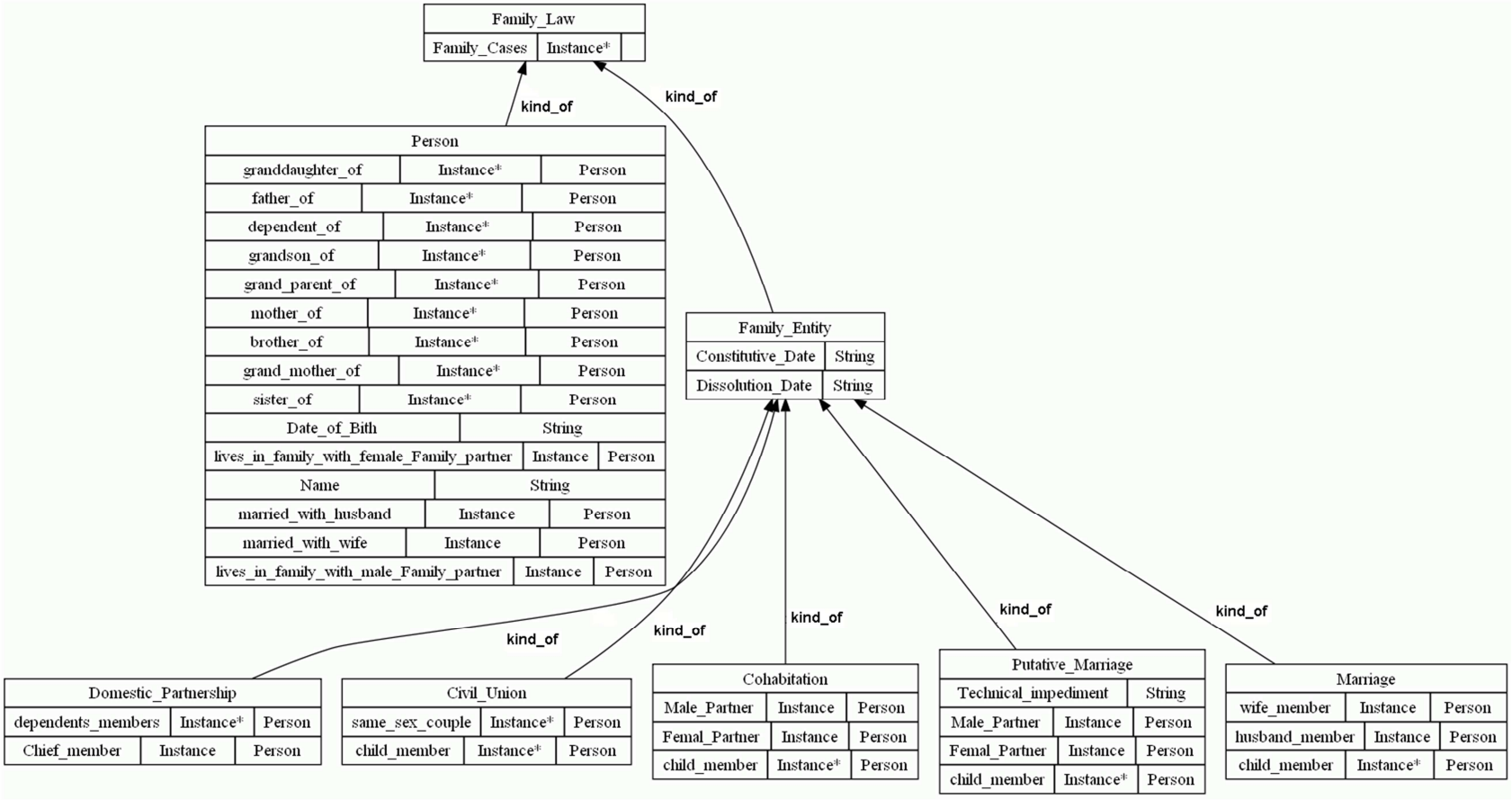


Figura 48: Classes, relacionamentos não taxonômicos e propriedades da ontologia FamilyLaw.

5.2.2. Ontologia OntoTur

A ontologia OntoTur foi desenvolvida pelo grupo GESEC com o uso da ferramenta Protégé. A OntoTur descreve conhecimento acerca do domínio Turístico e consiste de duas grandes classes “Tourism” e “Services”. Da classe “Tourism” (Figuras 49 e 50) derivam nove grandes classes “Business”, “Congressional”, “Cultural”, “Ecological”, “Ecotourism”, “Exchange”, “Religious”, “Scientific” e “Vacation”. Estas classes descrevem os principais tipos de turismo. Da classe “Services” (Figuras 51 e 52) derivam quatro grandes classes “Transport”, “Food”, “Accomodation” e “Entertainment”. Estas classes descrevem os principais serviços oferecidos aos turistas. Turista é todo indivíduo que permanece por mais de 24 horas em uma localidade visitada e não realiza nenhuma atividade remunerada.

Há diferentes tipos de turismo, conforme apresenta a hierarquia de classes da ontologia (Figuras 49 e 50). O Turismo de Negócios, representado pela classe “Business”, é caracterizado pelo deslocamento de executivos, que afluem aos grandes centros empresariais e cosmopolitas a fim de efetuar transações e atividades profissionais, comerciais e industriais. Da classe “Business” derivam três subclasses “Comercial”, “Industrial” e “Business Center”.

O Turismo de Congresso, representado pela classe “Congressional”, é caracterizado pelo deslocamento de turistas que se destinam a núcleos receptores eleitos para a realização de congressos e seminários de distintos assuntos e especialidades. Da classe “Congressional” derivam duas subclasses “Congress” e “Seminar”.

O Turismo Cultural, representado pela classe “Cultural”, é caracterizado pelo deslocamento de turistas a núcleos receptores que oferecem como produto essencial o legado histórico do homem em distintas épocas, representado a partir do patrimônio e do acervo cultural. Da classe “Cultural” derivam cinco subclasses “Archeological Sites”, “Archive”, “Museum”, “Cultural Heritage” e “Ruin”.

O Turismo Ecológico, representado pela classe “Ecological”, é caracterizado pelo deslocamento de turistas para espaços naturais, com ou sem

receptivos, motivados pelo desejo de estar em contato com a natureza. Da classe “Ecological” derivam três subclasses “Climbing”, “Rafting” e “Ecological Trail”.

O Ecoturismo, representado pela classe “Ecotourism”, é caracterizado pelo deslocamento de turistas para espaços naturais delimitados e protegidos pelo estado ou controlados em parceria com associações. Da classe “Ecotourism” derivam duas subclasses “Environmental Conservation” e “Environmental Protection”.

O Turismo de Intercâmbio, representado pela classe “Exchange”, é caracterizado pelo deslocamento de turistas para outros países a fim de estudar ou estagiar. Da classe “Exchange” derivam duas subclasses “Stage” e “Course”.

O Turismo Religioso, representado pela classe “Religious”, é caracterizado pelo deslocamento de turistas que se destinam a centros religiosos, motivados pela fé em distintas crenças. Da classe “Religious” derivam cinco subclasses: “Church”, “Eucharistic Seminar”, “Temple”, “Pilgrimage” e “Retreat”.

O Turismo Científico, representado pela classe “Scientific”, é caracterizado pelo deslocamento de turistas potenciais que se dirigem a grandes centros universitários com atuação no setor de pesquisa e desenvolvimento. Da classe “Scientific” deriva uma subclasse “Research Center”.

O Turismo de Férias, representado pela classe “Vacation”, é caracterizado pelo deslocamento de turistas potenciais para grandes centros urbanos, litoral ou campo para poderem descansar e praticar lazer em família. Da classe “Vacation” derivam duas subclasses “Beach” e “Mountain”.

O Serviço Turístico é caracterizado pela prestação de serviços com o propósito de satisfazer os desejos e as expectativas do turista. Há quatro tipos de “Services”, conforme mostra a hierarquia de classes da ontologia (Figuras 51 e 52): “Accomodation, “Food”, “Transport” e “Entertainment”.

O Serviço Turístico de Hospedagem, representado pela classe “Accomodation”, é caracterizado pela prestação do serviço de hospedagem ao

turista. O turista pode se hospedar em hotel, albergue, pousada, chalé, camping e flat. Da classe “Accomodation” derivam seis subclasses “Hotel”, “Camping”, “Hostel”, “Cottage”, “Lodging” e “Flat”. As propriedades que caracterizam as classes “Hotel”, “Cottage” e “Flat” são: “room”, “bathroom”, “kitchen” e “living room”. A “área” é a propriedade que caracteriza a classe “Camping”. As propriedades que caracterizam as classes “Hostel” e “Lodging” são: “room” e “bathroom”.

O Serviço Turístico de Alimentação, representado pela classe “Food”, é caracterizado pela prestação do serviço de alimentação ao turista. O turista pode se alimentar em restaurante, pizzaria, sorveteria e lanchonete. Da classe “Food” derivam quatro subclasses: “Restaurant”, “Pizzeria”, “Snack Bar” e “Ice Cream”.

O Serviço Turístico de Entretenimento, representado pela classe “Entertainment”, é caracterizado pela prestação do serviço de entretenimento ao turista. O turista pode se divertir em parque de diversão, parque temático, cinema, teatro, clube, boate e casa de espetáculo. Da classe “Entertainment” derivam seis subclasses: “Park”, “Cinema”, “Club”, “Nightclub” e “House Show”.

O Serviço Turístico de Transporte, representado pela classe “Transport”, é caracterizado pela prestação do serviço de transporte ao turista. O turista pode se transportar por via aérea, marítima, ferroviária e rodoviária. Da classe “Transport” derivam quatro subclasses: “Air Transportation”, “Ocean Transportation”, “Ground Transportation” e “Rail Transportation”. Da classe “Air Transportion” derivam duas subclasses: “Airplane” e “Helicopter”. Da classe “Ocean Transportation” derivam três subclasses: “Ship”, “Launch” e “Ferry Boat”. Da classe “Ground Transportation” derivam três subclasses: “Automobile”, “Bus” e “Van”. Da classe “Rail Transportion” deriva uma subclasse: “Train”.

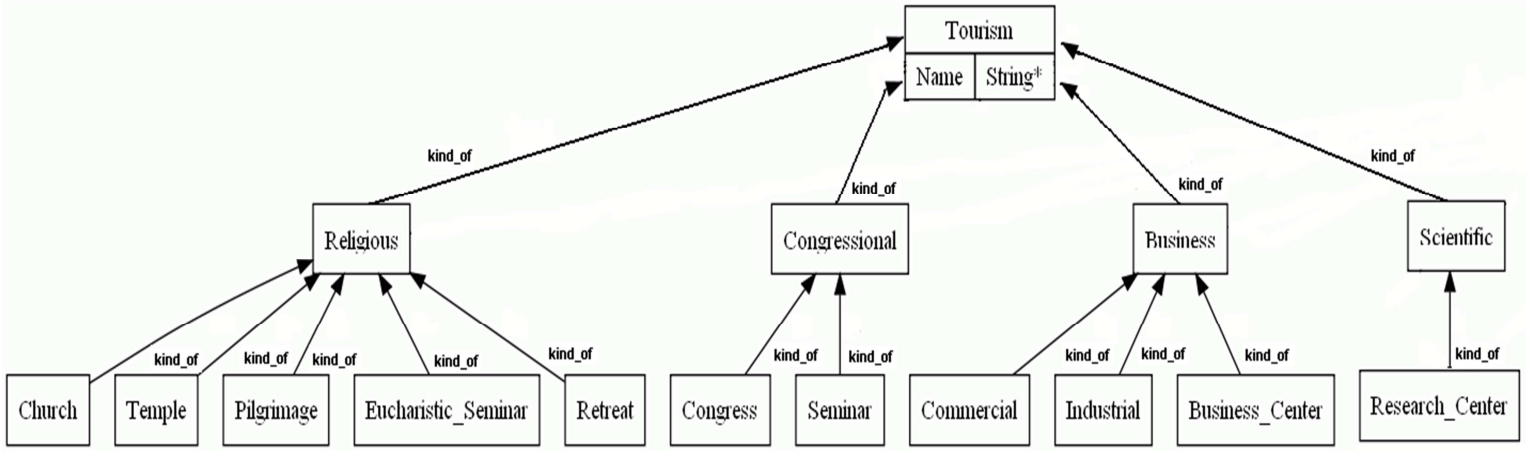


Figura 49: Classes, relacionamentos não taxonômicos e propriedades da ontologia OntoTur Parte I.

