PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

Universidade Federal do Rio de Janeiro

DISSERTAÇÃO DE MESTRADO

RESOLUÇÃO DE ENTIDADES NOMEADAS UTILIZANDO RECURSOS EM LINKED DATA

Bianca de Oliveira Pereira

Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais

Rio de Janeiro
2012

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE MATEMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

BIANCA DE OLIVEIRA PEREIRA

RESOLUÇÃO DE ENTIDADES NOMEADAS UTILIZANDO RECURSOS EM LINKED DATA

Dissertação de Mestrado submetida ao Corpo Docente do Departamento de Ciência da Computação do Instituto de Matemática, e Instituto Tércio Pacitti (iNCE) da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários para a obtenção do título de Mestre em Informática.

Orientadores: Profa. Dra. Adriana Santarosa Vivacqua

Prof. Dr. João Carlos Pereira da Silva

RIO DE JANEIRO

2012

P Pereira, Bianca de Oliveira.
Resolução de Entidades Nomeadas utilizando recursos em Linked Data. / Bianca de Oliveira Pereira. -- 2012.
224 f.: il.

Dissertação (Mestrado em Informática) – Universidade Federal do Rio de Janeiro, Instituto de Matemática, Núcleo de Computação Eletrônica, 2012.

Orientadora: Adriana Santarosa Vivacqua
Co-orientador: João Carlos Pereira da Silva
1. Resolução de Entidades Nomeadas. 2. Entidades Nomeadas. 3.Linked Data. 4. Processamento de Linguagem Natural – Teses. I. Vivacqua, Adriana Santarosa (Orient.). II. da Silva, João Carlos Pereira. (Co-orient.). III Universidade Federal do Rio de Janeiro, Instituto de Matemática, InstitutoTércio Pacitti. III Título.

CDD

Resolução de Entidades Nomeadas utilizando Recursos em Linked Data

Bianca de Oliveira Pereira

Dissertação de Mestrado submetida ao Corpo Docente do Departamento de Ciência da Computação do Instituto de Matemática, e Núcleo de Computação Eletrônica da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários para obtenção do título de Mestre em Informática.

Aprovado por:

Profa. Adriana Santarosa Vivacqua, D.Sc., UFRJ (Orientador)

Prof. João Carlos Pereira da Silva, D.Sc., UFRJ (Co-orientador)

Profa. Maria Luiza Machado Campos, Ph.D., UFRJ

Profa. Valeria Menezes Bastos, D.Sc., UFRJ

Profa. Vanessa Braganholo Murta, D.Sc., UFF

Rio de Janeiro, Dezembro de 2012

Dedico este trabalho a todos aqueles que um dia duvidaram do impossível e tiveram a coragem de sonhar.

AGRADECIMENTOS

Gostaria de agradecer primeiramente aos meus orientadores que realizaram esta jornada comigo. À Adriana pela motivação no momento em que as ideias fervilhavam e a organização ainda era pouca e ao João Carlos pelo suporte e auxílio em cada mínimo detalhe. Agradeço também aos meus amigos Fabrício e Maria Inês pelo apoio nos momentos mais difíceis do mestrado e da vida, aconselhando e motivando todo o tempo. Além disso, agradeço aos meus pais Elcio e Betty pois sem sua dedicação, suporte e carinho jamais teria chegado tão longe. Por último, agradeço ao meu Gabriel pela paciência durante os momentos de ausência e intenso *stress*, pela confiança, atenção e carinho durante cada dia de preparação deste trabalho.

À todos o meus mais sinceros agradecimentos.

RESUMO

PEREIRA, Bianca de Oliveira. **Resolução de Entidades Nomeadas utilizando Recursos em Linked Data**. 2012, 174f. Dissertação (Mestrado em Informática) – Instituto de Matemática, Instituto Tércio Pacitti, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2012

A Web vem evoluindo fortemente desde seu surgimento em 1989. Dos provedores de conteúdo às redes sociais, o volume de informação disponível cresce exponencialmente e novas abordagens vêm surgindo para resolver problemas de representação da informação e busca. Com o intuito de facilitar o processamento do conhecimento expresso por toda essa informação, surgiu o conceito de Linked Data. Este novo conceito visa a identificação e anotação semântica de entidades presentes em páginas Web. Uma das formas de realizar esta anotação se dá através da Resolução de Entidades Nomeadas, ou seja, uma tarefa que visa identificar as Entidades Nomeadas citadas no texto da página e anotá-las com recursos já presentes em bases de dados no formato de Linked Data.

Neste trabalho visamos o problema de anotação de textos em linguagem natural com recursos de bases em Linked Data. Descrevemos a metodologia adotada e desenvolvemos uma arquitetura de componentes que implementa esta metodologia. Cada componente (Seleção de Bases de Dados em Linked Data, Reconhecimento de Propriedades-nome, Geração de Dicionário Nomes, Reconhecimento de Menções a Entidades Nomeadas e Desambiguação de Entidades Nomeadas) trabalha de forma independente. Para cada componente desenvolvemos e avaliamos uma série de métodos e tecnologias através de um estudo de caso com as bases de dados Linked Movie Database e Jamendo. Assim, verificamos que nossa metodologia e métodos associados podem ser utilizados para a anotação semântica de textos em páginas Web utilizando uma base de dados em Linked Data. Nosso principal diferencial para os outros trabalhos na área reside na possibilidade da escolha da base de dados pelo usuário, pois trabalhamos com bases de dados com metadados genéricos ao invés de metadados pré-definidos.

Palavras-chave: Resolução de Entidades Nomeadas, Entidades Nomeadas, Linked Data, Processamento de Linguagem Natural.

ABSTRACT

PEREIRA, Bianca de Oliveira. **Named Entity Resolution using Linked Data Resources**. 2012, 174f. Dissertação (Mestrado em Informática) – Instituto de Matemática, Instituto Tércio Pacitti, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2012

The Web has evolved greatly since its creation in 1989. From content providers to social networks, the volume of information available grows exponentially and new approaches have emerging to solve knowledge representation and search problems. The Linked Data concept has emerged to facilitate the processing of knowledge expressed by all this information. This new concept aims the identification and semantic annotation of entities from Web pages. One of the ways to perform this annotation is through Named Entity Resolution, a task that intends to identify Named Entities in a text and to annotate it with resources from a Linked Data dataset.

In this work we worked on the problem of semantic annotation of natural language texts with Linked Data resources. We described the methodology used and developed an architecture of components that implements this methodology. Each component (Linked Data Dataset Selection, Recognition of Properties that Identify Names, Gazetteer Generation, Recognition of Mentions of Named Entities and Named Entity Disambiguation) works independently. For each component we developed and evaluated a series of methods and technologies through a case study using both Linked Movie Database and Jamendo datasets Thus, we shown that our methodology and methods can be used for semantic annotation in Web pages using Linked Data datasets. Our main difference from other works in this domain is that we enable users to choose which Linked Data dataset they want because we work with generic metadata and not predefined metadata.

Keywords: Named Entity Resolution, Named Entities, Linked Data, Natural Language Processing.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

DCAT	Data Catalog Vocabulary
FOAF	Friend of a Friend
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IMDB	Internet Movie Database
NER	Named Entity Resolution
OWL	Web Ontology Language
RDF	Resource Description Framework
SKOS	Simple Knowledge Organization System
URI	Uniform Resource Identifier
URL	Unified Resource Locator
XML	eXtensible Markup Language

SUMÁRIO

1 INTRODUÇÃO

Desde a década de 50, os cientistas almejam a criação de um computador capaz de compreender e comunicar-se com um humano utilizando linguagem natural (TURING, 1950). Apesar dos esforços empregados em décadas pelos cientistas da área de Inteligência Artificial, este computador ainda não é uma realidade. Um dos maiores desafios na busca por este supercomputador é torná-lo capaz de compreender a linguagem natural humana, altamente dinâmica e cheia de ambiguidades. E é neste sentido que as áreas de Processamento de Linguagem Natural e mais atualmente a área de Web Semântica vem trabalhando.

A área de Wem Semântica trabalha com a formalização do conhecimento, tentando expressar o conhecimento humano e suas relações em um formato processável por máquina. A área de Processamento de Linguagem Natural visa transformar um texto, que não passa de um conjunto aleatório de caracteres, em um conjunto de informações que podem permitir processamento por máquina. É neste campo que o nosso trabalho se insere. Visamos unir as duas áreas de estudo na identificação de entidades do mundo real mencionadas em textos em linguagem natural.

1.1 Motivação

A World Wide Web (ou apenas Web) vem crescendo de forma exponencial desde a sua idealização em 1989 por Tim Berners-Lee. Com o barateamento dos computadores e a facilidade de conexão com a internet, a participação dos usuários mudou de meros leitores de páginas a uma efetiva atuação na geração e disseminação de informação on-line. Blogs, mídias sociais, microblogs e fóruns foram apenas algumas das aplicações que deram origem à chamada Web Interativa (ou Web 2.0). Nesta nova versão da Web, ocorreu o grande fênomeno da explosão da quantidade de dados disponível on-line, pois as informações que antes eram geradas apenas por um grupo de fontes, conhecidas por "provedores de conteúdo", passam agora a ter sua origem em qualquer pessoa que esteja conectada à rede mundial de computadores (ANDERSON, 2007).

Devido a essa quantidade cada vez maior de dados, tornaram-se necessários a criação e o aprimoramento de mecanismos que permitissem que a informação certa pudesse ser encontrada no momento desejado. Com isso, foram criados algoritmos aprimorados de indexação de arquivos, ranking de documentos, indicação de conteúdo relacionado, dentre outros, que têm por função básica extrair o máximo de informação relevante em um documento, a fim de encontrar o que o usuário realmente está procurando. Para atender às demandas geradas pela Web 2.0 surgiu então o que chamamos de Web Semântica.

A Web Semântica, ou Web 3.0, surgiu com a meta de adicionar semântica aos dados publicados na Web, ou seja, extrair o significado inerente aos dados publicados e expressá-lo em um formato processável por máquina. Sendo capaz de processar todo o conhecimento contido em um documento, a máquina passa a poder efetuar buscas otimizadas retornando apenas as respostas realmente relevantes. Dentre os primeiros métodos utilizados para tentar tornar a Web Semântica uma realidade, podemos

citar as buscas facetadas, o uso de tesauros e taxonomias, e a contextualização de documentos através do uso de marcações (tags) (BREITMAN, 2005). Mas, apesar de todo seu potencial, a Web como um todo ainda não pode ser considerada como uma Web Semântica, pois ainda não houve a implantação de nenhuma tecnologia semântica de forma massiva na Web. Este fato tende a mudar com o surgimento da ideia de Linked Open Data.

Em junho de 2011, Google, Microsoft e Yahoo, as empresas responsáveis pelos três maiores motores de busca na Web, reconheceram publicamente a utilidade da anotação semântica de documentos e uniram-se no projeto Schema.org[1] (MACBETH, 2011) (GOEL; GUPTA, 2011) (SETH, 2011). Este projeto visa criar um descritor para a anotação de dados em páginas Web em domínios comuns do conhecimento (livros, filmes, música, eventos, televisão, receitas culinárias, pessoas, lugares, organizações, produtos, etc). O objetivo da anotação de páginas web com este novo esquema de dados é possibilitar buscas mais refinadas que permitam a desambiguação de termos em linguagem natural, ou seja, quando realizamos uma busca utilizando a palavra-chave "manga" podemos definir se a palavra encontra-se no contexto das frutas, das roupas ou até mesmo das pessoas. Definir o contexto possibilita que os resultados retornados para o usuário sejam mais precisos do que utilizando apenas palavras de forma isolada.

Anterior ao projeto Schema.org, Tim Berners-Lee participou da idealização de um novo projeto para tentar tornar reais as pretensões da Web Semântica. Apresentado no TED 2009[2], o projeto Linked Open Data surgiu como um apelo à publicação de dados disponíveis publicamente na Web e evoluiu para um conjunto de padrões para publicação de dados. Segundo Bizer et al. (2009), Linked Data é definido como um conjunto de boas práticas para a publicação e conexão de dados estruturados na

[1] http://schema.org/
[2] http://www.ted.com/talks/tim_berners_lee_on_the_next_web.html

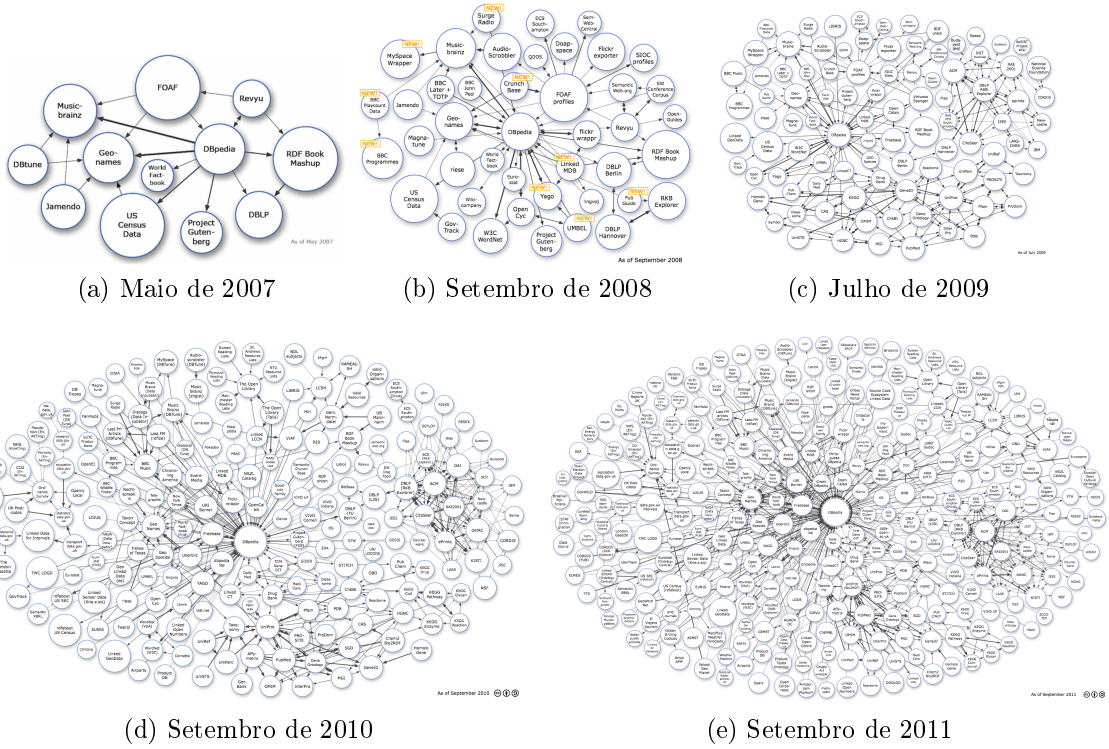
Web.

A vantagem primordial na adoção de Linked Data na publicação de dados é a utilização de URIs (Uniform Resource Identifier) como identificadores únicos para entidades e conceitos descritos por bases de dados na Web. O uso de URIs permite a identificação da individualidade de um item dentre todos os outros descritos na Web além de facilitar a sua consulta. Com a adoção de Linked Data é dada a possibilidade de modelagem do conhecimento humano de forma que seja processável por máquina.

A essa nova geração da Web, iniciada pela publicação de dados no formato de Linked Data, denominamos Web de Dados pois, diferentemente do que ocorria na Web 2.0, a ligação agora se dá entre dados diversos e não mais entre documentos. Enquanto na versão anterior (Web de Documentos) existem ligações sem nenhum significado associado, nessa nova Web é apresentado um conjunto de ontologias e vocabulários que têm por finalidade descrever não só o que os dados expressam mas o significado das ligações existentes entre eles. É neste ponto que ocorrem as maiores contribuições das mais diversas fontes não só de dados mas principalmente de descritores.

A publicação em Linked Data se popularizou com o surgimento da DBPedia (AUER et al., 2007), uma base de dados construída a partir das informações expressas na Wikipédia[3], a maior enciclopédia on-line construída de forma colaborativa. Por ser a pioneira a utilizar os padrões de Linked Data e por cobrir as mais diversas áreas do conhecimento, a DBPedia tornou-se o centro da chamada Nuvem de LOD, uma rede de bancos de dados abertos expressos usando Linked Data e ligados através da Web (Figura 1.1). A Nuvem de LOD é hoje a base da Web de Dados.

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



(a) Maio de 2007 (b) Setembro de 2008 (c) Julho de 2009

(d) Setembro de 2010 (e) Setembro de 2011

Figura 1.1: Bases de dados na nuvem de LOD ao longo do tempo (FONTE: *http://richard.cyganiak.de/2007/10/lod/*)

A anotação semântica de páginas web com recursos já existentes nas bases integrantes da Nuvem de LOD divide-se em dois processos: anotação de dados estruturados; e anotação de dados semi-estruturados ou sem estrutura.

Documentos com dados estruturados podem ser diretamente anotados utilizando recursos da Nuvem de LOD. A estrutura de planilhas, XML e bases relacionais, por exemplo, facilitam a anotação utilizando Linked Data pois as colunas das planilhas, as tags XML e os campos de uma base relacional podem ser diretamente mapeados para as estruturas de Linked Data, facilitando assim um possível processo de anotação. (BIZER; CYGANIAK, 2006) (AUER et al., 2009)

Enquanto isso, a anotação semântica de dados semi-estruturados ou sem estrutura

definida, como páginas em HTML (HyperText Markup Language) e textos em linguagem natural, requer um processamento maior para descoberta de quais são os conceitos e indivíduos expressos no página ou no texto. Devido a isto, a anotação deste tipo de fonte tende a ser feita manualmente. Contudo a anotação manual de uma fonte de dados com gigabytes de texto é um esforço que dispende muito tempo e dinheiro e, portanto, essa não é a forma mais adequada de se anotar. Além do mais, a ideia por trás da anotação semântica de dados não é somente descrever dados pertencentes a grandes bases de dados da Web, mas descrever também todo o conteúdo gerado diariamente pelos usuários advindos do ambiente da Web Interativa. Estes usuários não são cientistas da computação mas escritores, biólogos, publicitários, professores, adolescentes e donas de casa, logo não devem ser requisitados a demonstrar quaisquer conhecimentos técnicos para adequação de seu conteúdo à Web Semântica.

A fim de atender a essa demanda não suprida, nos últimos anos tem sido dada grande ênfase a ferramentas que auxiliem na anotação de textos e páginas HTML com URIs de entidades já presentes na nuvem de LOD. Estas ferramentas vão desde interfaces que auxiliam na anotação manual (KHALILI; AUER, 2012) a serviços e ferramentas para sugestão automática de entidades presentes no texto (RIZZO; TRONCY, 2012). Porém, mesmo com o grande empenho na geração de soluções para o reconhecimento automático de entidades, todas as soluções encontradas até o momento dedicam-se ao uso de um grupo restrito de bases da Nuvem de LOD. Esta restrição impõe que apenas um pequeno grupo de bases seja utilizado, enquanto a própria Nuvem de LOD vem crescendo fortemente ao longo dos anos. O problema gerado com as ferramentas atuais é que elas não permitem que o usuário possa anotar seus textos com as bases de seu interesse, bem como utilizar uma base que não esteja disponível na nuvem, como é o caso de bases de dados geradas a partir do conteúdo privado de empresas. Este fato torna o reconhecimento de entidades em texto utilizando recursos de bases em Linked Data genéricas um problema em aberto.

1.2 Problema

No contexto da Web Semântica, muitas ferramentas vêm sendo criadas como meio de extrair conhecimento de textos em linguagem natural e expressá-lo de forma que o computador seja capaz de processar seu conteúdo. Dentre estas ferramentas, identificamos aquelas que trabalham com a identificação e resolução de entidades nomeadas, ou seja, entidades que podem ser reconhecidas pela citação de seus nomes no texto.

Todas as ferramentas criadas até este momento utilizam um número limitado de bases de dados em Linked Data para a resolução de entidades nomeadas, impedindo que o usuário possa escolher a base de dados de seu interesse. O problema com o qual visamos trabalhar é, portanto, a ***inexistência de ferramentas de resolução de entidades nomeadas utilizando bases de dados em Linked Data que permitam a escolha da base pelo usuário***.

1.3 Proposta de Solução

Nossa proposta visa a identificação da metodologia mais comum na construção de ferramentas de resolução de entidades nomeadas utilizando bases de conhecimento prévio e sua adaptação permitindo a escolha da base de dados pelo usuário. A partir desta metodologia, desenvolveremos um conjunto de métodos que permitam o uso de bases de dados em Linked Data com uso de metadados genéricos como bases de conhecimento de entidades nomeadas. Por metadados genéricos queremos dizer que a base não precisa ter um conjunto pré-definido de metadados para que possamos utilizá-la. Os métodos desenvolvidos levarão em conta estruturas gerais de bases de dados em Linked Data mas não irão se ater a vocabulários e outras características

específicas de uma ou outra base.

Note que, apesar de permitir uma certa flexibilidade no uso dos metadados, não pretendemos permitir a utilização de qualquer base da nuvem de LOD. Antes, estabeleceremos um conjunto de critérios de qualidade que possibilitem a caracterização de uma base de dados como estando no formato de Linked Data e sua utilização em nossa aplicação.

A contribuição esperada com este trabalho é a geração de heurísticas para descoberta automática de propriedades que representam nomes em bases de dados em Linked Data, a construção de métodos para a transformação de bases em Linked Data em dicionários de Entidades Nomeadas e a geração de uma arquitetura para resolução (reconhecimento e desambiguação) de Entidades Nomeadas em textos em Linguagem Natural.

1.3.1 Objetivo Geral

Para resolvermos nosso problema de forma efetiva temos como objetivo geral: ***a automatização do processo de resolução de entidades nomeadas utilizando bases de dados em Linked Data com uso de metadados genéricos***. Ou seja, não podemos criar uma ferramenta que restrinja demais os metadados utilizados pela base de dados ou limitaremos o conjunto de bases a ser utilizado.

1.3.2 Objetivos Específicos

A partir de nosso objetivo geral e da proposta apresentada, apresentamos um conjunto de objetivos específicos a serem alcançados durante a execução de nosso tra-

balho:

- Identificação da metodologia a ser utilizada.
- Identificação de um conjunto de critérios de qualidade para bases em Linked Data visando a tarefa de Resolução de Entidades Nomeadas.
- Descoberta das estruturas que identificam as Entidades Nomeadas e seus nomes em bases de dados em Linked Data.
- Geração de um dicionário de nomes a partir de uma base de dados em Linked Data com uso de metadados genéricos.
- Identificação de menções a entidades nomeadas no texto utilizando o dicionário de nomes gerado.
- Desambiguação de entidades nomeadas utilizando bases de dados em Linked Data.

1.4 Hipótese

Durante este trabalho pretendemos realizar a validação da seguinte hipótese: ***É possível utilizar uma base de dados no formato de Linked Data para a tarefa de Resolução de Entidades Nomeadas sem que precisemos saber, a priori, o significado das classes e propriedades dado pelos descritores utilizados pela base.***

1.5 Delimitação do Trabalho

Nosso objetivo neste trabalho não é a anotação semântica conhecida como *Ontology Population* (CIMIANO, 2006), onde o texto é utilizado como fonte de informação para a construção de instâncias de uma ontologia ou vocabulário dados como entrada. Nosso trabalho está mais próximo da tarefa de *Entity Linking* (ZHANG et al., 2010) onde reconhecemos, no texto, entidades nomeadas já expressas em uma base de conhecimento externa.