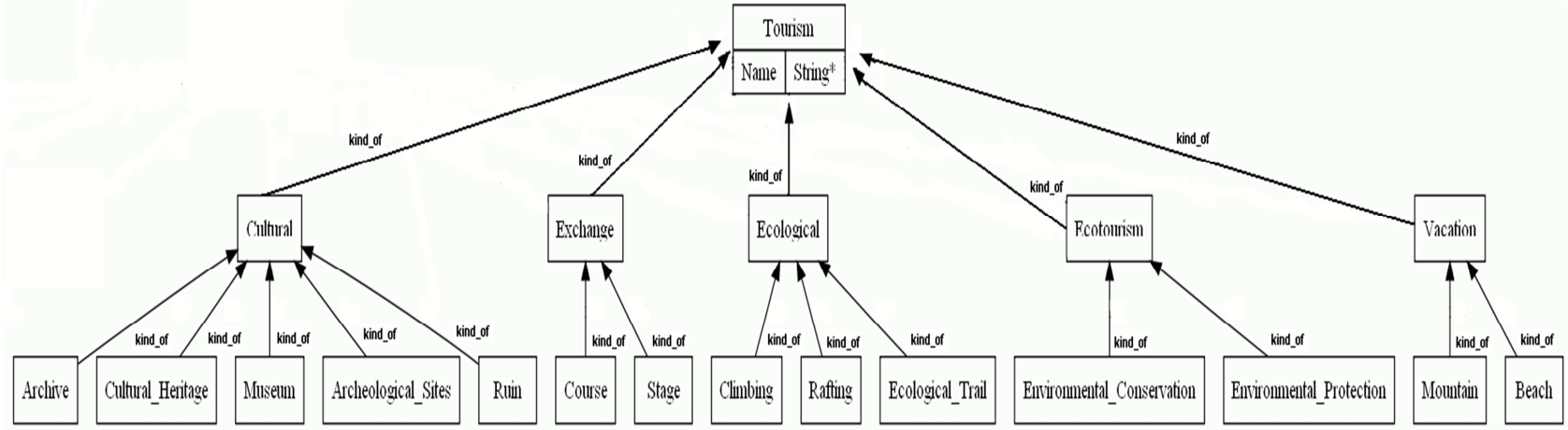


Figura 50: Classes, relacionamentos não taxonômicos e propriedades da ontologia OntoTur Parte II.

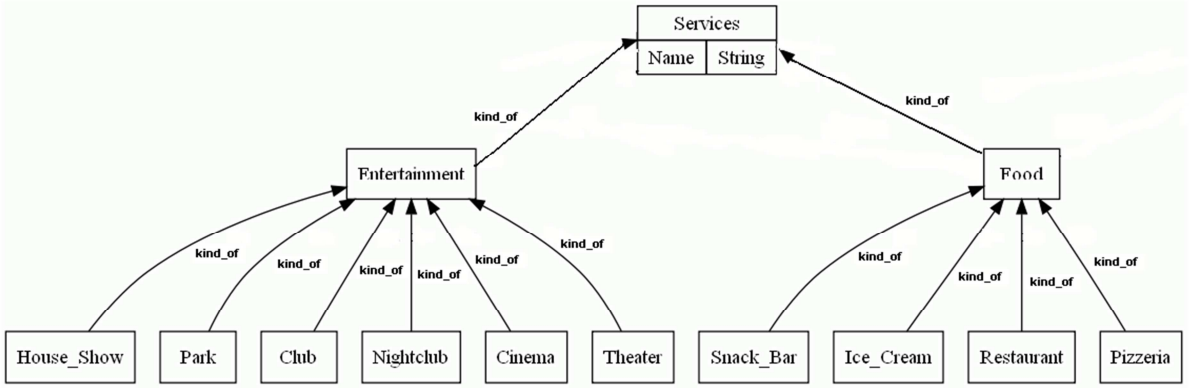


Figura 51: Classes, relacionamentos não taxonômicos e propriedades da ontologia OntoTur Parte III.

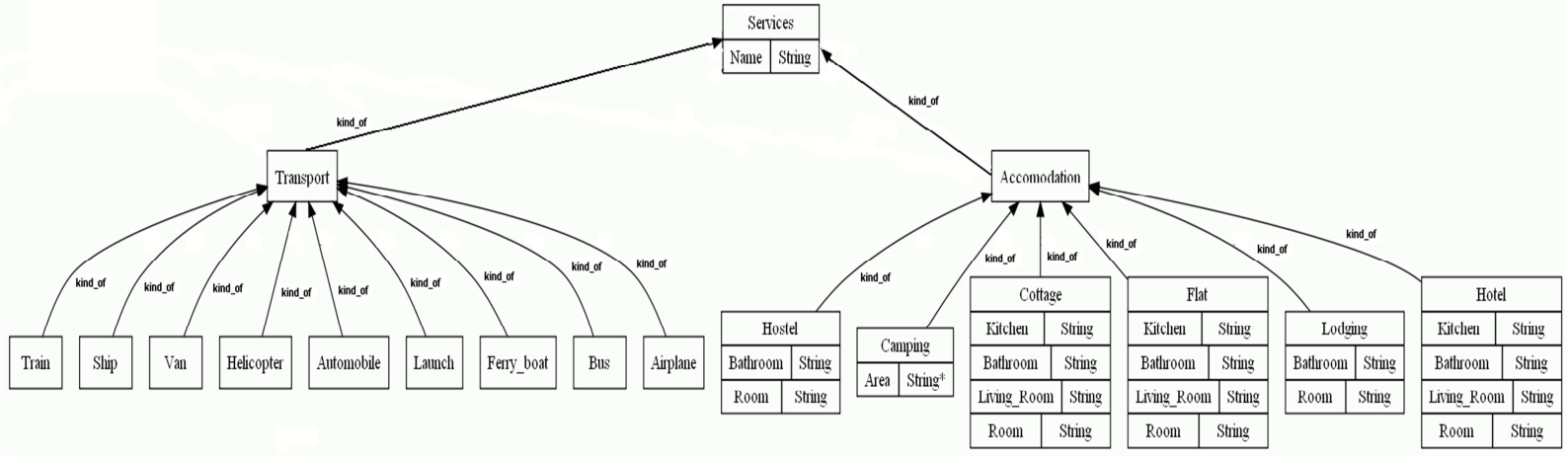


Figura 52: Classes, relacionamentos não taxonômicos e propriedades da ontologia OntoTur Parte IV.

5.2.3. Construção manual do classificador

Inicialmente o especialista de domínio analisa o corpus FamilyJuris (descrito na seção 5.1.1) para a especificação das regras que irão classificar as instâncias da ontologia FamilyLaw (descrita na seção 5.2.1).

Após a análise do corpus, o especialista de domínio especifica as regras em uma linguagem de especificação. A linguagem JAPE [18] (descrita no capítulo 2) foi escolhida por ser uma poderosa linguagem para a especificação de regras e é utilizado pelo GATE um poderoso sistema que aplica técnicas de PLN e EI em textos. Após a especificação das regras na linguagem JAPE, o classificador é testado. É dado como entrada o corpus e a ontologia para o classificador gerado manualmente que irá classificar as instâncias. As instâncias classificadas serão avaliadas pelo especialista de domínio para identificar se o classificador classificou corretamente ou não as instâncias.

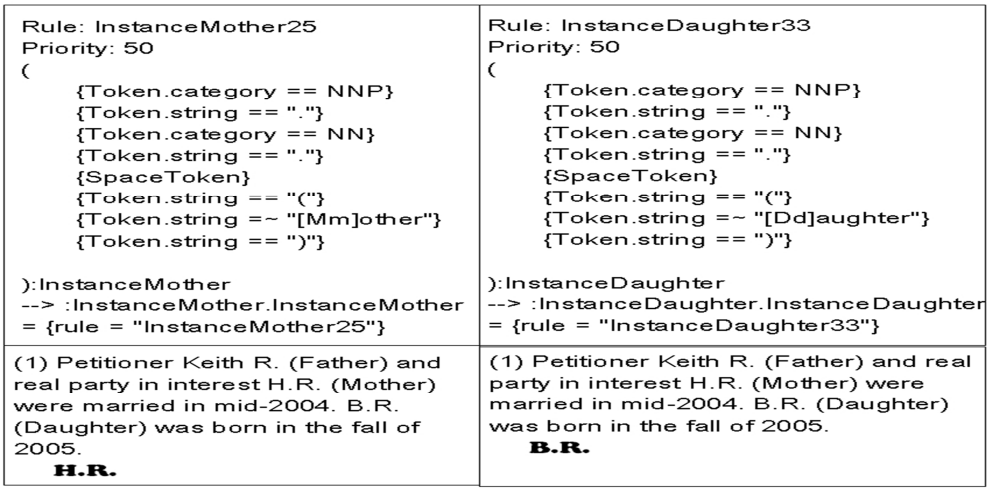
Uma regra especificada na linguagem JAPE [18] é do tipo < condição, ação >. Uma regra JAPE tem dois lados: o Esquerdo e o Direito. O lado esquerdo da regra contém uma expressão regular a ser detectada no conjunto de documentos. O lado direito descreve a ação a ser tomada sobre a expressão regular detectada. A Figura 53 ilustra dois exemplos de regras especificadas em JAPE com o uso da classes “mother” e “daughter” da ontologia FamilyLaw. O lado esquerdo da regra representa a expressão regular a ser detectada no corpus sugerindo instâncias da classe “mother” na primeira regra e da classe “daughter” na segunda regra. O lado direito da regra estabelece a ação a ser executada que é classificar “Nomes Próprios” como instâncias da classe “mother” na primeira regra e classificar “Nomes Próprios” como instâncias da classe “daughter” na segunda regra.

Por exemplo considerando o parágrafo do corpus FamilyJuris:

“Petitioner Keith R. (Father) and real party in interest H. R. (Mother) were married in mid 2004. B. R. (daughter) was born in the fall of 2005.”

O parágrafo do corpus FamilyJuris e a ontologia FamilyLaw são dados como entrada para o classificador (regras). O classificador aplica as regras neste parágrafo e são classificadas “H. R.” e “B. R.” como instâncias das classes “mother” e “daughter” respectivamente.

Os apêndices A e C mostram as regras desenvolvidas pelo especialista de domínio para o relacionamento não taxonômico “wife” da classe “Marriage” da ontologia “FamilyLaw” e para a propriedade “name” da classe “Hotel” da ontologia OntoTur, respectivamente.



```
Rule: InstanceMother25
Priority: 50
(
      {Token.category == NNP}
      {Token.string == "."}
      {Token.category == NN}
      {Token.string == "."}
      {SpaceToken}
      {Token.string == "("}
      {Token.string =~ "[Mm]other"}
      {Token.string == ")"}

):InstanceMother
--> :InstanceMother.InstanceMother
= {rule = "InstanceMother25"}
```

(1) Petitioner Keith R. (Father) and real party in interest H.R. (Mother) were married in mid-2004. B.R. (Daughter) was born in the fall of 2005.
H.R.

```
Rule: InstanceDaughter33
Priority: 50
(
      {Token.category == NNP}
      {Token.string == "."}
      {Token.category == NN}
      {Token.string == "."}
      {SpaceToken}
      {Token.string == "("}
      {Token.string =~ "[Dd]aughter"}
      {Token.string == ")"}

):InstanceDaughter
--> :InstanceDaughter.InstanceDaughter
= {rule = "InstanceDaughter33"}
```

(1) Petitioner Keith R. (Father) and real party in interest H.R. (Mother) were married in mid-2004. B.R. (Daughter) was born in the fall of 2005.
B.R.

Figura 53: Exemplos de regras de classificação na linguagem JAPE para a classes “mother” e “daughter”.

5.2.4. Geração automática do classificador

A geração automática do classificador foi realizada pela ferramenta DIAOP-Tool (descrita na seção 4.3) com a aplicação da fase “Construção de um Classificador” do processo proposto no capítulo 4. As regras são geradas automaticamente a partir da ontologia e da consulta a uma base de dados léxica. As regras são geradas em JAPE como explicado na seção 4.3 do capítulo 4. A Figura 54 ilustra um exemplo de regra gerada para o relacionamento não

taxonômico “wife” da classe “marriage” da ontologia FamilyLaw. O lado esquerdo da regra representa a expressão regular a ser detectada no corpus sugerindo a instância do relacionamento não taxonômico “wife”. O lado direito da regra estabelece a ação a ser executada que é classificar “Nomes Próprios” como instâncias do relacionamento não taxonômico “wife” da classe “marriage”.

```
Rule: Marriage_wife0
Priority: 50
(
        {Token.string == "wife" }
):Marriage_wife
-->
:Marriage_wife.Marriage_wife = { rule = "Marriage_wife0" ,findCategory="NNP",InterText="True",
RuleType="wife", owlPropName="wife_member", owlClassName="Marriage", owlRanger="#Person" }
```

Figura 54: Exemplo de regra gerada na linguagem JAPE para o relacionamento não taxonômico “wife” da classe “Marriage”.

A Figura 55 ilustra um exemplo de regra gerada para o relacionamento não taxonômico “husband” da classe “marriage” da ontologia FamilyLaw. O lado esquerdo da regra representa a expressão regular a ser detectada no corpus sugerindo a instância do relacionamento não taxonômico “husband”. O lado direito da regra estabelece a ação a ser executada que é classificar “Nomes Próprios” como instâncias do relacionamento não taxonômico “husband” da classe “marriage”.

```
Rule: Marriage_husband0
Priority: 50
(
        {Token.string == "husband" }
):Marriage_husband
-->
:Marriage_husband.Marriage_husband = { rule = "Marriage_husband0" ,findCategory="NNP",InterText="True",
RuleType="husband", owlPropName="husband_member", owlClassName="Marriage", owlRanger="#Person" }
```

Figura 55: Exemplo de regra gerada na linguagem JAPE para o relacionamento não taxonômico “husband” da classe “Marriage”.

A Figura 56 ilustra um exemplo de regra gerada para a propriedade “birth_date” da classe “person” da ontologia FamilyLaw. O lado esquerdo da regra representa a expressão regular a ser detectada no corpus sugerindo a instância da propriedade “birth_date”. O lado direito estabelece a ação a ser executada que é classificar “Datas” como instâncias da propriedade “birth_date” da classe “person”.

```
Rule: Person_Birth0
Priority: 50
(
        {Token.string =~ "[Bb]irth" }
        {SpaceToken}
        {Token.string =~ "[io]n" }
        {SpaceToken}
        {Token.category == CD }
):Person_Birth
-->
:Person_Birth.Person_Birth = { rule = "Person_Birth0" ,InterText="True", RuleType="birth [io]n @CD",
owlPropName="Date_of_Birth", owlClassName="Person", owlRanger="http://www.w3.org/2001/XMLSchema#date" }
```

Figura 56: Exemplo de regra gerada na linguagem JAPE para a propriedade “birth_date” da classe “Person”.

Os apêndices B e D mostram as regras geradas automaticamente para o relacionamento não taxonômico “wife” da classe “Marriage” da ontologia “FamilyLaw” e para a propriedade “name” da classe “Hotel” da ontologia OntoTur, respectivamente.

5.2.5. Análise da aplicação do classificador construído de forma manual e gerado de forma automática nos corpora FamilyJuris e Turístico e nas ontologias FamilyLaw e OntoTur

Esta seção apresenta uma análise da viabilidade da fase “Construção de um Classificador” com a aplicação do classificador construído de forma manual e gerado de forma automática nos corpora FamilyJuris e Turístico e nas ontologias FamilyLaw e Ontotur utilizando as métricas precisão, *recall* e medida-F.

As ontologias FamilyLaw (descrita na seção 5.2.1) e OntoTur (descrita na seção 5.2.2) e os corpora FamilyJuris [54] (descrito na seção 5.1.1) e Turístico (descrito na seção 5.2) foram utilizados como entradas para a fase “Construção de um Classificador” nos experimentos realizados. Do corpus FamilyJuris foram utilizados 30 documentos aleatórios, cada documento foi analisado manualmente para identificar e classificar as instâncias, sendo classificadas um total de 104 instâncias nesse corpus. Em relação ao corpus Turístico, cada documento foi analisado manualmente para identificar e classificar as instâncias, sendo classificadas um total de 80 instâncias nesse corpus.

A Tabela 17 resume os resultados da avaliação da fase “Construção de um Classificador”.

Avaliação do Classificador	**Classificador Manual**		**Classificador Automático**	
	FamilyJuris 30 documentos	**OntoTur 34 documentos**	**FamilyJuris 30 documentos**	**OntoTur 34 documentos**
Instâncias no corpus	104	80	104	80
Instâncias classificadas	78	62	103	81
Instâncias corretamente classificadas	70	61	90	52
Classes	8	69	8	69
Propriedades	4	18	4	18
Relacionamentos não taxonômicos	27	0	27	0
Regras geradas/ especificadas	1617	196	61	826
Precisão	89,7%	98%	87,3%	76,5%
Recall	67,3%	76%	86,5%	77,5%
Medida-F	76,9%	85,6%	86,8%	76,9%

Tabela17: Resultado dos experimentos da fase “Construção de um Classificador”

Um primeiro experimento foi realizado aplicando o classificador construído manualmente no corpus FamilyJuris e na ontologia FamilyLaw. As instâncias classificadas foram 78, as instâncias corretamente classificadas foram 70 das 104 instâncias no corpus. Obteve-se, portanto, uma precisão de 89,7%, um *recall* de 67,3% e uma medida-F de 76,9%.

Um segundo experimento foi realizado aplicando o classificador gerado automaticamente no corpus FamilyJuris e na ontologia FamilyLaw. As instâncias classificadas foram 103, as instâncias corretamente classificadas foram 90 das 104 instâncias no corpus. Obteve-se, portanto, uma precisão de 87,3%, um *recall* de 86,5% e uma medida-F de 86,8%.

Um terceiro experimento foi realizado aplicando o classificador construído manualmente no corpus Turístico e na ontologia OntoTur. As instâncias classificadas foram 62, as instâncias corretamente classificadas foram 61 das 80 instâncias no corpus. Obteve-se, portanto, uma precisão de 98%, um *recall* de 76% e uma medida-F de 85,6%.

Um quarto experimento foi realizado aplicando o classificador gerado automaticamente no corpus Turístico e na ontologia OntoTur. As instâncias classificadas foram 81, as instâncias corretamente classificadas foram 62 das 80 instâncias no corpus. Obteve-se, portanto, uma precisão de 76,5%, um *recall* de 77,5% e uma medida-F de 76,9%.

Uma desvantagem do classificador construído de forma manual é que é dependente de um especialista de domínio e este especialista necessita de tempo para analisar o corpus, desenvolver e especificar as regras em uma linguagem elevando os custos. Ao contrário do que ocorre com a geração do classificador de forma automática, que não depende de um especialista de domínio e mesmo assim apresenta boa efetividade.

5.2.6. Discussão dos resultados obtidos com a aplicação do classificador construído de forma manual e gerado de forma automática nos corpora FamilyJuris e Turístico e nas ontologias FamilyLaw e OntoTur

Considerando os resultados dos experimentos realizados (Tabela 17) podemos concluir que o classificador construído de forma manual apresenta melhores resultados do que o classificador gerado de forma automática. Isso se deve ao fato de que o classificador construído de forma manual é desenvolvido por um especialista de domínio por isso é mais preciso, enquanto que o classificador gerado de forma automática não depende de um especialista de domínio e apresenta boa efetividade.

Para o corpus FamilyJuris e a ontologia FamilyLaw o especialista de domínio desenvolveu um total de 1617 regras, enquanto que 61 regras foram geradas automaticamente pelo processo DIAOP-Pro. O domínio da linguagem de especificação das regras influencia diretamente na quantidade de regras especificadas pelo especialista de domínio. Na primeira especificação das regras no domínio jurídico, o especialista de domínio especificou uma enorme quantidade de regras. Essas regras desenvolvidas pelo especialista de domínio são muito específicas e classificavam somente uma instância, ao contrário das regras geradas de forma automática que foram em menor quantidade, mas classificam um maior número de instâncias, pois são regras genéricas.

Para o corpus Turístico e a ontologia OntoTur o especialista de domínio desenvolveu um total de 196 regras, enquanto que 826 regras foram geradas automaticamente pelo processo DIAOP-Pro. Já na segunda especificação das regras no domínio turístico, o especialista de domínio já tinha a experiência com a linguagem de especificação de regras, então especificou um número menor de regras, enquanto que uma grande quantidade de regras foram geradas automaticamente. Isso se deve ao fato da ontologia OntoTur ter uma grande quantidade de classes e sinônimos dessas classes no Wordnet, a base de dados léxica utilizada.

Considerando os resultados obtidos no segundo estudo de caso, a geração automática do classificador mostrou-se viável e foi escolhida para ser aplicada na fase “Construção de um Classificador” do processo proposto.

5.3.Estudo de Caso III: Povoamento da Ontologia FamilyLaw

O estudo de caso consistiu na utilização do corpus FamilyJuris e na utilização da ontologia FamilyLaw como entradas do processo DIAOP-Pro proposto, através da aplicação da ferramenta DIAOP-Tool assim como na avaliação dos resultados obtidos.utilizando as métricas precisão, *recall* e medida-F, descritas na seção 2.7 do capítulo 2.

As próximas seções detalham o estudo de caso que está organizado como segue. A seção 5.3.1 aplica o processo proposto no povoamento automático da FamilyLaw a partir do corpus FamilyJuris e finalmente a seção 5.3.2 descreve a avaliação do povoamento automático do FamilyLaw.

5.3.1. Aplicação do processo DIAOP-Pro proposto

O povoamento da ontologia FamilyLaw foi realizada através da execução das fases “Identificação de Instâncias Candidatas”, “Construção de um Classificador” e “Classificação de Instâncias” do processo proposto, através da aplicação da ferramenta DIAOP-Tool. O processo proposto é ilustrado através de exemplos utilizando o corpus FamilyJuris e a ontologia FamilyLaw.

5.3.1.1. Identificação de Instâncias Candidatas

A fase “Identificação de Instâncias Candidatas” consiste das seguintes tarefas: “Análise Morfo-Lexical”, “Reconhecimento de Entidades Nomeadas” e “Identificação de Co-Referencias”. Para a realização da “Análise Morfo-Lexical”

utiliza-se como entrada o texto de um documento do corpus FamilyJuris (Figura 57).

OPINION

MCADAMS, J.-

In this marital dissolution case, both parties challenge a spousal support order. In her appeal, the wife asserts that the trial court abused its discretion in reducing temporary spousal support and in setting permanent support. She contends that the court failed to account for all of the husband's income and that it erred in imputing investment income to her. In his cross-appeal, the husband contends that the permanent spousal support order unfairly charges him a second time for earnings from the business that he operates. In his view, since the wife received half of the business's going-concern value in the property division, the court should not consider the entire stream of business income in {Slip Opn. Page 2} assessing his ability to pay support. The husband asks us to announce a new rule in California prohibiting such "double dipping."

Rejecting both parties' contentions, we affirm the challenged order. In the published part of the opinion, we conclude that the trial court acted within its discretion in determining the husband's income for purposes of spousal support.

BACKGROUND fn. *

The parties to this appeal are Scott Blazer (husband) and Karen Nickles Blazer (wife). They married in November 1982 and separated in January 2002. There are two children of the marriage, both now adults.

The principal marital asset was a company created in 1996 by husband and a business partner, called Blazer-Wilkinson LLC (BW). BW is a brokerage company that buys and sells produce, principally strawberries and bush berries. In 2004, the court valued the community interest in BW at $5.6 million.

2002 -- Dissolution; Temporary Support

On January 31, 2002, husband petitioned for dissolution of the marriage in Monterey County Superior Court. In October 2002, the marriage was dissolved.

Figura 57: Exemplo de um documento do corpus FamilyJuris [54]

O resultado da tarefa “Análise Morfo-Lexical” para o texto da Figura 57 é mostrado na Figura 58.

OPINION [NNP] MCADAMS [NNP], [,] J [NNP]. [.] – [-] In [IN] this [DT] marital [JJ] dissolution [NN] case [NN], [,] both [DT] parties [NNS] challenge [VBP] a [DT] spousal [NN] support [NN] order [NN]. [.] In [IN] her [PRP] appeal [NN], [,] the [DT] wife [NN] asserts [VBZ] that [IN] the [DT] trial [NN] court [NN] abused [VBD] its [PRP] discretion [NN] in [IN] reducing [VBG] temporary [JJ] spousal [NN] support [NN] and [CC] in [IN] setting [VBG] permanent [JJ] support [NN]. [.] She [PRP] contends [VBZ] that [IN] the [DT] court [NN] failed [VBD] to [TO] account [VB] for [IN] all [DT] of [IN] the [DT] husband [NN] 's [POS] income [NN] and [CC] that [IN] it [PRP] erred [VBG] in [IN] imputing [NN] investment [NN] income [NN] to [TO] her [PRP]. [.] In [IN] his [PRP] cross-appeal [JJ], [,] the [DT] husband [NN] contends [VBZ] that [IN] the [DT] permanent [JJ] spousal [NN] support [NN] order [NN] unfairly [RB] charges [VBZ] him [PRP] a [DT] second [JJ] time [NN] for [IN] earnings [NNS] from [IN] the [DT] business [NN] that [DT] he [PRP] operates [VBZ]. [.] In [IN] his [PRP] view [NN], [,] since [IN] the [DT] wife [NN] received [VBD] half [NN] of [IN] the [DT] business [NN] 's [POS] going-concern [JJ] value [NN] in [IN] the [DT] property [NN] division [NN], [,] the [DT] court [NN] should [MD] not [RB] consider [VB] the [DT] entire [JJ] stream [NN] of [IN] business [NN] income [NN] in [IN] { [{] Slip [NNP] Opn [NNP]. [.] Page [NNP] 2 [LD]} [{] assessing [VBG] his [PRP] ability [NN] to [TO] pay [VB] support [NN]. [.] The [DT] husband [NN] asks [VBZ] us [PRP] to [TO] announce [VB] a [DT] new [JJ] rule [NN] in [IN] California [NNP] prohibiting [VBG] such [JJ] " [“] double [JJ] dipping [VBG]. [.] " [“] Rejecting [NNP] both [DT] partie [NNS] s' [POS] contentions [NNS], [,] we [PRP] affirm [NN] the [DT] challenged [VBN] order [NN]. [.] In [IN] the [DT] published [VBN] part [NN] of [IN] the [DT] opinion [NN], [,] we [PRP] conclude [VBP] that [IN] the [DT] trial [NN] court [NN] acted [VBD] within [IN] its [PRP] discretion [NN] in [IN] determining [VBG] the [DT] husband [NN] 's [POS] income [NN] for [IN] purposes [NNS] of [IN] spousal [NN] support [NN]. [.]BACKGROUND [NNP] fn [NN]. [.] * [*] The [DT] parties [NNS] to [TO] this [DT] appeal [NN] are [VBP] Scott [NNP] Blazer [NNP] ([(] husband [NN]) [)] and [CC] Karen [NNP] Nickles [NNP] Blazer [NNP] ([(] wife [NN]) [)]. [.] They [PRP] married [VBD] in [IN] November [NNP] 1982 [CD] and [CC] separated [JJ] in [IN] January [NNP] 2002 [CD]. [.]There [EX] are [VBP]

Figura 58: Parte do resultado da análise morfo-lexical do texto da Figura 57

O resultado da tarefa “Reconhecimento de Entidades Nomeadas” para o texto da Figura 57 são conjuntos de três entidades do tipo “Pessoa” (“Scott Blazer”,

“Karen Nickles Blazer” e “MCADAMS, J”), uma entidade do tipo “Localização” (“California”) e uma entidade do tipo “Organização” (“Blazer-Wilkinson LLC”).