Apesar de termos interesse em trabalhar com métodos que permitam o uso de diversos idiomas, iremos dar foco a textos em inglês. Este idioma foi escolhido por ser o idioma predominante em bases de dados em Linked Data. Ainda assim, não descartaremos a utilização de textos em outros idiomas como o espanhol, o português e o francês, pois muitos nomes tendem a mantêr sua escrita independente do idioma.

1.6 Estrutura do Trabalho

Iniciamos este trabalho com a apresentação da motivação que nos levou a estudar o tema, a identificação do problema a ser resolvido e a proposta de solução com os diversos objetivos a serem alcançados. No Capítulo 2 apresentamos alguns conceitos necessários à compreensão do nosso trabalho. A partir do Capítulo 3 iniciamos o desenvolvimento de nossa proposta de solução apresentando a metodologia levantada e sua adaptação ao nosso problema. Ainda no Capítulo 3 apresentamos a arquitetura de componentes desenvolvida para implementação de nossa metodologia. Cada um dos capítulos subsequentes detalha as características de cada um dos componentes: seleção de bases de dados em Linked Data (Capítulo 4), reconhecimento de propriedades-nome (Capítulo 5), geração de dicionário de nomes (Capítulo 6),

reconhecimento de menções a entidades nomeadas (Capítulo 7) e desambiguação de entidades nomeadas (Capítulo 8).

Criamos uma seção ao final de cada capítulo comentando os trabalhos relacionados à tarefa detalhada no capítulo. Da mesma forma, realizamos uma avaliação para cada componente de forma que cada um pudesse ser avaliado com o mínimo de interferência possível dos outros componentes. A cada capítulo tratamos da avaliação dos métodos desenvolvidos para cada um. Ao final, a avaliação do componente de desambiguação de entidades nomeadas pode ser vista como a avaliação do resultado de toda a nossa arquitetura.

Por fim, os critérios para utilização de uma base de dados em Linked Data com nossa metodologia são apresentados no Capítulo 4, nossas conclusões e trabalhos futuros são apresentados no Capítulo 9.

2 REFERENCIAL TEÓRICO

Neste capítulo faremos um breve levantamento de alguns conceitos necessários a uma melhor compreensão de nosso trabalho. Explicaremos a origem do termo *Entidade Nomeada* e seu significado enfatizando a diferença entre este tipo de entidade e uma *Entidade Comum*. Apresentaremos ainda o que são Gazetteers e como é feita a sua construção. A seguir, explicaremos o que caracteriza uma base de dados no formato de Linked Data e trataremos brevemente das tecnologias para representação e consulta a dados neste formato.

2.1 Entidades Nomeadas

Entidade Nomeada foi um termo cunhado em 1995 para a sexta *Message Understanding Conference*, uma conferência cujos trabalhos são agrupados em torno de um conjunto de tarefas a serem realizadas. Uma destas tarefas é a *Named Entity Recognition* (Reconhecimento de Entidades Nomeadas) onde o objetivo é reconhecer, em um texto, palavras que identificam indivíduos de determinadas classes, expressões temporais e quantidades.

Neste trabalho utilizaremos a seguinte definição de Entidade Nomeada.

Definição 2.1 ***Entidade Nomeada*** *é todo indivíduo que possui um nome próprio que o identifica unicamente dentre todos os outros membros de sua classe dado um determinado contexto.*

Entidades Nomeadas podem pertencer a diversas classes, desde pessoas, lugares e organizações a músicas, filmes e eventos. Se um indivíduo pode ser identificado por um nome próprio então ele tem grandes chances de ser considerado uma Entidade Nomeada.

A melhor forma de estabelecermos o que significa uma Entidade Nomeada é identificando também o que ela não é. Neste ponto devemos estabelecer a diferença entre uma Entidade Nomeada e uma Entidade Comum.

Sempre que um dado nome, em geral um nome próprio, nos remete automaticamente a um conjunto de características de um único indivíduo, caracterizando-o, podemos dizer que este indivíduo é uma entidade nomeada. Como exemplos podemos dizer que a cidade de Londres, o cantor Michael Jackson e a música Garota de Ipanema são Entidades Nomeadas pois ao citarmos seus nomes, expressamos implicitamente um conjunto de atributos intrínsecos aos indivíduos sem que sejam necessárias maiores descrições. O nome nos remete à história, à função ou a outras características das entidades. Isto é o que caracteriza o que chamamos de Entidade Nomeada.

Em outro caso, se um nome faz referência a qualquer um dos membros de um grupo de indivíduos, os indivíduos deste grupo são então conhecidos como Entidades Comuns. Entidades Comuns precisam de uma maior descrição para que haja diferenciação entre dois indivíduos. O *carro vermelho da Marina*, a *máquina de*

costura antiga no canto da sala e a *ração no pote do Rex* são exemplos de entidades comuns pois apenas os nomes carro, máquina de costura e ração não identificam a qual indivíduo estamos nos referindo dentro de uma dada classe. Neste caso apenas o nome não é suficiente, precisamos de um maior detalhamento para identificar o real indivíduo.

Enquanto as Entidades Nomeadas possuem um conjunto de atributos embutidos em seus nomes, as entidades comuns necessitam ter suas características explicitadas pois o nome não é suficiente para identificá-las.

Entidades Nomeadas não precisam ter um único nome como identificador e, em geral, há mais de um. “IBM”, “Big Blue"e “International Business Machines Corporation” são três possíveis nomes para a mesma organização e cada um faz o papel de identificador para esta entidade. Entidades Nomeadas podem também mudar de identificador conforme o contexto. Uma dada pessoa pode ser citada por seu nome completo em um documento oficial enquanto em uma carta para um parente pode ser citada por seu apelido. Estas são algumas das características que tornam a Resolução de Entidades Nomeadas uma tarefa tão complexa.

Outra característica complicadora é a duplicação de identificadores. Michael Jordan, por exemplo, pode ser um jogador de basquete em uma notícia sobre esportes ou pode ser o nome de um professor em um artigo científico. Um caso ainda mais complicado é o do nome “Orange” que serve como identificador para mais de 20 locais dentre cidades e vilarejos (WIKIPEDIA, 2012). Esses casos são tratados de forma mais profunda em algoritmos de desambiguação de Entidades Nomeadas onde é necessário não só reconhecer se um dado nome refere-se a uma Entidade Nomeada mas a qual entidade ele se refere.

É importante notar ainda que determinadas siglas podem ser consideradas referên-

cias (nomes) para Entidades Nomeadas tais quais USA, MEC, RJ, dentre outras. Porém, um identificador que não seja comum a um texto em linguagem natural não deve ser considerado um nome para uma Entidade Nomeada, tal qual o número do ISBN (International Standard Book Number) para um livro, por exemplo. Neste caso, se um número de ISBN é citado em um texto em linguagem natural, o leitor não é capaz de identificar automaticamente a que livro ele se refere sem ter de recorrer a um catálogo. Enquanto isso, se o nome do livro é citado, o leitor tem chance de reconhecer o livro rapidamente caso já tenha entrado em contato com ele anteriormente.

2.2 Gazetteer

Um dos métodos para o reconhecimento de Entidades Nomeadas parte da ideia de que há um conhecimento prévio sobre as entidades conhecidas e seus nomes. Uma das formas de representar este conhecimento prévio acontece através da utilização de um dicionário de Entidades Nomeadas, também conhecido como Gazetteer.

Definição 2.2 *Gazetteer é um dicionário de Entidades Nomeadas representado por uma estrutura do tipo chave-valor. A chave é um identificador para Entidades Nomeadas e o valor é uma lista com todas as Entidades Nomeadas conhecidas por este identificador.*

O campo chave de um Gazetteer é sempre o identificador da Entidade Nomeada em um texto em linguagem natural, seja seu nome ou uma sigla. O campo valor, por sua vez, pode mudar dependendo da fonte de dados utilizada para geração do Gazetteer. Se um Gazetteer tem como fonte um portal web, então o campo valor irá conter as URLs (Unified Resource Locator) que representam as Entidades Nomeadas nesse

portal. Por outro lado, se a fonte for uma base de dados em Linked Data então o campo valor será dado pelos URIs (Uniform Resource Identifier) dos recursos presentes na base.

A construção de um Gazetteer acontece de acordo com a seguinte sequência de passos:

1. Identificação das Entidades Nomeadas presentes na fonte dos dados.
2. Identificação dos nomes que servem como identificadores destas Entidades Nomeadas.
3. Construção dos pares chave-valor.

Como exemplo iremos começar citando três Entidades Nomeadas do mundo real, a saber: Londres, a capital da Inglaterra; Michael Jackson, o cantor pop que participou de um grupo musical chamado Jackson 5; e Apple, uma empresa de tecnologia criadora do iOS.

Apenas com estes dados já temos o passo 1. No passo seguinte identificamos os nomes das Entidades Nomeadas levantadas: Londres, Michael Jackson e Apple. A partir daqui podemos criar uma série de Gazetteers. O primeiro exemplo de Gazetteer visa a utilização exclusiva do parágrafo anterior como fonte para os dados, sendo assim o campo valor será composto pela descrição de cada entidade e o Gazetteer ficará como no Quadro 2.1. Neste exemplo o Gazetteer funciona como uma enciclopédia.

Outro exemplo de Gazetteer para as mesmas entidades é aquele onde, ao invés do valor ser a descrição textual, ele passa a ser a URL de uma página web que representa a mesma Entidade Nomeada. Se utilizarmos as páginas do sítio web da Wikipédia, por exemplo, o Gazetteer ficaria como mostrado no Quadro 2.2.

Quadro 2.1: Exemplo de Gazetteer com descrição textual

chave	valor
Londres	Capital da Inglaterra.
Michael Jackson	Cantor pop que participou de um grupo musical chamado Jackson 5.
Apple	Empresa de tecnologia criadora do iOS.

Quadro 2.2: Exemplo de Gazetteer usando um portal web como fonte dos dados

chave	valor
Londres	http://pt.wikipedia.org/wiki/Londres
Michael Jackson	http://pt.wikipedia.org/wiki/Michael_Jackson
Apple	http://pt.wikipedia.org/wiki/Apple

Um terceiro Gazetteer, ainda com as mesmas entidades, pode ser construído a partir de uma base de dados em Linked Data. Se utilizarmos a DBPedia (AUER et al., 2007) como fonte para os dados, o campo valor será composto pelo URI do recurso que representa a Entidade Nomeada de que se quer tratar (Quadro 2.3).

Quadro 2.3: Exemplo de Gazetteer usando uma base de dados em Linked Data como fonte dos dados

chave	valor
Londres	http://dbpedia.org/resource/London
Michael Jackson	http://dbpedia.org/resource/Michael_Jackson
Apple	http://dbpedia.org/resource/Apple

O Gazetteer de forma isolada é uma estrutura de dados que, para ter um uso efetivo, necessita de um serviço de consulta associado. A este serviço damos o nome de Serviço de Lookup, que pode ser traduzido como uma função que recebe como parâmetro um nome e retorna todas as Entidades Nomeadas conhecidas por este nome (Fórmula (2.1)).

$$f(chave) = valor \tag{2.1}$$

2.3 Linked Data

Linked Data é um novo conceito para publicação de dados na Web. Seu principal objetivo é a interligação de dados de fontes diferentes permitindo a adição de informação semântica. Este novo conceito pode ser visto como um novo modelo de Banco de Dados distribuído na Web ou como uma grande base de conhecimento para agentes de Inteligência Artifical. Neste trabalho a visão que adotaremos é a mesma de Bizer, Heath e Berners-Lee (2009) de que "Linked Data é um conjunto de boas práticas para publicação de dados estruturados na Web".