O resultado da tarefa “Identificação de Co-Referências” para o texto da Figura 57 são as co-referências pronominais listadas na Tabela 18.

Tokens	***Co-Referências Pronominais***
Scott Blazer	his, he, him
Karen Nickles Blazer	her, she

Tabela 18: Resultado das co-referências pronominais para o texto da Figura 57

5.3.1.2. Construção de um Classificador

A fase “Construção de um Classificador” consiste das seguintes tarefas: “Seleção de Classes, Propriedades e Relacionamentos”, “Seleção de Triggers” e “Geração de Regras”.

A tarefa “Seleção de Classes, Propriedades e Relacionamentos” é realizada a partir de uma ontologia. A Figura 59 mostra a classe “Person” com as propriedades: “birth_year”, “divorce” e “constitutive” e os relacionamentos não taxonômicos “married” e “father”, como exemplo do produto desta tarefa.

O resultado da tarefa “Seleção de Triggers” para as propriedades e os relacionamentos não taxonômicos identificados é mostrado na Figura 60.

O resultado da tarefa “Geração de Regras” para a classe “Person” da ontologia FamilyLaw e para o corpus ilustrado na Figura 57:

```
frase_nominal (“Scott Blazer”) ∧ instância (“Scott Blazer_instância”,
“Scott Blazer”) ∧ trigger (“Scott Blazer_instância”, “husband”) ∧
relacionamento_sinônimo (“husband”, “husband”) ∧
relacionamento_não_taxonômico (“husband”, “Person”)⇒ é_um (“Scott
```

Blazer_instância", "Person") ∧ relacionamento_não_taxonômico_associação ("Scott Blazer_instância", "husband", "Person")

frase_nominal ("Karen Nickles Blazer") ∧ instância ("Karen Nickles Blazer_instância", "Karen Nickles Blazer") ∧ trigger ("Karen Nickles Blazer_instância", "wife") ∧ relacionamento_sinônimo ("wife", "wife") ∧ relacionamento_não_taxonômico ("wife", "Person")⇒ é_um ("Karen Nickles Blazer_instância", "Person") ∧ relacionamento_não_taxonômico_associação ("Karen Nickles Blazer_instância", "wife", "Person")

frase_nominal ("They") ∧ instância ("They_instância", "They") ∧ trigger ("They", "married in") ∧ propriedade_sinônimo ("married in", "constitutive") ∧ propriedade ("constitutive", "person", "1982") ⇒ é_um ("They_instância", "Person") ∧ propriedade_associação ("They_instância", "constitutive", "1982" "Person")

frase_nominal ("They") ∧ instância ("They_instância", "They") ∧ trigger ("They", "separated in") ∧ propriedade_sinônimo ("separated in", "divorce") ∧ propriedade ("divorce", "person", "2002") ⇒ é_um ("They_instância", "Person") ∧ propriedade_associação ("They_instância", "divorce", "2002" "Person")

Person		
married	Instance	Person
father	Instance	Person
birth_year		Date
divorce		Date
constitutive		Date

Figura 59: Classe "Person" da ontologia FamilyLaw

Property	Trigger	Non Taxonomic Relationship	Trigger
Birth_Year	Born	Married	Married
	Birth		
Divorce	Divorce		Marry
	Divorcement		
	Disjoint		Spouse
	Disunite		
	Dissociate	Father	Father
	Disassociate		
Constitutive	Constitutive		Male Parent
	Marriage		
	Matrimony		Begetter
	Marriage		

Figura 60: Triggers identificados da classe “Person” da ontologia FamilyLaw.

5.3.1.3. Classificação de Instâncias

A fase “Classificação de Instâncias” consiste de duas tarefas: “Associação de Instâncias” e “Instanciação”.

O resultado da tarefa “Associação de Instâncias” é o conjunto I’ = *{wife_member (“Marriage1”, “Karen Nickles Blazer”), husband_member(“Marriage1”, “Scott Blazer”), dissolution_date(“Marriage1”, “2002”), constitutive_date(“Marriage1”, “1982”)*}.

O resultado da tarefa “Instanciação” (Figura 61) são as classes *“Person” e “Marriage”* da ontologia do Direito de família povoadas.

5.3.2. Avaliação do povoamento automático da FamilyLaw

Esta seção apresenta uma avaliação da efetividade do processo proposto a partir do corpus FamilyJuris e da ontologia FamilyLaw utilizando as métricas precisão, *recall* e medida-F.

Cada documento foi analisado manualmente para identificar e classificar as instâncias, sendo identificadas e classificadas um total de 794 instâncias nesse corpus. As instâncias identificadas e classificadas foram 783, as instâncias corretamente identificadas e classificadas foram 711 das 794 instâncias no corpus. Obteve-se, portanto, uma precisão de 90%, um *recall* de 89,50% e uma medida-F de 89,74% como mostra a Tabela 19. O DIAOP-Pro apresentou boa efetividade no domínio do Direito de família.

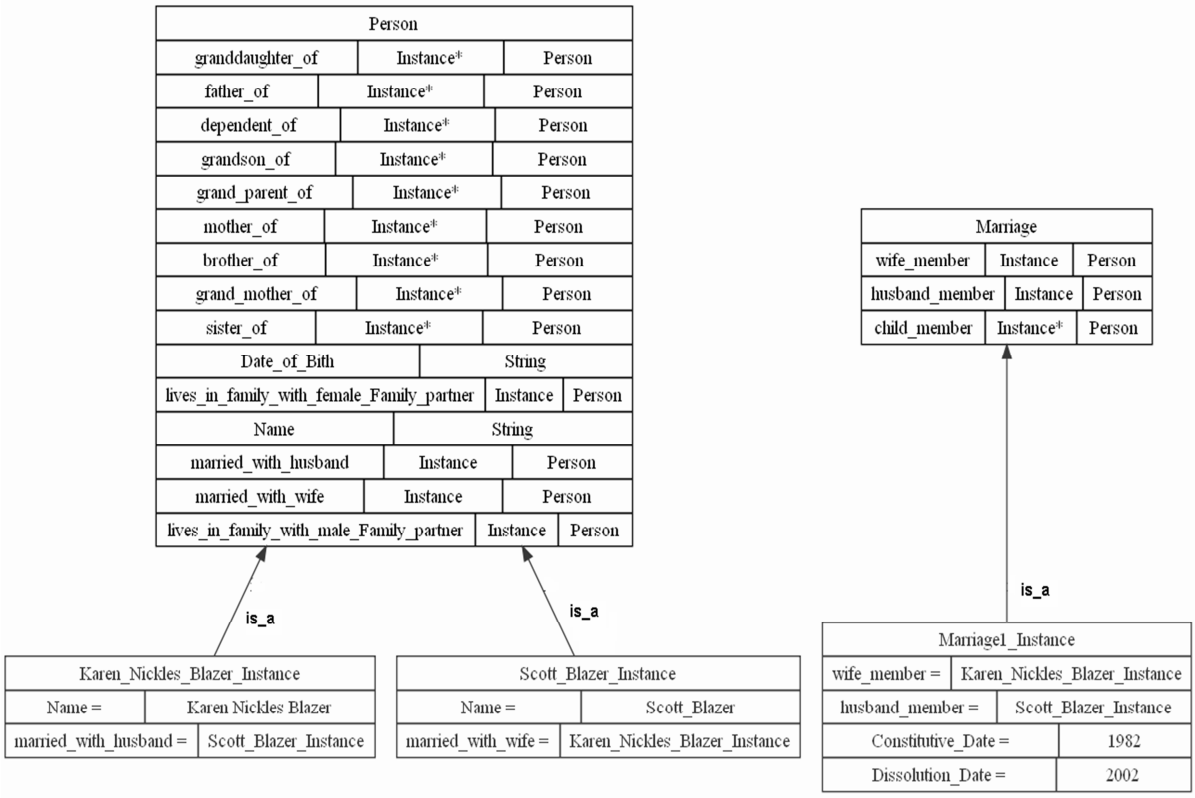


Figura 61: Classes “Marriage” e “Person” povoadas

5.4. Estudo de Caso IV: Povoamento da Ontologia OntoTur

O estudo de caso consistiu na utilização do corpus Turístico e da ontologia OntoTur como entradas do processo DIAOP-Pro proposto, através da aplicação da ferramenta DIAOP-Tool (descrita na seção 4.3) assim como na

avaliação dos resultados obtidos utilizando as métricas precisão, *recall* e medida-F, descritas na seção 2.7 do capítulo 2.

As próximas seções detalham o estudo de caso que está organizado como segue. A seção 5.4.1 aplica o processo proposto no povoamento automático da OntoTur a partir do corpus Turístico e a seção 5.4.2 descreve a avaliação do povoamento automático da OntoTur.

Avaliação	***FamilyJuris***
Instâncias no corpus	794
Instâncias identificadas e classificadas	783
Instâncias corretamente identificadas e classificadas	711
Precisão	90%
Recall	89,50%
Medida-F	89,74%

Tabela 19: Resultado do experimento do povoamento automático da FamilyLaw a partir do corpus FamilyJuris

5.4.1. Aplicação do processo DIAOP-Pro proposto

O povoamento da ontologia OntoTur foi realizada através da execução das fases "Identificação de Instâncias Candidatas", "Construção de um Classificador" e "Classificação de Instâncias" do processo proposto, através da aplicação da ferramenta DIAOP-Tool (descrita na seção 4.3). O processo proposto é ilustrado através de exemplos utilizando o corpus Turístico e a ontologia OntoTur.

5.4.1.1. Identificação de Instâncias Candidatas

A fase "Identificação de Instâncias Candidatas" consiste das seguintes tarefas: "Análise Morfo-Lexical", "Reconhecimento de Entidades Nomeadas" e

“Identificação de Co-Referencias”. Para a realização da “Análise Morfo-Lexical” utiliza-se como entrada o texto de um documento do corpus Turístico (Figura 62).

Hotel De Tassche This beautiful step-fronted traditional house has been an Inn since at least 1682 (though the façade is inscribed 1710). It's family-run, beautifully maintained and superbly positioned just a block off Bruges's central square. Behind the historic façade, room décor is modern and gently stylish. Some bed covers are a fascinatingly iridescent blue-black colour. Above on the grey bed-boards in calligraphic script is the phrase Zoete Zachte Dromen (Sweet soft dreams). The bathrooms are gleaming white cubes with decent showers. Even if you're not staying on the top floor it's worth taking the lift to have a look at the roof structure. Like in many Bruges houses the original beams remain (these date to 1450), and are held in place by wooden pegs. The beam-structure was prefabricated on the ground, then taken apart like a jigsaw puzzle to reassemble in situ. You can still see carpenters ' marks to show where each piece went. Rather than divide the high ceiling, the hotel has cunningly created boxes for individual upper-floor rooms. From the wiggling corridor between them the whole apex of the roof beams is visible.

Figura 62: Exemplo de um documento do corpus do domínio turístico

O resultado da tarefa “Análise Morfo-Lexical” para o corpus ilustrado na Figura 62 é mostrado na Figura 63.

5.4.1.2. Construção de um Classificador

A fase “Construção de um Classificador” consiste das seguintes tarefas: “Seleção de Classes, Propriedades e Relacionamentos”, “Seleção de Triggers” e “Geração de Regras”.

A tarefa “Seleção de Classes, Propriedades e Relacionamentos” é realizada a partir de uma ontologia. A Figura 64 mostra a classe “Hotel” com as propriedades: “name”, “kitchen”, “bathroom”, “living room” e “room”, como exemplo do produto desta tarefa.

Hotel [NNP] De [NNP] Tassche [NNP] This [DT] beautiful [JJ] step-fronted [JJ] traditional [JJ] house [NN] has [VBZ] been [VBN] an [DT] Inn [NNP] since [IN] at [IN] least [JJS] 1682 [CD] (though [IN] the [DT] façade [NN] is [VBZ] inscribed [JJ] 1710 [CD]). It[PRP]'s [VBZ] family-run [JJ], beautifully [RB] maintained [VBN] and [CC] superbly [RB] positioned [VBN] just [RB] a [DT] block [NN] off [RP] Bruges[NNP]'s [POS] central [JJ] square [NN]. Behind [IN] the [DT] historic [JJ] façade [NN], room [NN] décor [NN] is [VBZ] modern [JJ] and [CC] gently [RB] stylish [JJ]. Some [DT] bed [NN] covers [NNS] are [VBP] a [DT] fascinatingly [RB] iridescent [NN] blue-black [JJ] colour [NN]. Above [IN] on [IN] the [DT] grey [JJ] bed-boards [JJ] in [IN] calligraphic [JJ] script [NN] is [VBZ] the [DT] phrase [NN] Zoete [NNP] Zachte [NNP] Dromen [NNP] (Sweet [NNP] soft [JJ] dreams [NNS]). The [DT] bathrooms [NNS] are [VBP] gleaming [VBG] white [JJ] cubes [NNS] with [IN] decent [JJ] showers [NNS]. Even [RB] if [IN] you[PRP]'re [VBP] not [RB] staying [VBG] on [IN] the [DT] top [JJ]

Figura 63: Parte do resultado da análise morfo-lexical do texto da Figura 62

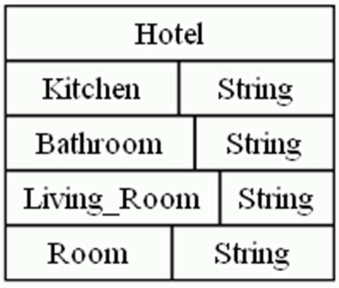


Figura 64: Classe Hotel da Ontologia OntoTur

O resultado da tarefa “Seleção de Triggers” para as propriedades da classe “Hotel” é mostrado na Figura 65.