Para que um dado seja considerado no formato de Linked Data, existem quatro princípios básicos a serem atendidos (BERNERS-LEE, 2006):

1. Uso de URIs como nome para coisas.
2. Uso de HTTP URIs para que pessoas possam visualizar mais informações sobre estes nomes.
3. Quando alguém procurar por um URI, deve ser fornecida informação útil utilizando padrões (RDF*, SPARQL).
4. Inclusão de links para outros URIs para que mais coisas possam ser descobertas.

Segundo o primeiro princípio, cada conceito ou indivíduo do mundo real é modelado como um recurso único com uma URI individual que o identifica.

De acordo com o segundo princípio, estes URIs devem poder ser acessáveis via protocolo HTTP (Hypertext Transfer Protocol). Ao ser realizado um acesso, devem

ser retornadas informações sobre o recurso que está sendo consultado, tais como seus atributos e relacionamentos com outros recursos.

O terceiro princípio indica a utilização dos padrões RDF (Resource Description Framework) e SPARQL. O RDF é um formato para descrição de dados baseado em triplas e que se divide em vários subformatos: RDF/XML, RDFa, Turtle, N3, dentre outros. SPARQL, por sua vez, é uma linguagem de consulta própria para bases de triplas.

Por fim, o último princípio indica que recursos em Linked Data não devem estar isolados, mas relacionados a outros recursos. Este conjunto de relacionamentos é o que dará o contexto de existência de um recurso em Linked Data.

Além dos princípios indicados, existe ainda uma característica muito própria ao conceito de Linked Data e à descrição de dados utilizando RDF. As classes às quais os recursos pertencem, seus atributos e relacionamentos são descritos de forma a que um humano também possa compreender seu significado. Sendo assim, vocabulários e ontologias são utilizados para definir o que significa cada um dos itens relacionados a um recurso.

A partir das bases de dados publicadas na Web no formato de Linked Data, surgiu o que chamamos de Nuvem de LOD. Criada a partir do incentivo dado pela DBPedia, a nuvem de LOD vem crescendo rapidamente ao longo dos anos ao ponto de muitas bases já nem sequer aparecerem no desenho da Nuvem de LOD (Figura 1.1 - Capítulo 1). Os dados publicados hoje já se apresentam nos mais diversos domínios de conhecimento e a tendência é que essa oferta se expanda cada vez mais.

2.3.1 RDF

O Resource Description Framework, ou apenas RDF, é um "modelo padrão para troca de dados na Web" (IVAN, 2004). Sua principal característica é a capacidade de representar dados descritos sob os mais diversos esquemas e, por este motivo, foi escolhido como formato para a descrição de dados em Linked Data.

Dados em RDF são representados no formato de triplas sujeito-predicado-objeto. O sujeito é sempre o recurso que está sendo descrito, o predicado pode ser uma propriedade que identifica um atributo ou um relacionamento com outro recurso; e o objeto depende do predicado utilizado. Caso o predicado seja uma propriedade que indica um atributo, então o objeto é um literal representando o valor que o atributo assume. Caso o predicado indique um relacionamento, então o objeto passa a ser o URI que identifica o recurso alvo do relacionamento (Figura 2.1).

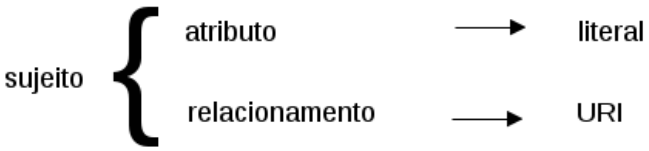


Figura 2.1: Componentes de Triplas RDF

Um dos relacionamentos mais comuns na descrição de recursos RDF é o rdf:type que especifica a classe à qual o recurso pertence. Como objeto deste tipo de relacionamento podemos utilizar as mais diversas classes. Dentre os vocabulários mais utilizados (BIZER; JENTZSCH; CYGANIAK, 2011) podemos utilizar, por exemplo, a classe Organization do vocabulário Friend of a Friend (foaf:Organization) para a indicação de que o recurso identifica uma organização ou a classe Record da Music Ontology (mo:Record) para indicar a representação de uma gravação musical.

Cada um dos itens na descrição de um recurso RDF possui um URI próprio. Por

item entendemos o recurso propriamente dito, as propriedades e as classes. Todos os itens exceto o próprio recurso têm seu significado descrito por um vocabulário ou ontologia.

Como exemplo, iremos demonstrar a construção de um recurso RDF representando o cantor pop Michael Jackson com base em algumas informações dispostas na base de dados da DBPedia. Iremos trabalhar com o texto apresentado no Quadro 2.4a de forma que ele é todo o conhecimento que possuímos sobre o cantor.

O primeiro passo na criação de um recurso RDF consiste em decidir qual o URI que irá representar o recurso. No caso da DBPedia, o URI é formado pelo endereço principal da DBPedia seguido pelo texto "resource" e depois o nome do recurso na Wikipédia. Sendo assim, o URI para o nosso recurso será http://dbpedia.org/resource/Michael_Jackson.

O segundo passo está em decidirmos qual será a classe à qual o recurso irá pertencer. Como trata-se de uma pessoa, iremos utilizar a classe Person (foaf:Person) descrita pelo vocabulário Friend of a Friend (FOAF) (BRICKLEY; MILLER, 2010).

O terceiro passo reside em definir quais os atributos do recurso. Como atributos podemos identificar o nome do cantor, sua página web e a sua foto. Cada um deles deve ser representado por uma propriedade com URI próprio. A melhor forma de fazer isto é aproveitando propriedades descritas por vocabulários já existentes ou, caso não exista, devemos criar um novo vocabulário. Em nosso exemplo, verificamos que o FOAF já possui propriedades adequadas para representação destes atributos, sendo elas: foaf:name para definição do nome, foaf:homepage para descrição da página web e foaf:depiction para identificar uma foto que represente o recurso.

O quarto passo é a identificação de relacionamentos. No nosso exemplo podemos

identificar que Michael Jackson conhece Janet Jackson que, no caso, é sua irmã. Para isto podemos utilizar o relacionamento foaf:knows com o URI de identificação do recurso que representa Janet Jackson (http://dbpedia.org/resource/Janet_Jackson). A descrição completa do recurso utilizando o formato RDF/XML consta no Quadro 2.4b.

Quadro 2.4: Exemplo de triplificação de informações

(a) Texto a ser triplificado.

O cantor pop Michael Jackson, cujo nome de batismo é Michael Joseph Jackson, publicou em sua página web http://www.michaeljackson.com todas as fotos de sua turnê com sua irmã, Janet Jackson. De toda forma sua foto mais famosa não foi publicada na página oficial e sim em http://upload.wikimedia.org/wikipedia/commons/0/04/Michael_Jackson_1984.jpg

(b) Resultado da triplificação em RDF/XML

```
<rdf: RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:foaf="http://xmlns.com/foaf/0.1/">
    <rdf:Description rdf:about="http://dbpedia.org/resource/Michael_
Jackson">
        <rdf:type rdf:resource="http://xmlns.com/foaf/0.1/Person"/>
        <foaf:name>Michael Joseph Jackson</foaf:name>
        <foaf:homepage> http://www.michaeljackson.com</foaf:homepage>
        <foaf:depiction> http://upload.wikimedia.org/wikipedia/commons/
0/04/Michael_Jack
son_1984.jpg</foaf:depiction>
        <foaf:knows rdf:resource="http://dbpedia.org/resource/Janet_Jack
son"/>
    </rdf:Description>
</rdf:RDF>
```

2.3.2 SPARQL

SPARQL é uma das linguagens de consulta associadas a bases de dados que guardam triplas RDF. Apesar de existirem outras linguagens para o mesmo fim, a linguagem SPARQL foi citada por Tim Berners-Lee como a linguagem padrão para uso com Linked Data (BERNERS-LEE, 2006) e por isso é a mais utilizada para consultas neste tipo de base.

Para que possamos efetuar uma consulta SPARQL sobre um conjunto de dados em RDF precisamos de um conjunto de mecanismos capazes de processar a consulta e buscar, da forma mais eficiente possível, a resposta desejada. O conjunto de mecanismos que permitem a realização de consultas SPARQL dá origem ao que chamamos de SPARQL Endpoint.

Um SPARQL Endpoint pode permitir acesso a uma única base de dados ou a um conjunto de bases. Esta característica muda de uma implementação para outra e está fora do escopo deste trabalho (Para maiores informações consulte os trabalhos do W3C SPARQL Working Group[1]).

Para melhor compreensão dos capítulos que tratam especificamente da arquitetura de nossa solução, trataremos nesta seção sobre a construção de consultas SPARQL e qual o resultado esperado por cada uma delas.

O primeiro passo para compreensão de como as consultas SPARQL funcionam é entender qual a sua estrutura básica. Toda consulta SPARQL é formada por, pelo menos, um conjunto de variáveis que se quer conhecer e um conjunto de restrições. As variáveis que estão sendo consultadas aparecem na cláusula SELECT e as restrições fazem parte da cláusula WHERE (Quadro 2.5).

[1] http://www.w3.org/2009/sparql/wiki/Main_Page

Quadro 2.5: Estrutura básica de uma consulta SPARQL

```
SELECT ?variavel1 ?variavel2 ... ?variaveln
WHERE {
    restrição1
    restrição2
    ...
    restriçãon
}
```

As restrições básicas presentes na cláusula WHERE aparecem sempre na forma de triplas (sujeito, predicado, objeto) como na estrutura RDF. Se nossa consulta é por informações no campo do sujeito, a variável que queremos consultar será colocada nesta posição, ocorrendo o mesmo para informações sobre o predicado ou o objeto. Sendo assim, uma consulta onde queremos descobrir todos os sujeitos existentes na base seria uma consulta como a apresentada no Quadro 2.6. Ou seja, todas as três possíveis posições das triplas são ocupadas por variáveis, portanto não estamos restringindo que o predicado ou o objeto tenham qualquer valor pré-definido. Ao indicarmos na cláusula SELECT que estamos interessados em conhecer apenas o conteúdo da variável sujeito, o resultado da consulta será uma lista com todos os sujeitos de cada tripla presente na base.

Quadro 2.6: Exemplo de uma consulta SPARQL

```
SELECT ?sujeito
WHERE {
    ?sujeito ?predicado ?objeto.
}
```

Se quisermos restringir nossa consulta anterior para que ela retorne apenas os sujeitos de triplas cujo predicado é a propriedade http://xmlns.com/foaf/0.1/name

substituímos então a variável predicado pelo valor que queremos, ficando com o apresentado no Quadro 2.7.

Quadro 2.7: Exemplo de uma consulta SPARQL com restrição

```
SELECT ?sujeito
WHERE {
    ?sujeito <http://xmlns.com/foaf/0.1/name> ?objeto.
}
```

Outra forma de realizar esta restrição acontece através da utilização do comando FILTER. Este comando permite restringir os possíveis valores de uma variável. Nosso exemplo utilizando FILTER ficaria como apresentado no Quadro 2.8.

Quadro 2.8: Exemplo de uma consulta SPARQL utilizando FILTER

```
SELECT ?sujeito
WHERE {
    ?sujeito ?predicado ?objeto.
    FILTER(?predicado = <http://xmlns.com/foaf/0.1/name>)
}
```

O campo FILTER permite ainda um tipo especial de filtro chamado regex. Este filtro utiliza expressões regulares para restringir o valor de uma determinada variável. Sua utilidade maior está na restrição de valores literais no objeto das triplas. Em um exemplo onde procuramos pelo sujeito de toda tripla cujo valor do objeto inicia-se por uma letra "A" maiúscula utilizamos a consulta presente no Quadro 2.9.