O resultado da tarefa “Geração de Regras” para a classe “Hotel” da ontologia FamilyLaw e para o corpus ilustrado na Figura 62:

frase_nominal (“Hotel De Tassche”) ∧ instância (“Hotel De Tassche_instância”, “Hotel De Tassche”) ∧ trigger (“Hotel De Tassche”, “hotel”) ∧ propriedade_sinônimo (“hotel”, “hotel”) ∧ propriedade (“hotel”, “hotel”, “De Tasshe”) ⇒ é_um (“Hotel De Tassche_instância”, “hotel”) ∧

propriedade_associação (“Hotel De Tassche_instância”, “hotel”, “De Tasshe”, “hotel”)

Property	Ontology Element Indicator
Hotel	Hotel
Bathroom	Bathroom
	Lavatory
	Toilet
Kitchen	Kitchen
Living room	Living room
	Sitting room
Room	Room
	Board

Figura 65: Triggers identificados da classe “Hotel” da ontologia OntoTur.

5.4.1.3. Classificação de Instâncias

A fase “Classificação de Instâncias” consiste de duas tarefas: “Associação de Instâncias” e “Instanciação”.

O resultado da tarefa “Associação de Instâncias” é o conjunto I’ = *{“hotel” (“Hotel1”, “De Tassche”)*}.

O resultado da tarefa “Instanciação” (Figura 66) é a classe *“Hotel”* da ontologia do domínio turístico povoada.

5.4.2. Avaliação do povoamento automático da ontologia OntoTur

Esta seção apresenta uma avaliação da efetividade do processo proposto a partir do corpus Turístico e da ontologia OntoTur utilizando as métricas precisão, recall e medida-F.

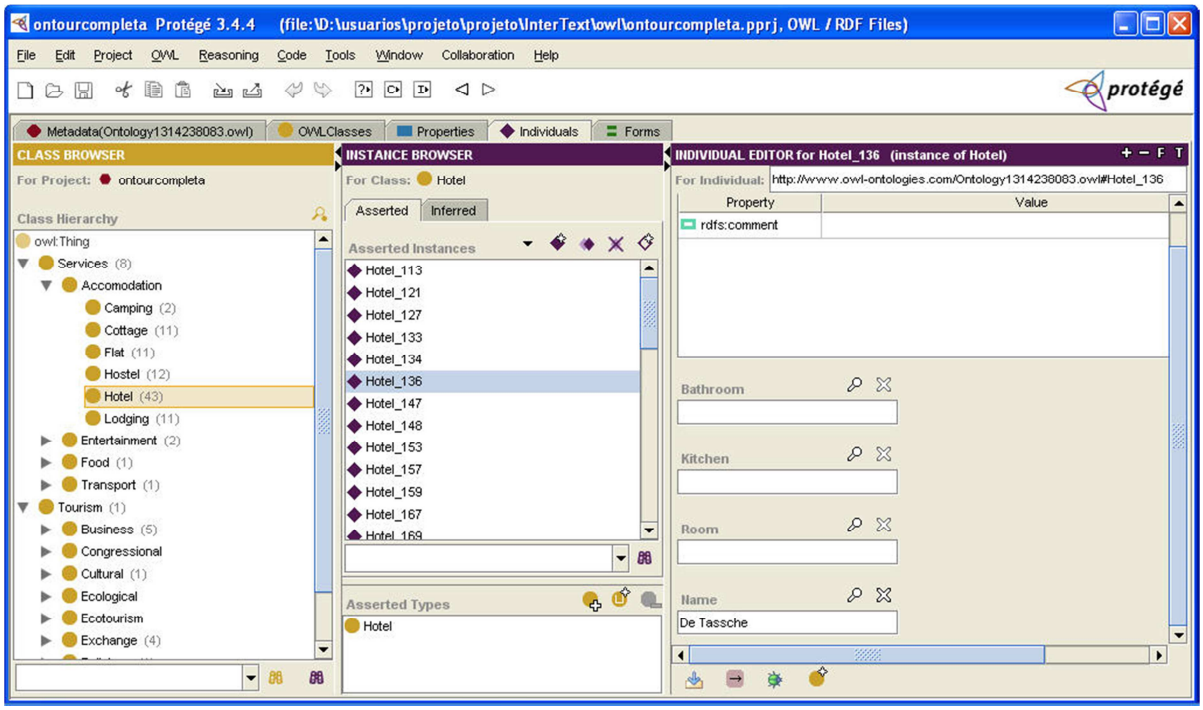


Figura 66: Classe “Hotel” povoada

Cada documento foi analisado manualmente para identificar e classificar as instâncias, sendo identificadas e classificadas um total de 80 instâncias nesse corpus. As instâncias identificadas e classificadas foram 81, as instâncias corretamente identificadas e classificadas foram 52 das 80 instâncias no corpus. Obteve-se, portanto, uma precisão de 76,50%, um recall de 77,50% e uma Medida-F de 76,90% como mostra a Tabela 20. O DIAOP-Pro apresentou boa efetividade no domínio turístico.

5.5.Considerações Finais

Neste capítulo foram apresentados os estudos de caso realizados com o objetivo de avaliar a efetividade e a viabilidade do processo proposto. O primeiro estudo de caso realizou um estudo comparativo da aplicação de técnicas

estatísticas e de técnicas puramente lingüísticas na fase “Identificação de Instâncias Candidatas” e foi realizado paralelamente ao processo DIAOP-Pro, sendo de fundamental importância para o refinamento do mesmo. Os resultados dos experimentos são mostrados na Tabela 16 e podemos concluir que a aplicação de técnicas puramente lingüísticas oferece melhores resultados que a aplicação de técnicas estatísticas podendo ser aplicadas em corpus de qualquer tamanho. Então a aplicação de técnicas puramente lingüísticas foi escolhida para ser aplicada no processo proposto auxiliando assim sua avaliação parcial.

Avaliação	***Turístico***
Instâncias no corpus	80
Instâncias identificadas e classificadas	81
Instâncias corretamente identificadas e classificadas	52
Precisão	76,50%
Recall	77,50%
Medida-F	76,90%

Tabela 20: Resultado do experimento do povoamento automático da OntoTur a partir do corpus Turístico

O segundo estudo de caso realizou uma análise da viabilidade da geração automática do classificador na fase “Construção de um Classificador” e aplicando-os em corpora FamilyJuris e Turístico e nas ontologias FamilyLaw e OntoTur. Os resultados dos experimentos são mostrados na Tabela 17 e podemos concluir que o classificador construído de forma manual apresenta melhores resultados que o classificador gerado de forma automática. Entretanto o classificador construído de forma manual é dependente de um especialista de domínio para realizar uma tarefa custosa, tediosa, dispendiosa e sujeita a erros. Enquanto que o classificador gerado de forma automática possui a vantagem de não depender de um especialista de domínio e automatizar uma tarefa custosa, tediosa, dispendiosa e sujeita a erros sem afetar a efetividade. Então, a geração

automática do classificador apresentou-se viável e foi aplicada no processo proposto.

O terceiro estudo de caso realizou o povoamento automático da ontologia FamilyLaw a partir do corpus FamilyJuris com o objetivo de avaliar a efetividade do processo proposto, sendo importante, pois auxiliou também na avaliação das suas fases, atividades e produtos. O processo proposto DIAOP-Pro mostrou-se efetivo no domínio do Direito de família como mostra a Tabela 19. A ontologia FamilyLaw povoada poderá ser utilizada na execução de sistemas baseados em conhecimento para auxiliar a tomada de decisões.

O quarto estudo de caso realizou o povoamento automático da ontologia OntoTur a partir do corpus Turístico com o objetivo de avaliar a efetividade do processo proposto em um outro domínio distinto, sendo importante, pois auxiliou a verificação da independência de domínio. O DIAOP-Pro mostrou-se efetivo no domínio turístico como mostra a Tabela 20. A ontologia OntoTur povoada poderá ser utilizada também na execução de sistemas baseados em conhecimento para auxiliar a tomada de decisões.

A Tabela 21 resume o resultado dos experimentos na avaliação do processo DIAOP-Pro proposto.

No próximo capítulo são apresentadas as conclusões destacando os resultados obtidos e os trabalhos futuros que irão dar continuidade a esta pesquisa.

Avaliação do processo DIAOP-PRO	**Identificação de Instâncias Candidatas**	**Construção de um Classificador**		**Classificação de Instâncias**	
	Jurídico	**Jurídico**	**Turístico**	**Jurídico**	**Turístico**
Precisão	90%	87,30%	76,50%	90%	76,50%
Recall	89,50%	86,50%	77,50%	89,50%	77,50%
Medida-F	89,74%	86,80%	76,90%	89,74%	76,90%

Tabela 21: Avaliação do processo DIAOP-Pro

6. Conclusão

Esta tese analisou o problema e as principais abordagens para o povoamento de ontologias, destacando aspectos positivos e limitações, e propôs um processo genérico para o povoamento de ontologias que consiste de três fases: “Identificação de Instâncias Candidatas”, “Construção de um Classificador” e “Classificação de Instâncias”.

Uma das principais limitações identificadas nas abordagens analisadas para o povoamento de ontologias é a dependência de um domínio específico.

Por este motivo foi proposto um Processo Independente de Domínio para o Povoamento Automático de Ontologias que aplica técnicas de PLN e EI. Trata-se de uma nova abordagem que propõe o povoamento automático de ontologias utilizando uma ontologia para a geração automática de regras para extrair instâncias a partir de textos e classificá-as como instâncias de classes da ontologia. Estas regras podem ser geradas a partir de ontologias específicas em qualquer domínio tornando o processo independente de domínio.

A DIAOP-Tool - uma ferramenta para o Povoamento Automático de Ontologias - foi desenvolvida para automatizar o processo DIAOP-Pro proposto. A DIAOP-Tool tem como entradas o corpus em língua inglesa e a ontologia OWL; e como saída a ontologia OWL povoada. A ferramenta foi desenvolvida em linguagem Java e utiliza a biblioteca de classes do GATE, do Wordnet e do JENA.

Para avaliar o processo proposto quatros estudos de caso foram desenvolvidos. O primeiro estudo de caso descreveu uma análise comparativa entre a aplicação de técnicas estatísticas e de técnicas puramente lingüísticas na fase “Identificação de Instâncias Candidatas” a partir do corpus FamilyJuris. Considerando os resultados obtidos podemos concluir que as técnicas puramente lingüísticas apresentam melhores resultados que as técnicas estatísticas quando aplicadas em corpus de diferentes tamanhos. Isso ocorre porque as técnicas estatísticas necessitam de um corpus com um grande volume de documentos para

que apresente uma boa efetividade. Por este motivo as técnicas puramente lingüísticas foram escolhidas para serem aplicadas na primeira fase do processo proposto. Com este estudo de caso tivemos uma avaliação parcial do processo proposto.

O segundo estudo de caso apresentou uma análise da viabilidade da geração automática do classificador na fase “Construção de um Classificador”, além da sua aplicação nos corpora FamilyJuris e Turístico e nas ontologias FamilyLaw e OntoTur. Considerando os resultados obtidos podemos concluir que o classificador construído de forma manual apresenta melhores resultados do que o classificador gerado de forma automática, isso se deve ao fato de que o classificador construído de forma manual é desenvolvido por um especialista de domínio, que realiza manualmente uma tarefa custosa e dispendiosa, por isso é mais preciso. O classificador gerado de forma automática não depende de um especialista de domínio e automatiza esta tarefa sujeita a erros sem afetar a efetividade da classificação de instâncias. Então, a geração automática do classificador apresentou-se viável e foi aplicada no processo proposto.

O terceiro e o quarto estudos de caso descreveram o povoamento automático de ontologias em dois domínios distintos, o jurídico e o turístico. Considerando os resultados obtidos podemos concluir que o processo DIAOP-Pro proposto é efetivo e adequado para a realização do povoamento automático de ontologias específicas em um domínio.

No restante deste capítulo são discutidas as principais contribuições desta tese bem como possíveis trabalhos futuros a serem desenvolvidos.

6.1. Contribuições Científicas e Tecnológicas

O objetivo geral desta tese foi contribuir com soluções para a aquisição automática de conhecimento de forma a diminuir os custos e esforços no

povoamento de bases de conhecimento. Consideramos esse objetivo plenamente alcançado principalmente através das seguintes contribuições científicas:

1. Sistematização do problema do Povoamento de Ontologias.
2. Elaboração de técnicas híbridas para o Povoamento Automático de Ontologias.
3. Formalização de um Processo Independente de Domínio para o Povoamento Automático de Ontologias.
4. Desenvolvimento de uma ferramenta de software que provê suporte automatizado a aplicação do processo proposto.

O estado da arte do problema do povoamento de ontologias mostra apenas propostas que na sua maioria são definidas precariamente e sem nenhuma sistematização que permita uma adequada compreensão e aplicação dessas abordagens para solucionar o problema. Dessa forma, nossa contribuição ao estado da arte é a definição e sistematização do problema do Povoamento de Ontologias, através de um processo genérico que têm três fases: “Identificação de Instâncias Candidatas”, “Construção de um Classificador” e “Classificação de Instâncias” (Capítulo 3). Cada uma dessas fases ocorre através da aplicação de técnicas das principais áreas de conhecimento relacionadas ao Povoamento de Ontologias: Processamento da Linguagem Natural, Extração de Informação e/ou Aprendizagem de Máquina. As principais áreas de conhecimento também foram analisadas e apresentadas no Capítulo 2.

Um estudo comparativo das principais técnicas propostas para solucionar o problema do povoamento de ontologias (Capítulo 3) foi também apresentado motivando a elaboração de técnicas híbridas, ou seja, técnicas que utilizam as principais áreas de conhecimento de Processamento da Linguagem Natural, Extração de Informação e Aprendizagem de Máquina relacionada ao Povoamento de Ontologias.

A principal contribuição deste trabalho, o DIAOP-Pro (Capítulo 4) se constitui em uma nova abordagem uma vez que propõe o povoamento automático

de ontologias utilizando uma ontologia para a geração automática de regras para extrair instâncias a partir de textos e classifica-as como instâncias de classes da ontologia. Estas regras podem ser geradas a partir de ontologias específicas de qualquer domínio, tornando o DIAOP-Pro independente de domínio. Entretanto, quando a ontologia de entrada é definida, o DIAOP-Pro realiza o povoamento automático de ontologia em um domínio específico. O DIAOP-Pro foi avaliado (Capítulo 5) nos domínios do direito de família e turístico, apresentando resultados promissores e demonstrando a sua viabilidade em qualquer domínio de aplicação.

Foi desenvolvida a DIAOP-Tool - uma ferramenta para o Povoamento Automático de Ontologias – que provê suporte automatizado a aplicação do DIAOP-Pro. A DIAOP-Tool utiliza a API do GATE na aplicação da fase “Identificação de Instâncias Candidatas”; utiliza a API do Wordnet e a API do JENA na aplicação da fase “Construção de um Classificador”; e utiliza a API do JENA na aplicação da fase “Classificação de Instâncias”. A DIAOP-Tool recebe como entrada o corpus em língua inglesa e a ontologia OWL e como saída a ontologia OWL povoada. A DIAOP-Tool foi fundamental para a realização e avaliação dos experimentos.

Além das contribuições científicas, essa tese traz três contribuições tecnológicas:

1. Os corpora dos domínios do Direito de família e do turístico que serviram para os experimentos se configuram em um importante recurso lingüístico possível de serem utilizados em outros estudos de caso na pesquisa desenvolvida pelo grupo GESEC.
2. As ontologias FamilyLaw, do domínio do Direito de família e OntoTur, do domínio turístico povoadas automaticamente poderão ser utilizadas em sistemas baseado em conhecimento para auxiliar a tomada de decisão.
3. A ferramenta DIAOP-Tool que provê suporte automatizado ao processo DIAOP-Pro proposto.

Essas contribuições estarão disponíveis no grupo GESEC[6], que é o grupo no qual essa tese se insere. Essa tese também se insere no contexto do projeto Hermes, que desde janeiro 2010 está em execução e em parceria entre pesquisadores das universidades UFMA, UMinho, UFPE e UFAL e financiado pela CAPES e FCT. O HERMES busca viabilizar técnica e economicamente os sistemas baseados em conhecimento através do desenvolvimento de técnicas para a aprendizagem e povoamento automático de ontologias. Pretende-se ainda desenvolver e disponibilizar um ambiente de suporte a aprendizagem e povoamento de ontologias que integre todas as ferramentas de suporte às técnicas desenvolvidas no contexto do projeto.

Dentre os trabalhos concluídos no contexto desse projeto e diretamente relacionados com a proposta desta tese pode-se destacar:

- Foram elaboradas técnicas para a identificação de classes e instâncias candidatas para as áreas de aprendizagem e povoamento de ontologias respectivamente. São aplicadas técnicas estatísticas e técnicas de processamento da linguagem natural. Foi também desenvolvida a ferramenta NLP-Dumper que auxilia a aplicação das técnicas elaboradas [54].
- Foi proposto um processo para a aquisição de relações taxonômicas de ontologias na área de aprendizagem automática de ontologias a partir de fontes textuais. Este processo aplica técnicas de processamento de linguagem natural e uma base de dados léxica, Wordnet. A ferramenta desenvolvida é a Taxonomy NLP-Dumper [15];
- Foram elaboradas as técnicas PRECE e PREHE para a aprendizagem de conceitos (conjunto C) e hieraquias de conceitos (conjunto H) respectivamente. Ambas utilizam uma solução que consiste em uma aplicação inovadora das redes lógicas de

[6] http://gesec.deinf.ufma.br/

Markov. Também foi desenvolvida uma ferramenta que auxilia a aplicação das técnicas PRECE e PREHE [25];

Outros trabalhos estão ainda em desenvolvimento.

- Está sendo desenvolvido um framework e uma ferramenta para a aprendizagem de relacionamentos não taxonômicos de ontologias a partir de textos em língua inglesa que utiliza técnicas de PLN e estatísticas [72].

6.2.Limitações

Nesta seção são discutidas as limitações do processo DIAOP-Pro e da ferramenta DIAOP-Tool.

Em sua atual versão a DIAOP-Tool não aborda a construção do corpus e nem a especificação da ontologia, por entender que são tarefas que necessitam de uma inspeção visual e julgamento humano, sendo portanto delegada ao especialista de domínio a responsabilidade da construção de um corpus e a especificação de uma ontologia. Todas as técnicas do estado da arte analisadas semelhantemente a DIAOP-Tool também não abordam a construção do corpus e nem a especificação da ontologia.

A DIAOP-Tool, ferramenta de software que auxilia a aplicação do processo DIAOP-Pro, trabalha somente com textos em língua inglesa, pelo fato de termos boas ferramentas que auxiliam a aplicação de técnicas de PLN e EI, como o GATE [18], e uma base de dados léxica, como o WordNet [35] que possui uma API que facilita a busca pelos sinônimos propostos no processo. Todas as técnicas do estado da arte analisadas, semelhantemente a DIAOP-Tool, trabalham somente com textos em língua inglesa.

Uma análise para o desenvolvimento da ferramenta que utiliza como entrada o corpus em língua portuguesa foi realizada, porém a falta de ferramentas robustas e a falta de acessibilidade a uma base de dados léxica inviabilizou o seu

desenvolvimento. As ferramentas disponíveis que aplicam as técnicas de PLN para a língua portuguesa, como o NLTK [6] e o PALAVRAS [5] realizam somente a análise morfo-lexical proposta no processo, entretanto não realizam o reconhecimento de entidades nomeadas e nem as co-referências propostas no processo DIAOP-Pro. Por outro lado, a base de dados léxica disponível para a língua portuguesa é o WordNet Br [22], que não possui uma API que facilite as buscas pelos sinônimos propostos no processo DIAOP-Pro.

6.3. Trabalhos Futuros

A partir desta tese outros trabalhos estão sendo e serão desenvolvidos para dar continuidade a esta pesquisa, que se manifestam em dois eixos de pesquisa, um experimental e outro de desenvolvimento.

- Integrar a ferramenta DIAOP-Tool desenvolvida ao ambiente de desenvolvimento para a Aprendizagem e Povoamento de Ontologias – APONTO – que está em desenvolvimento pelo grupo GESEC.
- Adaptar a ferramenta DIAOP-Tool como plugin do GATE para ser disponibilizado e utilizado em domínio público.
- Efetuar uma extensão na ferramenta DIAOP-Tool para processar uma língua diferente da língua inglesa, como a portuguesa. Esse tipo de experiência permitirá a verificação prática de que o processo proposto pode ser empregado em qualquer língua alvo desejada. Este trabalho futuro é de grande relevância prática, por aumentar o escopo de aplicação do processo DIAOP-Pro proposto e da ferramenta DIAOP-Tool. A ferramenta NLTK [6] que provê suporte a aplicação de técnicas de PLN poderia ser utilizada e a base de dados léxica WordNet.Br [22] poderia também ser utilizada para a consulta dos sinônimos.
- Realizar experimentos com o mesmo corpus e ontologia

comparando os resultados apresentados pela DIAOP-Tool com uma abordagem do estado da arte. Esses novos experimentos são importantes para aumentar a confiabilidade do processo DIAOP-Pro.

- Avaliar as ontologias povoadas automaticamente na execução de sistemas baseados em conhecimento para o acesso à informação e suporte às decisões nos domínios jurídico e turístico. As ontologias povoadas automaticamente são úteis na medida em que melhoram a efetividade dos sistemas nos quais elas são empregadas. Assim, a avaliação da ontologia povoada em uma aplicação executável visa medir a efetividade de um sistema que utiliza as ontologias que estão sendo avaliadas.
- Avaliar as ontologias povoadas automaticamente, sendo utilizadas como entrada para a aplicação de uma técnica de extração de axiomas (conjunto A da definição de ontologia apresentada no capítulo 2) que utiliza a programação em lógica indutiva, que está em desenvolvimento pelo grupo GESEC [52].
- Realizar novos experimentos com um outro especialista de domínio para comparar com os resultados obtidos no segundo estudo de caso.

6.4. Publicações

- Faria, C., Girardi, R., Serra, I., Macedo, M., Maranhão, D. Using Natural Language Processing for Automatic Extraction of Ontology Instances. Proceedings of 12th International Conference on Enterprise Information Systems (ICEIS 2010), Ed. INSTIIC, pp. 278 a 283, Funchal, Madeira – Portugal. 8-12 June of 2010. Qualis Capes B1.

Nesta publicação foi avaliada uma técnica que aplica técnicas estatísticas da área de Recuperação de Informação na fase “Identificação de Instâncias Candidatas” do processo DIAOP-Pro.

- Faria, C., Girardi, R. Um Processo Semi- Automático para o Povoamento Automático de Ontologias a partir de Fontes Textuais, iSys - Revista Brasileira de Sistemas de Informação - PPGI / UNIRIO, vol. 3, 2010. Qualis Capes B3 Ciência da Computação e B5 em Engenharias IV

Nesta publicação foi avaliado um processo que aplica técnicas de PLN na fase de “Identificação de Instâncias Candidatas” e que aplica técnicas de EI na fase “Construção de um Classificador” gerando o classificador de forma manual do processo DIAOP-Pro.

- Faria C., Girardi, R. An Information Extraction Process for Semi-Automatic Ontology Population, In: International Conference on Soft Computing Models in Industrial and Environmental Applications - SOCO 2011, Springer: E. Corchado Eds. Vol 87, pp. 319-328, Salamanca – Spain. 06-08 April of 2011. Qualis Capes B4.

Nesta publicação foi avaliado um processo que aplica técnicas de PLN na fase de “Identificação de Instâncias Candidatas” e que aplica técnicas de EI na fase “Construção de um Classificador” gerando o classificador de forma manual do processo DIAOP-Pro.

- Faria, C., Girardi, R. and Novais P. Using Domain Specific Generated Rules for Automatic Ontology Population. In: The International Conference on Intelligent Systems Design and Applications (ISDA 2012). pp. 297-302, Koshi – India, 27-29 November of 2012. Qualis Capes B2.

Nesta publicação foi avaliado o processo DIAOP-Pro proposto nesta tese (Capítulo 4) que aplica técnicas de PLN na fase de “Identificação de Instâncias Candidatas” e aplica técnicas de EI na fase “Construção de um Classificador” gerando o classificador de forma automática e independente de domínio.

- Faria C., Girardi, R., Novais P. Analysing the Problem and Main Approaches for Ontology Population, In: 10th International Conferenceon Information Technology: New Generations (ITNG 2013). Qualis Capes B1.

Nesta publicação foi avaliado o processo genérico para o problema do Povoamento Automático de Ontologias proposto nesta tese (Capítulo 3).

- Faria C. and Girardi R. A Domain-Independent Process for Automatic Ontology Population, artigo submetido no dia 31.01.2013 para o Special Issue on Systems development by means of semantic technologies in Science of Computer Programming journal. Qualis Capes B1.

Nesta publicação é avaliado o processo genérico para o problema do Povoamento Automático de Ontologias proposto nesta tese (Capítulo 3), o processo DIAOP-Pro proposto nesta tese (Capítulo 4) que aplica técnicas de PLN na fase de “Identificação de Instâncias Candidatas” e aplica técnicas de EI na fase “Construção de um Classificador” gerando o classificador de forma automática e independente de domínio e a avaliação (Capítulo 5) do DIAOP-Pro proposto nesta tese.

- Faria, C., Girardi, R. A Domain-Independent Approach for Ontology Instantiation from Documents in Natural Language, artigo submetido no dia 02.09.2012 para a iSys - Revista Brasileira de Sistemas de Informação - PPGI / UNIRIO. Qualis Capes B3 Ciência da Computação e B5 em Engenharias IV

Nesta publicação é avaliado o processo proposto nesta tese (Capítulo 4) que aplica técnicas de PLN na fase de “Identificação de Instâncias Candidatas” e aplica técnicas de EI na fase “Construção de um Classificador” gerando o classificador de forma automática e independente de domínio.

Referências

[1] Aitken, J.S. Learning information extraction rules: An inductive logic programming approach. In Proceedings of the 15th European Conference on Artificial Intelligence (ECAI'02), IOS Press. Amsterdam, 2002.

[2] Alexiev, V., Breu, M., de Bruijn, J., Fensel, D., Lara, R. and Lausen, H., "Information Integration with Ontologies: Experiences from an Industrial Showcase", Wiley, 2005.

[3] Allen J. "Natural Language Understanding", Redwood City, CA: The Benjamin/Cummings Publishing Company, Inc, 1995.

[4] Baeza-Yates, R. and Ribeiro-Neto, B. "Modern Information Retrieval". New York: ACM Press, 1999.