Outra forma de lidar com a busca por informações em um campo se dá através do uso de full-text search. Este tipo de busca é limitada a alguns SPARQL Endpoints,

Quadro 2.9: Exemplo de uma consulta SPARQL utilizando expressão regular

```
SELECT ?sujeito
WHERE {
    ?sujeito ?predicado ?objeto.
    FILTER(regex(?objeto," ^ A"))
}
```

sobretudo os que funcionam sobre a base de triplas da OpenLink Virtuoso[2]. Por ser altamente otimizada para busca textual, este tipo de consulta tende a ser muito utilizado na busca por literais na posição de objeto nas triplas. O uso de full-text search ocorre através da inserção de mais uma linha na cláusula WHERE. Esta nova linha contém: a variável que se quer restringir, o uso da diretiva bif:contains no campo de predicado e o valor que se quer buscar no objeto. Uma consulta simples pelo nome Michael pode ser vista no Quadro 2.10.

Quadro 2.10: Exemplo de uma consulta SPARQL utilizando full-text search

```
SELECT ?sujeito
WHERE {
    ?sujeito ?predicado ?objeto.
    ?objeto bif:contains "'Michael"'.
}
```

No tipo de seleção que estamos realizando, permitimos que o valor da variável sujeito possa repetir-se várias vezes. Se nosso objetivo é incluir na lista de resultados apenas valores distintos devemos então utilizar a cláusula DISTINCT (Quadro 2.11).

Existem ainda algumas outras cláusulas como a LIMIT, a ORDER BY e a COUNT.

[2]http://virtuoso.openlinksw.com/

Quadro 2.11: Exemplo de uma consulta SPARQL utilizando DISTINCT

```
SELECT DISTINCT ?sujeito
WHERE {
    ?sujeito ?predicado ?objeto .
    FILTER(?predicado = <http://xmlns.com/foaf/0.1/name>) .
}
```

A LIMIT limita o número de resultados retornados, a ORDER BY ordena os resultados segundo o valor de uma dada variável, de forma crescente ou decrescente; e a COUNT é utilizada para contar quantos valores foram retornados para uma determinada variável.

O Quadro 2.12 mostra um exemplo onde limitamos o número de resultados aos 50 primeiros resultados encontrados e ordenamos alfabeticamente de acordo com o valor da variável sujeito. O Quadro 2.13 conta quantos sujeitos distintos existem na base de dados.

Quadro 2.12: Exemplo de uma consulta SPARQL utilizando LIMIT e ORDER BY

```
SELECT ?sujeito
WHERE {
    ?sujeito ?predicado ?objeto .
} ORDER BY (?sujeito) LIMIT 50
```

Maiores informações sobre a construção de consultas SPARQL podem ser encontradas na descrição da linguagem por Prud'hommeaux e Seaborne (2008).

Quadro 2.13: Exemplo de uma consulta SPARQL utilizando COUNT

```
SELECT COUNT(DISTINCT ?sujeito)
WHERE {
    ?sujeito ?predicado ?objeto .
}
```

2.3.3 Vocabulários e Ontologias

A característica mais importante de tecnologias semânticas é a atribuição de metadados aos dados que estão sendo descritos. Em outras palavras, todo dado é acompanhado por seu significado. Este significado pode ser descrito de diversas formas (BREITMAN, 2005), sendo a utilização de Vocabulários e Ontologias a mais comum.

Vocabulários são a forma mais simples de descrição de dados pois são formados unicamente por uma lista de conceitos e seus significados. Por sua vez, ontologias são, segundo Studer, Benjamins e Fensel (1998, p. 24): "especificações formais e explícitas de uma conceitualização compartilhada". Ou seja, ontologias contam não só com conceitos e seus significados mas também com a formalização lógica destes conceitos.

Segundo a W3C não há distinção clara entre vocabulários e ontologias ao aplicarmos no contexto da Web Semântica. Costuma-se chamar vocabulário para descrições mais simples e ontologias para descrições mais complexas. Sendo assim, doravante utilizaremos apenas o termo Vocabulário para indicar o descritor de dados aplicados à Web Semântica. A visão de Vocabulário utilizada pela W3C (2012) é:

> "Na Web Semântica, vocabulários definem conceitos e relacionamentos (também referidos como "termos") usados para descrever e representar uma área de interesse. Vocabulários são utilizados para classificar os termos que podem ser usados em uma aplicação particular, caracterizar possíveis relacionamentos e definir possíveis restrições na utilização destes termos. Na prática, vocabulários podem ser muito complexos (com vários milhares de termos) ou muito simples (descrevendo um ou dois conceitos apenas)."

Linked Data surgiu como uma metodologia para publicação de dados semânticos, sendo assim, uma das características desta publicação é o uso de descritores para os dados. Os vocabulários são aplicados na descrição das classes, atributos e relacionamentos expressos em um arquivo RDF. Cada um destes itens possui um URI próprio que remete ao vocabulário que o descreve.

Cada vocabulário possui o que chamamos de *namespace*. O *namespace* é o identificador (URI) do vocabulário e todos os conceitos descritos por um vocabulário estão dentro de seu *namespace*. A partir desta característica torna-se fácil descobrir a localização do significado de qualquer item presente em um arquivo RDF. Um exemplo para o uso de *namespace* é o vocabulário Friend of a Friend (BRICKLEY; MILLER, 2010) que é amplamente utilizado para descrição de relações sociais. Seu *namespace* é http://xmlns.com/foaf/0.1/ e a propriedade "name" descrita por este vocabulário tem como URI http://xmlns.com/foaf/0.1/name cuja parte inicial é justamente o *namespace* do vocabulário.

Se bem utilizados, vocabulários são muito úteis para aplicações de Resolução de Entidades Nomeadas. São eles que irão especificar as classes que identificam Entidades Nomeadas e as propriedades que indicam os nomes pelos quais estas entidades são conhecidas. No caso de Linked Data, estes vocabulários permitem ainda verificar a

consistência dos dados, seja identificando as classes que podem estar presentes em um relacionamento, seja definindo se a instância de uma classe pode ou não conter um determinado atributo.

3 METODOLOGIA

Para realização da tarefa de Resolução de Entidades Nomeadas precisamos fazer um levantamento para especificar qual a metodologia a ser seguida. Apesar da literatura não formalizar uma metodologia para esta tarefa, pudemos notar que grande parte dos trabalhos (MENDES et al., 2011) (YOSEF et al., 2011) (HAKIMOV; OTO; DOGDU, 2012)seguem um conjunto de passos que podem, de forma geral, ser agrupados em duas etapas principais: o mapeamento de menções a entidades nomeadas e a desambiguação de entidades nomeadas.

O **mapeamento de menções a entidades nomeadas** visa identificar os trechos textuais que fazem referência a Entidades Nomeadas e identificar as possíveis entidades citadas. A esses trechos chamaremos menções, que são nomes ou siglas pelas quais as Entidades Nomeadas são conhecidas. O mapeamento em si consiste na identificação das menções e no levantamento de todas as entidades nomeadas que podem estar sendo mencionadas, ou seja, todas as entidades que são conhecidas por cada um destes nomes. Como um dado nome, sem análise de contexto, pode fazer referência a mais de uma entidade nomeada então o resultado desta etapa consiste em uma ou mais entidades nomeadas candidatas a anotarem cada uma das menções.

Concluída a etapa anterior, passamos então à **desambiguação de entidades nomeadas**. Esta etapa é responsável por identificar qual a real Entidade Nomeada que esta sendo referenciada por uma dada menção. Para isso é feita a análise do contexto no qual a menção aparece. Como produto desta etapa temos um mapa formado pelas menções existentes no texto e por uma Entidade Nomeada para cada menção.

Esta metodologia (Figura 3.1) contempla todos os passos da Resolução de Entidades Nomeadas. O maior problema deste modelo é o fato de que ele não permite ao usuário a escolha da(s) base(s) que deseja utilizar.

Figura 3.1: Metodologia adotada por ferramentas de Resolução de Entidades Nomeadas

Nosso trabalho visa resolver a brecha deixada pelas ferramentas que adotam a metodologia apresentada. Para isto, modificamos a metodologia adicionando uma etapa anterior às outras (Figura 3.2). Esta nova etapa a qual denominamos **seleção da base de dados em Linked Data** consiste em permitir que o usuário da aplicação possa definir qual a base de dados que deseja utilizar. Esta nova etapa poderia lidar com qualquer tipo de base de dados mas neste trabalho iremos nos limitar a trabalhar apenas com bases de dados em Linked Data.

Figura 3.2: Metodologia modificada para permitir a escolha da base de dados

A partir da identificação da metodologia a ser adotada, desenvolvemos uma arquitetura com os componentes a serem desenvolvidos em nossa aplicação.

Definimos que a primeira etapa da metodologia fica, por hora, a cargo do usuário. Ou seja, não nos deteremos na etapa de identificação de qual a melhor base a ser utilizada para anotar o texto. Limitar-nos-emos a definir os critérios mínimos para que uma base de dados possa ser utilizada com nossa abordagem. O foco deste trabalho irá residir nas etapas de mapeamento de menções e desambiguação de entidades nomeadas utilizando a base de dados disponibilizada na primeira etapa.

As duas etapas restantes podem ser construídas das mais diversas formas (vide Seção 3.1). Sendo assim, optamos por utilizar 3 diferentes componentes para a realização da etapa de mapeamento de menções a entidades: reconhecimento de propriedades-nome, geração de dicionário de nomes e reconhecimento de menções a entidades nomeadas. A etapa de desambiguação de entidades nomeadas foi contemplada por

um único componente de mesmo nome. Na Figura 3.3 apresentamos todos os componentes desenvolvidos para nossa aplicação (em cinza claro) bem como todas as entradas necessárias e as saídas esperadas por cada um (em cinza escuro). As setas indicam o fluxo de processamento das informações desde o texto dado como entrada até a anotação do texto já desambiguada.

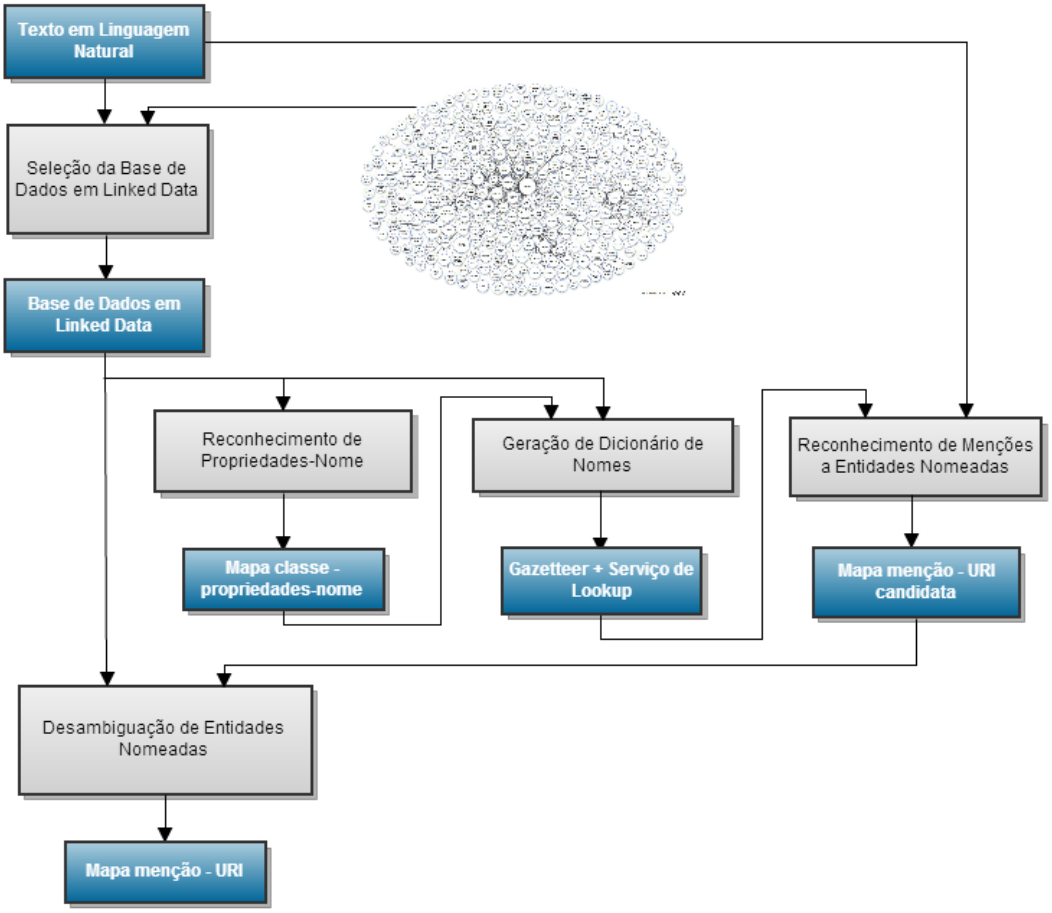


Figura 3.3: Arquitetura proposta para a Resolução de Entidades Nomeadas em texto em linguagem natural

A arquitetura como um todo espera como entrada um texto em linguagem natural, sem qualquer tipo de marcação tais quais tags ou outras marcações de estilo, além da base de dados em Linked Data selecionada. Ela prevê, como saída, um mapa com

todas as menções identificadas no texto relacionadas aos identificadores dos recursos da base em Linked Data que representam a real Entidade Nomeada citada.

Cada componente da arquitetura bem como os métodos desenvolvidos para construção de cada um serão apresentados em detalhes nos capítulos seguintes.

3.1 Trabalhos Relacionados

Existem diversos trabalhos que tencionam identificar as entidades nomeadas mencionadas em texto. Todos estes trabalhos são conhecidos como atuando na tarefa de Resolução de Entidades Nomeadas (Named Entity Resolution, em inglês) (PILZ; MOLZBERGER; PAASS, 2009). Ao longo do tempo, várias tarefas foram sendo criadas para o reconhecimento de entidades nomeadas e, posteriormente, para a sua desambiguação utilizando alguma base de conhecimento como suporte. Dentre estas tarefas podemos citar: Reconhecimento de Entidades Nomeadas (Named Entity Recognition), Desambiguação de Entidades (Named Entity Disambiguation) e Entity Linking.

Estas áreas de estudo são extremamente interligadas e costumam mesclar-se em diversos trabalhos. Sendo assim, foi complicado identificar um padrão de arquitetura ou de metodologia para todos os trabalhos. Portanto, apresentaremos uma descrição de cada uma das tarefas citadas e levantaremos um conjunto de trabalhos que apresentam o mesmo objetivo que o nosso trabalho: identificar as menções a entidades nomeadas em texto e realizar a desambiguação através de links para representações computacionais das entidades nomeadas.

A primeira tarefa a relacionar-se com o termo 'entidade nomeada' surgiu em 1995 em uma conferência baseada em Maryland, nos Estados Unidos. A sexta Message

Understanding Conference (MUC) foi o evento responsável por cunhar o termo 'entidade nomeada' (do inglês, named entity) (GRISHMAN; SUNDHEIM, 1996) e a dar origem à tarefa de Reconhecimento de Entidades Nomeadas.

Podemos definir Reconhecimento de Entidades Nomeadas como a tarefa de identificar e subcategorizar todas as menções a nomes de entidades, expressões temporais e expressões numéricas no texto (GRISHMAN, 1995). Ou seja, esta tarefa abrange a identificação de menções a entidades nomeadas, além da identificação da categoria (ou classe) a qual a entidade mencionada pertence.

Diversos trabalhos foram realizados a fim de melhorar o processo de Reconhecimento de Entidades Nomeadas, desde trabalhos apresentados nas diversas versões da MUC (MILLER, 1995) (KRUPKA, 1995) (COWIE, 1995) (YU; BAI; WU, 1998) (CHEN et al., 1998) (BORTHWICK et al., 1998) até trabalhos amplamente citados pela comunidade científica (NADEAU; SEKINE, 2007) (MIKHEEV; MOENS; GROVER, 1999).

Posteriormente, surgiu a necessidade não só de identificar nomes no texto mas de realizar a desambiguação em casos onde um mesmo nome pode referir-se a um conjunto de entidades pertencentes a classes diferentes. Um exemplo seria o nome "Orange" que pode referir-se a diversos tipos de localização, desde pequenos vilarejos até cidades maiores (WIKIPEDIA, 2012). Dessa necessidade surgiu o que chamamos de Desambiguação de Entidades Nomeadas (WACHOLDER; RAVIN; CHOI, 1997). Ou seja, a identificação de qual a real classe da entidade citada no texto.

Até este ponto, a única preocupação dos trabalhos era a identificação de nomes no texto e a classe de entidade a qual o nome se referia. Com o passar do tempo, os trabalhos em Reconhecimento e Desambiguação de Entidades Nomeadas foram mudando seu foco para identificar exatamente qual a entidade que estava sendo mencionada no texto. A partir de então, alguma base de conhecimento que descreve

entidades nomeadas é utilizada para identificar exatamente que entidade está sendo mencionada (BUNESCU; PASCA, 2006) (CUCERZAN, 2007). Esta identificação se dá através de anotações no texto com o identificador da entidade utilizado pela base de conhecimento. Com esta nova abordagem surgiu o termo Resolução de Entidades Nomeadas (PILZ; MOLZBERGER; PAASS, 2009) que abrange as antigas tarefas de Reconhecimento e Desambiguação de Entidades Nomeadas.

A tarefa de Desambiguação assume que todo o conhecimento sobre entidades nomeadas está contido na base de conhecimento utilizada, ou seja, se o nome reconhecido no texto não corresponde a uma entidade existente na base então este nome não deve ser reconhecido como menção a uma entidade nomeada. Com o tempo, como o algoritmo utilizado para identificação dos nomes no texto nem sempre utilizava a base de conhecimento dedicada ao processo de desambiguação, surgiu a tarefa que ficou conhecida como Entity Linking (MCNAMEE; DANG, 2009) (RAO; MCNAMEE; DREDZE, 2011). A diferença fundamental entre Desambiguação e Entity Linking é que a segunda assume que um nome no texto pode ser uma menção a uma entidade nomeada mesmo que a entidade não seja encontrada na base de conhecimento.

Hoje os trabalhos utilizam as diversas expressões de forma intercambiável, algumas vezes com os sentidos iniciais, noutras com o significado mais atual. Neste trabalho, visamos a tarefa de Resolução de Entidades Nomeadas com a utilização de bases de dados em Linked Data como fonte para os nomes das entidades nomeadas mencionadas no texto e para as descrições responsáveis por sua desambiguação.

Dentre os trabalhos na área de Resolução de Entidades Nomeadas podemos citar os trabalhos que utilizam a Wikipedia (CUCERZAN, 2007) e aqueles que utilizam bases em Linked Data. Dos trabalhos que utilizam Linked Data podemos identificar o DBPedia Spotlight (MENDES et al., 2011) e o AIDA (YOSEF et al., 2011), além de algumas aplicações comerciais.

O DBPedia Spotlight trabalha tanto na tarefa de Entity Linking quanto em Resolução de Entidades Nomeadas, dependendo apenas do tipo de algoritmo utilizado. A base de conhecimento utilizada é a DBPedia (BIZER; HEATH; BERNERS-LEE, 2009), que pode ser a fonte para os nomes ou pode ser utilizada apenas para o processo de desambiguação.

O AIDA é um trabalho que utiliza a base de dados YAGO (SUCHANEK; KASNECI; WEIKUM, 2007) para a Resolução de Entidades Nomeadas, tendo por foco o processo de desambiguação.

Dentre as aplicações comerciais que utilizam Linked Data em sua construção podemos citar a Alchemy API[1], a Open Calais[2] e a Zemanta[3]. O grande problema na análise das aplicações comerciais é que podemos avaliar seu resultado mas não podemos avaliar o processo, visto que não há muita informação sobre ele.

Dentre os trabalhos que levantamos, verificamos que o processo de Resolução de Entidades Nomeadas, de uma forma geral, não inclui a escolha da base de conhecimento a ser utilizada. Ou seja, cada trabalho já pressupõe um conjunto fixo de bases de conhecimento a ser utilizado. A diferença para o nosso trabalho é que incluímos uma etapa anterior onde o usuário pode definir a base de dados que deseja utilizar.

[1] www.alchemyapi.com
[2] http://www.opencalais.com/
[3] http://www.zemanta.com/

4 SELEÇÃO DA BASE DE DADOS EM LINKED DATA

A grande diferença na metodologia que adotamos com relação à adotada por outras ferramentas é a seleção da base de dados em Linked Data a ser utilizada. Esta etapa tem grande importância em nossa abordagem pois permite ao usuário um maior controle sobre seus dados.

Apesar de trazer maior liberdade de escolha, a aplicação de nossa metodologia requer que um conjunto de especificações sejam atendidas. Ao componente responsável pela avaliação e seleção das bases de dados aptas a anotar um texto em linguagem natural damos o nome de **Seleção de Base de Dados em Linked Data**. Este componente recebe como entrada o texto que se quer anotar e um conjunto de bases de dados em Linked Data e tem como saída esperada o conjunto de bases de dados aptas a anotar o texto (Figura 4.1).

O método utilizado no componente de Seleção da Base de Dados em Linked Data avalia as bases de dados escolhidas pelo usuário a fim de verificar se elas realmente estão no formato de Linked Data e se possuem um nível mínimo de qualidade para que sejam utilizadas na Resolução de Entidades Nomeadas.

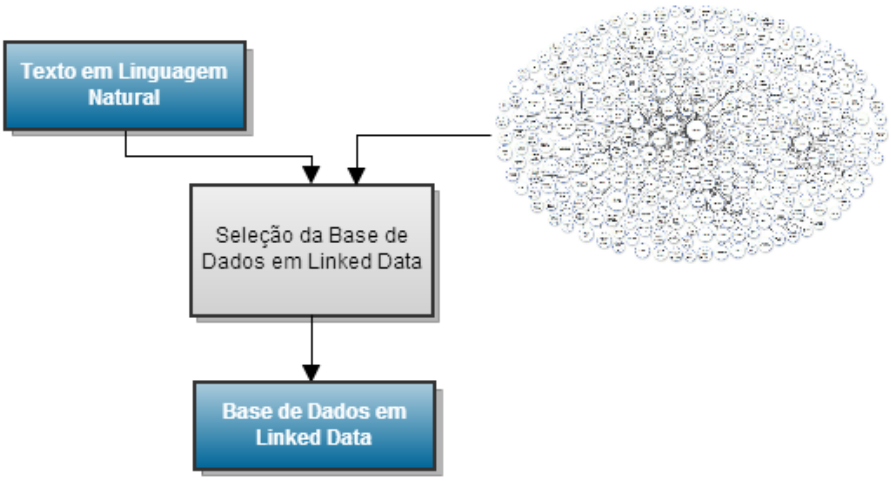


Figura 4.1: Entradas e Saídas esperadas para o componente de Seleção da Base de Dados em Linked Data

Na seção 4.1 explicamos o funcionamento do método de avaliação com as dimensões de qualidade e seus respectivos indicadores. Além disso, delimitamos as características mínimas para que uma base de dados em Linked Data possa ser utilizada junto à nossa metodologia. Na seção seguinte (Seção 4.2) avaliamos um conjunto de bases de dados, de onde serão escolhidas as bases em Linked Data a serem utilizadas para avaliação dos outros componentes de nosso trabalho. Por último, na seção 4.3, levantamos os trabalhos relacionados ao processo de avaliação de qualidade de bases em Linked Data.

4.1 Método

Para que a base de dados possa ser utilizada para o processo de Resolução de Entidades Nomeadas devemos atentar para um conjunto de fatores:

- A base deve ter o seu contexto alinhado ao contexto expresso no texto;

- A base deve descrever um conjunto de Entidades Nomeadas;
- A base deve seguir as principais premissas de Linked Data.