[5] Bick, Eckhard. The parsing system PALAVRAS: automatic grammatical analysis of portuguese in constraint grammar framework, PhD thesis, Arhus University, Arhus, Danemark, 2000.

[6] Bird, S., Klein, E., Loper, E. Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit. Editora O'Really Media, 2009.

[7] Bishop, C. M. Pattern Recognition and Machine Learning, Springer, 2006.

[8] Buitelaar, P., Cimiano, P. and Magnini, P., Ontology Learning from Text: Methods, Evaluation and Applications, IOS Press, Amsterdam, The Netherlands, 2006.

[9] Buyko E., Wermter J., Poprat M. and Hahn U. Automatically mapping an NLP core engine to the biology domain. In Proceedings of the ISMB 2006 joint BioLINK/Bio-Ontologies meeting, 2006.

[10] Ceh, I., Crepinsek, M., Kosar, T. and Mernik, M. Ontology Driven Development of Domain-Specific Languages. Computer Science and Information Systems 8(2): p. 317-342, 2011.

[11] Central Juridica. Noções Gerais de Direito de Família. Disponível em: http://www.centraljuridica.com/doutrina/120/direito_civil/nocoes_gerais_de_direito_de_familia.html . Acesso em Novembro de 2009.

[12] Cimiano, P. Ontology Learning and Population from Text: Algorithms, Evaluation and Applications. Springer-Verlag New York, Inc. Secaucus, NJ, USA, 2006.

[13] Cimiano, P. and Volker, J., Towards large-scale, open-domain and ontology-based named entity classification. In: Proceedings of RANLP'05, p. 166–172, Borovets, Bulgaria, 2005.

[14] Cimiano, P., Ladwig, G., and Staab, S. Gimme 'the context: Context-driven automatic semantic annotation with C-PANKOW. In Proceedings of the 14th World Wide Web Conference (WWW), p. 332-341, 2005.

[15] Correia, J. Um Processo para a Aquisição de Relações Taxonômicas de uma Ontologia. Dissertação – Curso de Pós-Graduação em Engenharia de Eletricidade na área de Ciência da Computação, Universidade Federal do Maranhão, 2011.

[16] Cowie J. and Wilks Y. Information Extraction, Handbook of Natural Language Processing, Robert Dale, Hermann Moisl and Harold Somers, p. 241–260, 2000.

[17] Cowie, J. and Lenhnert, W. Information Extraction. Communications of the ACM 39(1), p. 80-91. 1996.

[18] Cunningham H. Developing Language Processing Components with GATE - Version 5 (a User Guide). University of Shelfield, 2009.

[19] Cunningham H. Information Extraction, Encyclopedia of Language and Linguistics, 2nd Edition, 2005.

[20] Dale R., Moisl H. and Somers H. L. Handbook of natural language processing, CRC, 2000.

[21] Dellschaft K. and Staab S. On how to perform a gold standard based evaluation of ontology learning, In: Proceedings of the 5th International Semantic Web Conference, p. 228 – 241, Athens. Springer, 2006.

[22] Dias-da-Silva, B.C., Rocha, M.A.E, Nunes, M.G.V. Projeto Montagem da Base Wordnet para o Português do Brasil-Processo CNPq 552057/01-0. Relatório Técnico. Araraquara: FCL-Unesp, 52p, 2004.

[23] Diniz, M. H. Curso de Direito Civil Brasileiro – Vol. 5 – Direito de Família. 27ª edição. Editora Saraiva, 2012.

[24] Drake, Miriam A. Encyclopedia of Library and Information Science: Lib-Pub. CRC Press, 2003.

[25] Drumond, L. Aquisição Automatizada de Hierarquias de Conceitos de Ontologias Utilizando Aprendizagem Estatística Relacional. Dissertação – Curso de Pós-Graduação em Engenharia de Eletricidade na área de Ciência da Computação, Universidade Federal do Maranhão, 2009.

[26] Embley, D. W. Toward semantic understanding: An approach based on information extraction ontologies. In Proceedings of the Fifteenth Conference on Australasian Database, p. 3–12. Australian Computer Society, 2004.

[27] Embley, D.W., Kurtz, B.D. and Woodfiel, S.N. Object-oriented systems analysis: a model-driven approac. Prentice-Hall, 1992.

[28] Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A. M., Shaked, T., Soderland, S., Weld, D., and Yates, A. Web-scale information extraction in KnowItAU (preliminary results). In Proceedings of the 13th World Wide Web Conference (WWW), p. 100-109, 2004.

[29] Etzioni, O., Cafarella, M., Downey, D., Popescu, A. M., Shaked, T., Soderland, S., Weld, D., and Yates, A. Methods for domain-independent information extraction from the web: An experimental comparison. In Proceedings of the 19th National Conference on Artificial Intelligence (AAAI), p. 391-398, 2004.

[30] Etzioni, O., Cafarella, M., Downey, D., Popescu, A. M., Shaked, T., Soderland, S., Weld, D., and Yates, A. Unsupervised named-entity extraction from the web: An experimental study. Artificial Intelligence, 165(1):91-134, 2005.

[31] Evans, R. A framework for named entity recognition in the open domain. In Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP), p. 137-144, 2003.

[32] Faria, C., Girardi, R. Um Processo Semi- Automático para o Povoamento Automático de Ontologias a partir de Fontes Textuais, iSys - Revista Brasileira de Sistemas de Informação - PPGI / UNIRIO, vol. 3, 2010.

[33] Faria C., Girardi, R. An Information Extraction Process for Semi-Automatic Ontology Population, In: International Conference on Soft Computing Models in Industrial and Environmental Applications - SOCO 2011, Salamanca : Springer, 2011.

[34] Faria, C., Girardi, R., Serra, I., Macedo, M., Maranhão, D. Using Natural Language Processing for Automatic Extraction of Ontology Instances. Proceedings of the 12th International Conference on Enterprise Information Systems (ICEIS 2010), Ed. INSTIIC, p. 278 a 283, Funchal, Madeira – Portugal. 8-12 June of 2010.

[35] Fellbaum, C. Wordnet: An Electronic Lexical Database. Cambridge: MIT Press, 1998.

[36] Findlaw, http://www.findlaw.com/casecode/index.html. Acesso em Novembro de 2010.

[37] Finin, T., Fritzson, R., McKay, D. and McEntire, R. KQML as an Agent Communication Language. In: Proceedings of the 3rd International Conference on Information and Knowledge management (CIKM'94), p. 456-463, 1994.

[38] Fleischman, M. and Hovy, E., Fine Grained Classification of Named Entities. In: Proceedings of COLING, Taipei, Taiwan, August, 2002.

[39] Freitag, D. Information extraction from HTML: Application of a general machine learning approach. In Proceedings of the 15th Conference on Artificial Intelligence (AAAI'98). p. 517-523, 1998.

[40] Gaizauskas, R. and Wilks, Y. Information Extraction: Beyond Document Retrieval. Journal of Documentation 54(1), p. 70 - 105, 1998.

[41] Girardi, R. Guiding Ontology Learning and Population by Knowledge System Goals, In: Proceedings of International Conference on Knowledge Engineering and Ontology Development, Ed. INSTIIC, Valence, October, 2010.

[42] Girardi, R. Classification and Retrieval of Software through their Descriptions in Natural Language, Ed. Imprimerie de l´Université de Geneve. 212 pgs, Geneva, Switzerland, 1995.

[43] Girardi, R. Using English to Retrieve Software, The Journal of Systems and Software, v. 30, n. 3, p. 249-270, 1995.

[44] Gonzalez, M. e Lima, V. L. S. Recuperação de Informação e Processamento de Linguagem Natural. XXIII Congresso da Sociedade Brasileira de Computação, Campinas. Anais do III Jornada de Mini-Cursos de Inteligência Artificial, Volume III, p.347-395, 2003.

[45] Gruber, T. R. Toward Principles for the Design of Ontologies used for Knowledge Sharing. International Journal of Human-Computer Studies, nº43, p. 907-928, 1995.

[46] Guarino N., Masolo C. and Vetere C. Ontoseek: Content-based Access to the web, IEEE Intelligent Systems, v. 14(3), p. 70-80, 1999.

[47] Harris, Z. Distributional structure. In J.J. Katz, editor, The Philosophy of Linguistics, pages 26-47. New York: Oxford University Press, 1995.

[48] Haykin, S. Redes Neurais: Princípios e Prática. Porto Alegre: Bookman, 2001.

[49] Hearst, M., Automated Discovery of Word-Net Relations. In WordNet: An Electronic Lexical Database. MIT Press, 1998.

[50] Jielin, D. Network Dictionary. Javvin Technologies, 2007.

[51] Karkaletsis V., Valarakos A. and Spyropoulos C. D. Populating ontologies in biomedicine and presenting their content using multilingual generation, Acquiring and Representing Multilingual, Specialized Lexicons: the Case of Biomedicine, Italy, Genoa, 2006.

[52] Leite N. Aquisição Automática de Axiomas de Ontologias utilizando Programação em Lógica Indutiva. Relatório de Iniciação Científica, Universidade Federal do Maranhão, 2012.

[53] Lewis, D. Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval. Proceedings of ECML-98, 10th European Conference on Machine Learning. Chemnitz, DE: Springer Verlag, Heidelberg, DE. p. 4–15, 1998.

[54] Macedo, M. J. C. Processamento da Linguagem Natural para a Identificação de Classes e Instâncias de uma Ontologia. Monografia – Curso de Graduação em Ciência da Computação, Universidade Federal do Maranhão, 2010.

[55] Marcus, M., Santorini, B. and Marcinkiewicz, M. Building a Large Annotated Corpus of English: The Penn Treebank. Computational Linguistics: Special Issue on Using Large Corpora, [S. l.], v. 19, n. 2, p. 313-330, 1993.

[56] Marneffe, M. C. and Manning, C. D. The Stanford Typed Dependencies Representation. In Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation (Coling 2008), p. 1-8, 2008.

[57] Maynard, D., Tablan V., Ursu C., Cunhigan H. and Wilks Y. Named Entity Recognition from Diverse Text Types. In Recent Advances in Natural language Processing 2001 Conference, Tzigov Chark, Bulgaria, 2001.

[58] McDowell, L., Cafarella, M. Ontology-driven information extraction with OntoSyphon. In: International Semantic Web Conference. p. 428–444, 2006.

[59] Mitchell, T. Machine Learning, McGraw Hill, 1997.

[60] Monard, M. C. e Baranauskas, J. A. Indução de Regras e Árvores de Decisão. In: REZENDE, Solange Oliveira (coord.). Sistemas Inteligentes. Barueri: Manole, 2005.

[61] Monard, M. C. e Baranauskas, J. A. Conceitos sobre Aprendizado de Máquina. In: REZENDE, Solange Oliveira (coord.). Sistemas Inteligentes. Barueri: Manole, 2005.

[62] Mosqueira A. et al. JavaBeans. Faculdade de Computação e Informática. Universidade Presbiteriana Mackenzie, 2006.

[63] Muslea, I. Extraction Patterns for Information Extraction Tasks: A Survey. In Proceedings of the AAAI-99 Workshop on Machine Learning for Information Extraction (Orlando, FL), p. 1-6, 1999.

[64] Mitkov R., Orasan C. and Evans R. The importance of annotated corpora for NLP: the cases of anaphora resolution and clause splitting. In Proceedings of Corpora and NLP: Reflecting on Methodology Workshop, TALN'99, 1999.

[65] Neches, R., Fikes, R., Finin, T., Gruber, T., Patil, R., Senator, T. and Swartout, W. R. Enabling technology for knowledge sharing. AI Magazine, 12(3):16--36, 1991.

[66] Nierenburg S. and Raskin V. Ontological Semantics, MIT Press, 2004.

[67] OWL, http://www.w3.org/2001/sw/WebOnt/. Acesso em Novembro de 2010

[68] Porzel, R. and Malaka, R. A Task-based Approach for Ontology Evaluation. In ECAI Workshop on Ontology Learning and Population, Valencia, Spain, 2004.

[69] Ruiz-Martínez, J. M., Miñarro-Giménez, J. A., Guillén-Cárceles, L., Castellanos-Nieves, D., Valencia-García, R., García-Sánchez, F., Fernández-Breis, J. T. and Martínez-Béjar, R. Populating Ontologies in the eTourism Domain. In Proceedings of the 2008 IEEE/WIC/ACM international Conference on Web intelligence and intelligent Agent Technology - Volume 03. Web Intelligence & Intelligent Agent. IEEE Computer Society, Washington, DC, p. 316-319, December 09 - 12, 2008.

[70] Russel, S. e Norvig, P. Inteligência artificial. Rio de Janeiro: Editora Campus, 2004.

[71] Salton, G. and Buckley, C. Term Weighting Approaches in Automatic Text Retrieval. Cornell University, 1987.

[72] Serra, I. Extração automatizada de relacionamentos não taxonômicos de ontologias a partir de fontes textuais. Relatório Técnico do Exame de

Qualificação – Curso de Pós-Graduação em Engenharia de Eletricidade na área de Ciência da Computação, Universidade Federal do Maranhão, 2011.

[73] Sirin, E., Hendler, J. and Parsia, B. Semi-automatic composition of web services using semantic descriptions. In: Proceedings of the ICEIS Workshop on Web Services: Modeling, Architecture and Infrastructure, 2002.

[74] Steven, B., Ewan, K., Edward L. Natural Language Processing with Pyton. O'Reilly Media, 2009.

[75] Tan, P. N., Steinbach, M. and Kumar, V. Introduction to Data Mining, Pearson Addison Wesley, 2005

[76] Tanev, H. and Magnini, B. Weakly Supervised Approaches for Ontology Population. In: Proceedings of of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL), p. 17-24, 2006.