Por mesmo contexto, entendemos que a base de dados deve conter a representação de entidades que atuem no mesmo domínio tratado pelo texto. Se o texto trata de música, falando de grupos musicais e seus trabalhos, a base deve conter representações destas entidades. O mesmo se dá com filmes, termos de biologia, eventos, etc.

Da mesma forma, a base de dados deve representar um conjunto de Entidades Nomeadas, visto que nosso objetivo é a resolução destas, além de explicitar seus nomes a fim de que possa ser feita a correlação com o texto. De outra forma, não seria reconhecida nenhuma menção a Entidades Nomeadas no texto visto que seus nomes não são previamente conhecidos.

Por último, nossa abordagem visa a utilização de bases de dados em Linked Data, logo é necessário que as bases apresentem um conjunto de requisitos mínimos que caracterizam o uso de Linked Data.

Nosso método visa o levantamento de dimensões de qualidade a fim de verificar todos estes itens necessários à aplicadação de nossa metodologia. Cada dimensão possui um conjunto de indicadores que permitem a mensuração da qualidade da base de dados.

As dimensões de qualidade foram escolhidas com base no trabalho de Flemming e Hartig (2010) e são as seguintes:

- Acessibilidade

- Compreensibilidade
- Validade dos Documentos
- Riqueza nos Dados

Um resumo com todas as dimensões de qualidade com seus respectivos indicadores pode ser visto no Quadro 4.1. As características de cada dimensão serão tratadas em detalhes nas subseções seguintes.

Quadro 4.1: Dimensões e Indicadores para avaliação da qualidade de bases de dados em Linked Data para uso na Resolução de Entidades Nomeadas

Dimensão	Indicador de Qualidade
Acessibilidade	Uso de URIs derreferenciáveis
	Disponibilização da base em arquivo RDF
	Disponibilização de SPARQL Endpoint
Compreensibilidade	Existência de descrição dos vocabulários utilizados
	Uso de rdfs:label para nomes de classes e propriedades
	Uso de rdfs:label para nomes de recursos
	Uso de rdfs:comment na descrição de classes e propriedades
	Presença de metadados sobre a base de dados
Validade dos Documentos	Ausência de erros de sintaxe
	Uso exclusivo de classes e propriedades previamente definidas
	Uso de todas as classes e propriedades previamente definidas
Riqueza nos Dados	Número de relacionamentos de domínio por classe
	Número de classes que identificam Entidades Nomeadas

4.1.1 Accessibilidade

O primeiro critério para análise de uma base de dados, de qualquer tipo, é a sua capacidade de prover acesso a seus dados. A acessibilidade mede a capacidade que

a base de dados tem de ser acessada tanto por máquina quanto por seres humanos. Nesta dimensão não estamos avaliando a Usabilidade dos dados mas sim se eles podem ser extraídos e acessados automaticamente ou manualmente. Os indicadores que utilizamos para medir esta dimensão em bases em Linked Data são: uso de URIs derreferenciáveis, disponibilização dos dados em arquivos RDF, e disponibilização de um SPARQL Endpoint.

A forma mais simples de acesso a uma base de dados em Linked Data se dá através da disponibilização direta dos arquivos RDF que compõem a base. Sendo assim, utilizamos a ***disponibilização dos dados em arquivos RDF*** como um indicador que informa se a base de dados pode ser utilizada.

Outra forma um pouco mais sofisticada de acesso a bases em Linked Data se dá através do uso de consultas SPARQL. Segundo Bizer, Heath e Berners-Lee (2009), toda base no formato de Linked Data deve utilizar os formatos padrão RDF, para descrição dos dados, e SPARQL, para consulta. Sendo assim, a ***disponibilização de um SPARQL Endpoint*** é mais um dos indicadores de qualidade no acesso à base.

Por último, outra característica de bases de dados em Linked Data, segundo (BERNERS-LEE, 2006), é o uso de HTTP URIs para que pessoas possam visualizar mais informações sobre os recursos da base de dados. Aos recursos em Linked Data cuja URI pode ser acessada via protocolo HTTP damos o nome de URIs derreferenciáveis. Sendo assim, nosso terceiro indicador de qualidade para a dimensão de Acessibilidade foi o ***Uso de URIs derreferenciáveis***.

4.1.2 Compreensibilidade

Não adianta uma base permitir o acesso a seus dados se eles não podem ser compreendidos. A segunda dimensão avaliada é a Compreensibilidade. Esta dimensão visa verificar se o significado dos dados, também conhecidos como metadados, podem ser acessados, lidos e compreendidos por seres humanos. Para a tarefa de Resolução de Entidades Nomeadas, em especial, é indispensável que haja compreensão dos dados utilizados para representar uma Entidade Nomeada na base em Linked Data. Sem isso, o próprio processo de Resolução fica comprometido visto que um ser humano não pode descobrir que Entidade Nomeada está sendo efetivamente relacionada ao texto.

Os indicadores utilizados para medir a qualidade na dimensão de Compreensibilidade foram: existência de descrição dos vocabulários utilizados; uso de *rdfs:label* para nomes de classes e propriedades; uso de *rdfs:label* para nomes de recursos; uso de *rdfs:comment* na descrição de classes e propriedades; e presença de metadados sobre a base de dados.

Toda base de dados em Linked Data possui um esquema formado pelas classes e propriedades que utiliza. Este esquema pode ter ou não um vocabulário ou ontologia associado a fim de descrever o significado das classes e propriedades utilizadas. Este tipo de informação é de grande relevância para nossa tarefa, pois facilita o reconhecimento de classes que identificam Entidades Nomeadas e de propriedades que descrevem seus nomes. Por este motivo, um dos indicadores que utilizamos foi a ***existência de descrição dos vocabulários utilizados***.

Mesmo que a base de dados não utilize vocabulários ou ontologias, existem outras formas para identificar uma classe ou propriedade. Uma destas formas é a utilização da propriedade *rdfs:label* para indicar o nome das classes e propriedades e o uso de

rdfs:comment para dar uma descrição às mesmas. Sendo assim, utilizamos o ***uso de rdfs:label para nomes de classes e propriedades*** e ***uso de rdfs:comment na descrição de classes e propriedades*** como indicadores da dimensão de Compreensibilidade.

Além das classes e propriedades, necessitamos obter informações sobre os recursos. A propriedade *rdfs:label* é uma das propriedades padrão para descrição de recursos em um formato legível por humanos. Sendo assim, o indicador de ***uso de rdfs:label para nomes de recursos*** foi utilizado em nosso processo de avaliação de qualidade como mais um fator a identificar o nível de Compreensibilidade dos dados da base.

Por último, precisamos de informações sobre a própria base de dados. Necessitamos saber qual o tema da base sem precisar acessar os dados para isso, além de precisarmos de informações sobre o endereço do SPARQL Endpoint ou dos arquivos RDF ou mesmo quais as classes e propriedades presentes na base. Sendo assim, precisamos dos metadados da própria base em Linked Data. O indicador de ***presença de metadados sobre a base de dados*** verifica esta característica.

4.1.3 Validade dos Documentos

Mesmo que a base de dados seja acessível, precisamos ser capazes de processar os dados disponibilizados por ela. Neste sentido, a dimensão de Validade dos Documentos é responsável por medir a capacidade de processamento dos dados em uma base em Linked Data. Seus indicadores são: a ausência de erros de sintaxe; o uso exclusivo de classes e propriedades previamente definidas; e o uso de todas as classes e propriedades previamente definidas.

O indicador de ***ausência de erros de sintaxe*** indica se os arquivos RDF da base

estão corretamente escritos e podem ser processados por um leitor de RDF. Se este indicador mostra que a sintaxe está incorreta então a base não pode ser processada a menos que o erro seja consertado.

Os outros dois indicadores que verificam o ***uso exclusivo de classes e propriedades previamente definidas*** e o ***uso de todas as classes e propriedades previamente definidas*** são importantes principalmente no processamento de consultas SPARQL. Uma consulta SPARQL requer que a estrutura da base de dados seja previamente conhecida. Sendo assim, as bases de dados em Linked Data utilizam vocabulários ou conjunto de vocabulários previamente definidos por seus criadores para descrição dos dados. Estes vocabulários são então posteriormente utilizados como base para a construção das consultas SPARQL. Por este motivo, o uso de classes e propriedades que não constam em nenhum dos vocabulários previamente definidos pode resultar em respostas indesejadas às consultas, assim como a não utilização de propriedades e classes já definidas pode resultar em respostas vazias.

4.1.4 Riqueza nos Dados

Esta última dimensão visa a utilização da base de dados em Linked Data como base de conhecimento para a Resolução de Entidades Nomeadas.

A Riqueza nos Dados é a dimensão que indica se a base de dados em Linked Data tem a quantidade e o tipo de dados necessários para sua aplicação como base de conhecimento. Quanto mais detalhados e completos são os dados expressos na base, melhor a avaliação adquirida nesta dimensão de qualidade.

Os indicadores de qualidade para a dimensão de Riqueza nos Dados são: o número de relacionamentos de domínio por classe; e o número de classes que identificam

Entidades Nomeadas.

Se a base não possui a descrição de Entidades Nomeadas então ela não pode ser utilizada para a tarefa que estamos visando trabalhar. O indicador do ***número de classes que identificam Entidades Nomeadas*** é responsável por apontar o quão viável é a utilização da base de dados com nossa tarefa.

Além das classes, temos de verificar o número de relacionamentos de domínio existentes na base, ou seja, o número de relacionamentos que expressam o contexto de existência de uma entidade, excetuando relacionamentos que indicam superclasse, subclasse e equivalência. Este tipo de informação será utilizada como base para o processo de Desambiguação (Capítulo 8). Portanto, o indicador do ***número de relacionamentos de domínio por classe*** indica se a base de dados pode ser utilizada junto ao nosso processo de Desambiguação.

4.2 Avaliação

O componente de Seleção da base de dados em Linked Data visa identificar que bases podem ser utilizadas para a tarefa de Resolução de Entidades Nomeadas. Além disso, a base de dados escolhida deve pertencer ao mesmo contexto expresso pelo texto (música, filmes, biologia, etc). Nesta seção faremos a avaliação de um conjunto de bases de dados em Linked Data disponíveis na nuvem de LOD.

Primeiro explicaremos o processo de avaliação adotado e, a seguir, trataremos da avaliação propriamente dita.

4.2.1 Processo de Avaliação

Grande parte dos indicadores de nosso método de avaliação contam apenas com resultados do tipo Sim/Não. Porém, para chegar à conclusão correta, precisamos realizar uma análise minuciosa das diversas características necessárias para o preenchimento de cada indicador. O procedimento utilizado para aquisição das informações necessárias à avaliação seguiu cinco passos, a saber:

1. Busca por informações relevantes no sítio web do projeto de criação da base.
2. Busca por informações adicionais no sítio web do CKAN[1].
3. Busca pelo sítio web dos vocabulários utilizados para descrição dos dados.
4. Extração de informações adicionais através de consultas SPARQL à base de dados.
5. Acesso a um conjunto de URIs utilizadas como identificadores.

No passo 1 recolhemos o máximo de informação possível do sítio web de cada projeto a fim de saber sobre os endereços para acesso aos dados e a existência de metadados. No passo 2 procuramos por informações adicionais no sítio web CKAN. Este sítio é responsável por um catálogo de bases de dados abertos, onde todas as bases representadas na imagem da nuvem de LOD (Figura 1.1 - Capítulo 1) apresentam suas informações.

No passo 3 buscamos por informações de classes, propriedades e informações sobre seu significado. Além disso, verificamos qualquer informação sobre a presença de Entidades Nomeadas e relacionamentos de domínio através da análise do significado

[1] http://thedatahub.org/

de cada item dos vocabulários. No passo 4, utilizamos consultas SPARQL para extrair informações não encontradas nos passos anteriores. Para bases de dados que não disponibilizaram SPARQL Endpoints fizemos o download dos arquivos RDF, quando disponíveis, e geramos um SPARQL Endpoint para a base. No último passo, selecionamos um conjunto de URIs aleatórias de recursos da base a fim de verificar a presença de URIs derreferenciáveis.