[77] Uschold, M. and Grüninger, M. Ontologies: Principles, Methods and Applications. Knowledge Engineering Review. Vol. 11, Nº 02. June, 1996.

[78] Vieira, R. e Lima, V. L. S. Lingüística Computacional: princípios e aplicações. Anais do Congresso da Sociedade Brasileira de Computação, v. 2, p. 47-88, Fortaleza: SBC, 2001.

[79] Waibel, A. Modular Construction of time-delay neural networks for speech recognition. Neural Computation, 1:39 46, 1989.

[80] Welty, C. A. and Ide, N. Using the right tolls: enhancing retrieval from marked-up documents, Computers and Humanities, v. 33(10), p. 59-84, 1999.

[81] Wimalasuriya, D. C. and Dou, D. Ontology-based information extraction: An introduction and a survey of current approaches. In Proceedings of Journal Information Science, p. 306-323, 2010.

[82] Yildiz, Burcu and Miksch, Silvia. ontoX - a method for ontology-driven information extraction. In Proceedings of the 2007 international conference on Computational science and its applications - Volume Part III (ICCSA'07),

Osvaldo Gervasi and Marina L. Gavrilova (Eds.), Vol. Part III. Springer-Verlag, Berlin, Heidelberg, p. 660-673, 2007.

Anexo A – Conjunto de marcação Penn TreeBank

Marcação	Significado
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
FW	Foreign Word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PP	Personal pronoun
PPS	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	To
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle

VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun
WP$	Possessive wh-pronoun
WRB	Wh-adverb

Anexo B – Dependências de Stanford

Tipo da Dependência	*Significado*
abbrev	*abbreviation modier*
acomp	*adjectival complement*
advcl	*adverbial clause modier*
advmod	*adverbial modier*
agent	*Agent*
amod	*adjectival modier*
appos	*appositional modier*
arg	*Argument*
attr	*Attributive*
aux	*Auxiliary*
auxpass	*passive auxiliary*
cc	*Coordination*
ccomp	*clausal complement with internal subject*
comp	*Complement*
compl	*Complementizer*
conj	*Conjunct*
cop	*Copula*
csubj	*clausal subject*
csubjpass	*passive clausal subject*
dep	*Dependent*
det	*Determiner*
dobj	*direct object*
expl	*expletive (expletive \there")*
infmod	*innitival modier*
iobj	*indirect object*
mark	*marker (word introducing an advcl)*
mod	*Modier*
neg	*negation modier*
nsubj	*nominal subject*
nsubjpass	*passive nominal subject*
obj	*Object*
partmod	*participial modier*

pobj	*object of preposition*
preconj	*Preconjunct*
predet	*Predeterminer*
purpcl	*purpose clause modier*
rcmod	*relative clause modier*
rel	*relative (word introducing a rcmod)*
subj	*Subject*
xcomp	*clausal complement with external subject*

Apêndice A – Regras desenvolvidas pelo especialista de domínio para o relacionamento não taxonômico “wife” da classe “Marriage” da ontologia FamilyLaw

```
Phase: InstanceWife
Input: Token Lookup SpaceToken
Options: control = appelt

Rule: Wife1
Priority: 50
(
      ({Lookup.minorType == female})
      {SpaceToken}
      {Token.category == NNP}
      {SpaceToken}
      {Token.string == "married"}

)
:Wife
-->
:Wife.Wife = { rule = "Wife1"}

Rule: Wife2
Priority: 50
(
      ({Lookup.minorType == female})
      {SpaceToken}
      {Token.string == "were"}
      {SpaceToken}
      {Token.string == "married"}

)
:Wife
-->
:Wife.Wife = { rule = "Wife2"}
```

```
Rule: Wife3
Priority: 50
(
  {Token.category == NNP}
  ({Token.string == "."}
  {SpaceToken}
  ({Lookup.minorType == female})) +
        ({Token.string == "."})
  ({SpaceToken})
  {Token.string == "were"}
  {SpaceToken}
        {Token.string == "married"}

)
:Wife
-->
:Wife.Wife = { rule = "Wife3"}

Rule: Wife4
Priority: 50
(
  {Token.category == NNP}
  ({Token.string == "."}
  {SpaceToken}
  ({Lookup.minorType == female})) +
        ({Token.string == "."})
  ({SpaceToken})
  {Token.string == "married"}

)
:Wife
-->
:Wife.Wife = { rule = "Wife4"}

Rule: Wife5
Priority: 50
```

```
(
      {Token.category == NNP}
      {SpaceToken}
      {Token.string == "and"}
      {SpaceToken}
      ({Lookup.minorType == female})
      {SpaceToken}
      {Token.string == "have"}
      {SpaceToken}
      {Token.string == "been"}
      {SpaceToken}
      {Token.string == "married"}

)
:Wife
-->
:Wife.Wife = { rule = "Wife5"}

Rule: Wife6
Priority: 50
(
    ({Lookup.minorType == female})
    {SpaceToken}
    {Token.category == NNP}
    {SpaceToken}
    {Token.string == "were"}
    {SpaceToken}
    {Token.string == "married"}

)
:Wife
-->
:Wife.Wife = { rule = "Wife6"}

Rule: Wife7
Priority: 50
```

```
(
    ({Lookup.minorType == female})
    {Token.string == "."}
    {Token.category == NNP}
    {Token.string == "."}
    {SpaceToken}
    {Token.string == "were"}
    {SpaceToken}
    {Token.string == "married"}

)
:Wife
-->
:Wife.Wife = { rule = "Wife7"}

Rule: Wife8
Priority: 50
(
      ({Lookup.minorType == female})
      {SpaceToken}
      {Token.category == NNP}
      {SpaceToken}
      {Token.string == "("}
      {SpaceToken}
      {Token.category == NNP}
      {SpaceToken}
      {Token.string == ")"}
      {SpaceToken}
      {Token.string == "were"}
      {SpaceToken}
      {Token.string == "married"}

)
:Wife
-->
:Wife.Wife = { rule = "Wife8"}
```

```
Rule: Wife9
Priority: 50
(
      ({Lookup.minorType == female})
      {SpaceToken}
      {Token.category == NNP}
      {Token.string == "."}
      {SpaceToken}
      {Token.category == NNP}
      {SpaceToken}
      {Token.string == "were"}
      {SpaceToken}
      {Token.string == "married"}

)
:Wife
-->
:Wife.Wife = { rule = "Wife10"}

Rule: Wife10
Priority: 50
(
      {Token.category == NNP}
      {SpaceToken}
      {Token.string == "married"}
      {SpaceToken}
      ({Lookup.minorType == female})

)
:Wife
-->
:Wife.Wife = { rule = "Wife10"}

Rule: Wife11
Priority: 50
(
```

```
      {Token.category == NNP}
      {SpaceToken}
      {Token.string == "was"}
      {SpaceToken}
      {Token.string == "married"}
      {SpaceToken}
      {Token.string == "to"}
      {SpaceToken}
      ({Lookup.minorType == female})

)
:Wife
-->
:Wife.Wife = { rule = "Wife12"}

Rule: Wife13
Priority: 50
(
      {Token.category == NNP}
      {SpaceToken}
      {Token.category == NNP}
      {Token.string == "."}
      {SpaceToken}
      {Token.string == "was"}
      {SpaceToken}
      {Token.string == "married"}
      {SpaceToken}
      {Token.string == "to"}
      {SpaceToken}
      ({Lookup.minorType == female})
      {SpaceToken}
      {Token.category == NNP}
      {Token.string == "."}

)
:Wife
```

```
-->
:Wife.Wife = { rule = "Wife13"}
```

Apêndice B – Regras geradas automaticamente para o relacionamento não taxonômico “wife” da classe “Marriage” da ontologia FamilyLaw

```
Phase: Marriage_wife
Input: Token Loookup SpaceToken
Options: control = appelt

Rule: Marriage_wife0
Priority: 50
(
  {Token.string == "wife" }
):Marriage_wife
-->
:Marriage_wife.Marriage_wife = { rule = "Marriage_wife0"
,findCategory="NNP",InterText="True", RuleType="wife",
owlPropName="wife_member", owlClassName="Marriage", owlRanger="#Person" }

Rule: Marriage_wife1
Priority: 50
(
  {Token.string == "Wife" }
):Marriage_wife
-->
:Marriage_wife.Marriage_wife = { rule = "Marriage_wife1"
,findCategory="NNP",InterText="True", RuleType="Wife",
owlPropName="wife_member", owlClassName="Marriage", owlRanger="#Person" }

Rule: Marriage_wife2
Priority: 50
(
  {Token.string == "married" }
  {SpaceToken}
  {Token.string == "woman" }
):Marriage_wife
-->
```

```
          :Marriage_wife.Marriage_wife = { rule = "Marriage_wife2"
,findCategory="NNP",InterText="True", RuleType="married woman",
owlPropName="wife_member", owlClassName="Marriage", owlRanger="#Person" }


          Rule: Marriage_wife3
          Priority: 50
          (
            {Token.string == "Married" }
            {SpaceToken}
            {Token.string == "woMan" }
          ):Marriage_wife
          -->
          :Marriage_wife.Marriage_wife = { rule = "Marriage_wife3"
,findCategory="NNP",InterText="True", RuleType="Married woMan",
owlPropName="wife_member", owlClassName="Marriage", owlRanger="#Person" }
```

Apêndice C – Regras desenvolvidas pelo especialista de domínio para a propriedade “name” da classe “Hotel” da ontologia OntoTur

```
Phase:  InstanceHotel
Input:  Token Lookup SpaceToken
Options: control = appelt

Macro: NP
(
{Token.category == NN} | {Token.category == NNS} | {Token.category == NNP} | {Token.category == NPS} | {Lookup.majorType == person}
)

Rule: InstanceHotel1
Priority: 50
(
  {Token.category == NNP}
  ({SpaceToken})
  {Token.string == ","}
  ({SpaceToken})
  {Token.string =~ "[Hh]otel"}

):InstanceHotel
-->
:InstanceHotel.InstanceHotel = {rule = "InstanceHotel1"}

Rule: InstanceHotel2
Priority: 50
(
  {Token.category == NNP}
  {SpaceToken}
  {Token.string =~ "[Hh]otel"}

):InstanceHotel
-->
```

```
:InstanceHotel.InstanceHotel = {rule = "InstanceHotel2"}

Rule: InstanceHotel3
Priority: 50
(
  {Token.category == NNP}
  {SpaceToken}
  {Token.category == NNP}
  {SpaceToken}
  {Token.string =~ "[Hh]otel"}

):InstanceHotel
-->
:InstanceHotel.InstanceHotel = {rule = "InstanceHotel3"}

Rule: InstanceHotel4
Priority: 50
(
  {Token.category == NNP}
  {SpaceToken}
  {Token.category == NNP}
  {SpaceToken}
  {Token.category == NNP}
  {SpaceToken}
  {Token.string =~ "[Hh]otel"}

):InstanceHotel
-->
:InstanceHotel.InstanceHotel = {rule = "InstanceHotel4"}

Rule: InstanceHotel5
Priority: 50
(
  {Token.string =~ "[Hh]otel"}
  ({SpaceToken})
  {Token.string == ","}
```

```
  ({SpaceToken})
  {Token.category == NNP}

):InstanceHotel
-->
:InstanceHotel.InstanceHotel = {rule = "InstanceHotel5"}

Rule: InstanceHotel6
Priority: 50
(
  {Token.string =~ "[Hh]otel"}
  {SpaceToken}
  {Token.category == NNP}

):InstanceHotel
-->
:InstanceHotel.InstanceHotel = {rule = "InstanceHotel6"}

Rule: InstanceHotel7
Priority: 50
(
  {Token.string =~ "[Hh]otel"}
  {SpaceToken}
  {Token.category == NNP}
  {SpaceToken}
  {Token.category == NNP}

):InstanceHotel
-->
:InstanceHotel.InstanceHotel = {rule = "InstanceHotel7"}

Rule: InstanceHotel8
Priority: 50
(
  {Token.string =~ "[Hh]otel"}
  {SpaceToken}
```

```
  {Token.category == NNP}
  {SpaceToken}
  {Token.category == NNP}
  {SpaceToken}
  {Token.category == NNP}

):InstanceHotel
-->
:InstanceHotel.InstanceHotel = {rule = "InstanceHotel8"}
```

Apêndice D – Regras geradas automaticamente para a propriedade "name" da classe "Hotel" da ontologia OntoTur

```
Phase: Hotel_Hotel

Input: Token Loookup SpaceToken

Options: control = appelt


Rule: Hotel_Hotel0

Priority: 50

(

  {Token.string == "hotel" }

):Hotel_Hotel

-->

:Hotel_Hotel.Hotel_Hotel    =    {    rule    =    "Hotel_Hotel0"
,findCategory="NNP",InterText="True",                    RuleType="hotel",
owlPropName="Name",                                  owlClassName="Hotel",
owlRanger="http://www.w3.org/2001/XMLSchema#string" }


Rule: Hotel_Hotel1

Priority: 50

(

  {Token.string == "Hotel" }

):Hotel_Hotel

-->

:Hotel_Hotel.Hotel_Hotel    =    {    rule    =    "Hotel_Hotel1"
,findCategory="NNP",InterText="True",                    RuleType="Hotel",
owlPropName="Name",                                  owlClassName="Hotel",
owlRanger="http://www.w3.org/2001/XMLSchema#string" }
```