Cada indicador de qualidade foi analizado de uma forma diferente dependendo das características de cada base. As bases com o maior número de informações em seu sítio web facilitaram o processo de avaliação enquanto que em outros casos precisamos acessar e analisar um conjunto de dados brutos para chegar à resposta que procurávamos. A forma de avaliação mais geral para cada dimensão será explicada nas quatro subseções abaixo. A seguir, faremos a avaliação de todas as bases de dados selecionadas a fim de que possamos escolher duas delas para nosso estudo de caso e avaliação dos componentes posteriores. Encerramos esta seção com uma análise comparativa que nos auxiliou na escolha destas duas bases de dados.

4.2.1.1 Acessibilidade

As informações necessárias à mensuração da dimensão de acessibilidade foram extraídas do sítio web do projeto, do CKAN e através de uma pequena amostra dos dados presentes na base.

Os endereços e arquivos necessários aos indicadores de *disponibilização dos dados em arquivos RDF* e *disponibilização de um SPARQL Endpoint* foram extraídos do sítio web do projeto e do CKAN. Para cada endereço de SPARQL Endpoint disponível, realizamos uma consulta simples a fim de verificar se o Endpoint realmente estava funcional.

No caso do indicador de *uso de URIs derreferenciáveis* extraímos um pequeno grupo de recursos presentes na base de dados e verificamos se as URIs são derreferenciáveis, ou seja, se a URI retorna mais informações sobre o recurso quando acessada diretamente pelo browser.

4.2.1.2 Compreensibilidade

As respostas para os indicadores da dimensão de Compreensibilidade podem ser extraídas de diversos locais dependendo das características de cada base de dados.

As informações para avaliação do indicador de *existência de descrição dos vocabulários utilizados* foram extraídas do sítio do projeto, do CKAN ou mesmo através do início da URL das classes e propriedades existentes na base de dados. A partir do conhecimento do endereço dos vocabulários, acessamo-los e verificamos se havia alguma descrição associada aos itens descritos pelos mesmos.

A presença de *rdfs:label* e *rdfs:comment* para os indicadores de *uso de rdfs:label para nomes de classes e propriedades*, *uso de rdfs:label para nomes de recursos* e *uso de rdfs:comment na descrição de classes e propriedades* foi verificada acessando diretamente os dados. Realizamos consultas SPARQL verificando a presença de cada uma destas propriedades em todas as classes e propriedades descritas na base. Da mesma forma, realizamos a verificação manual de um conjunto de recursos de cada classe a fim de verificar o uso de *rdfs:label* para descrição dos recursos.

Por último, procuramos pela *presença de metadados sobre a base de dados* diretamente no sítio web e no CKAN. Buscamos primeiro por metadados descritos utilizando os descritores DCAT (Data Catalog Vocabulary)[2] e VoiD (Vocabulary

[2]http://www.w3.org/2011/gld/wiki/Data_Catalog_Vocabulary

of Interlinked Datasets)[3]. A seguir, verificamos maiores informações no texto do próprio sítio web e do CKAN.

4.2.1.3 Validade dos Documentos

Para avaliar cada indicador da dimensão de Validade dos Documentos utilizamos todas as fontes disponíveis.

Primeiro, utilizamos a informação do endereço dos arquivos RDF e do SPARQL Endpoint. A partir dos arquivos RDF verificamos o indicador de *ausência de erros de sintaxe* carregando o arquivo numa base de triplas e verificando se o processo retorna algum erro. A seguir, de posse dos vocabulários usados para descrição dos dados na base, utilizamos consultas SPARQL para verificar o *uso exclusivo de classes e propriedades previamente definidas*, além do *uso de todas as classes e propriedades previamente definidas*.

4.2.1.4 Riqueza nos Dados

Os indicadores da dimensão de Riqueza nos Dados sofreram um processo de análise manual mais minucioso que os das outras dimensões. Para estes indicadores precisamos buscar pelo significado de todas as classes e propriedades, além de selecionar uma pequena amostra de recursos em cada classe para realização de análise manual.

Analisamos todas as propriedades existentes na base a fim de verificar aquelas que se referem a relacionamentos que indicam o contexto de existência das entidades que descrevem. Por relacionamento, excluímos todas as propriedades que significam

[3]http://vocab.deri.ie/void/

relacionamentos mas cujos valores eram literais e não recursos. Assim, pudemos definir o *número de relacionamentos de domínio por classe*.

A partir do significado das classes e de uma pequena amostra de recursos em cada uma, verificamos o *número de classes que identificam Entidades Nomeadas*. Foram consideradas identificando Entidades Nomeadas apenas as classes cujos recursos possuem uma propriedade representando seu nome.

4.2.2 Avaliação das Bases de Dados

Como estudo de caso para a avaliação dos componentes de nossa metodologia, selecionamos um conjunto de textos em dois domínios: música e filmes. Como as bases de dados tem de estar relacionadas aos textos a serem processados, selecionamos bases da nuvem de LOD em um destes domínios. As bases de dados foram selecionadas a partir dos sítios web que utilizam como fonte de dados pois tratam-se de sítios amplamente conhecidos nos domínios de conhecimento escolhidos. O conjunto de bases utilizado bem como o domínio que abrangem e os sítios web utilizados como fonte podem ser encontrados no Quadro 4.2.

Quadro 4.2: Bases de Dados em Linked Data selecionadas para avaliação da viabilidade de uso em Resolução de Entidades Nomeadas

Domínio	Base de Dados em Linked Data	Sítio Web
Filmes	Linked Movie Database	IMDB
Música	BBC Music	BBC Music
	Data Incubator: Music Brainz	Music Brainz
	Linked Brainz	
	Music Brainz (DBTune.org)	
	Magnatune RDF Server (DBTune.org)	Magnatune
	Jamendo RDF Server (DBTune.org)	Jamendo

Como os textos escolhidos para anotação foram biografias de músicos e textos sobre

filmes levantamos bases nos domínios de música e cinema.

4.2.2.1 Linked Movie Database

A Linked Movie Database (Linked MDB)(HASSANZADEH; CONSENS, 2009) é uma base de dados na área de produção cinematográfica. Ela possui ligações com diversas outras bases da nuvem de LOD e com diferentes sítios web, dentre eles o Internet Movie Database (IMDB)[4], um sítio web especializado em cinema e televisão.

O projeto tem um sítio web próprio e também possui informações disponíveis no CKAN. O endereço para o SPARQL Endpoint pode ser encontrado no sítio do projeto enquanto o arquivo RDF só é encontrado através do CKAN. Através do sítio web pudemos perceber que há uma série de URIs derreferenciáveis permitindo navegação através dos relacionamentos existentes entre os recursos da base. Estas observações nos dão as respostas para os três indicadores da dimensão de ***Acessibilidade***.

Analisando a dimensão de ***Compreensibilidade*** percebemos que a base utiliza um conjunto de descritores próprio (com o namespace http://data.linkedmdb.org/resource/movie) fazendo uso apenas da classe Person e de algumas propriedades descritas pelo vocabulário Friend of a Friend (FOAF) (BRICKLEY; MILLER, 2010), além de outras propriedades descritas por RDF (HAYES, 2004), RDFS (BRICKLEY; GUHA, 2004), Dublin Core (DCMI, 2012) e OWL (Web Ontology Language) (PATEL-SCHNEIDER; HAYES; HORROCKS, 2004). Uma visão de todas as classes e propriedades utilizadas na descrição da base de dados pode ser vista no Apêndice A (Quadro A.1).

Buscando o significado das classes e propriedades, notamos que não há descrição

[4]http://www.imdb.com/

disponível no sítio web do projeto, no endereço do namespace do descritor usado pela base ou mesmo expresso na base através do uso de *rdfs:comment*. As classes e propriedades não contam com seus nomes ou descrições expressos na base com *rdfs:label* e *rdfs:comment*.

Ao contrário das classes, a Linked Movie Database utiliza a propriedade *rdfs:label* em todos os recursos, porém as instâncias da classe "Film" são as únicas onde o valor de *rdfs:label* é realmente o nome das Entidades Nomeadas representadas pelos recursos. Em outras classes, o valor de *rdfs:label* é constituído pelo nome das Entidades Nomeadas mais a classe a qual elas pertencem.

Finalizando a avaliação da dimensão de Compreensibilidade, verificamos que o sítio web da Linked Movie Database não oferece nenhum tipo de metadado em formato processável por máquina, porém um conjunto de informações relevantes para o acesso aos dados encontra-se disponível em linguagem natural. O sítio do projeto fornece informações sobre os endereços a serem utilizados para acesso aos dados bem como exemplos de recursos presentes na base. O sítio não fornece, porém, nenhuma informação sobre os vocabulários utilizados para a descrição dos dados tornando impossível a verificação de alguns dos indicadores utilizados como medida de ***Validade dos Documentos***.

O sítio web provê ainda o acesso aos dados mas não os disponibiliza em arquivo RDF. Assim, utilizamos o arquivo RDF disponível no CKAN para análise da sintaxe das triplas RDF. Durante o processo de carregamento dos dados na base de triplas constatamos que há uma série de problemas na sintaxe, principalmente em triplas da classe "Interlink". Esta classe é responsável por registrar links com outras bases em Linked Data tais como DBPedia e YAGO, por exemplo. O erro foi constatado na presença de aspas na URI dos recursos das bases as quais estavam sendo ligadas, gerando assim um problema na hora de realizar o processamento do arquivo.

A partir da base com todas as triplas RDF que não apresentaram erro de sintaxe, pudemos identificar as propriedades que representam relacionamentos de domínio e as classes que identificam Entidades Nomeadas para avaliação da dimensão de ***Riqueza nos Dados***. Foram reconhecidas 34 classes identificando Entidades Nomeadas dentre as 53 classes presentes na base, possuindo em média 0,96 relacionamentos de domínio por classe.

Outras características importantes que identificamos mas que não influenciaram diretamente na avaliação foram a duplicação de identificadores (URIs) para itens diferentes e a existência de recursos representando relacionamentos ternários.

Durante a análise de URIs percebemos que algumas delas são utilizadas tanto na identificação de classes como de propriedades. Um exemplo claro disto é o identificador http://data.linkedmdb.org/resource/movie/actor que é usado tanto para identificar a classe "Actor" quanto para identificar uma propriedade da classe "Film".

A base conta ainda com um conjunto de classes que se referem não a entidades mas sim a relacionamentos ternários. Um exemplo deste tipo de classe é a "Performance", que representa um relacionamento entre as classes "Actor", "Character" e "Film". Neste tipo de classe o valor de *rdfs:label* utilizado nos recursos é um valor genérico desprovido de significado para um humano, o que entra em contraste com o objetivo no uso de *rdfs:label*. Os valores de *rdfs:label* para recursos desta classe, por exemplo, são do tipo "performance #1", ou seja, apenas uma numeração que tem como origem provável a sua utilização como identificador para o recurso em uma outra base de dados utilizada como fonte das informações.

Os dados gerais da Linked Movie Database podem ser encontrados no Quadro 4.3.

Quadro 4.3: Dados Gerais da Linked Movie Database

Sítio Web	http://www.linkedmdb.org/
SPARQL Endpoint	http://data.linkedmdb.org/sparql
Namespace	http://data.linkedmdb.org/resource/movie/
Descritores	Dublin Core
	Friend of a Friend
	RDF
	RDFS
	OWL
Número de Classes	53

4.2.2.2 BBC Music

A BBC Music é uma base de dados criada a partir das anotações em RDFa nas páginas do sítio web da BBC Music[5]. A BBC é uma empresa de mídia do Reino Unido onde a BBC Music é a sua vertente ligada ao domínio da música. A BBC Music trata do relato de shows e eventos relacionados a artistas e grupos musicais, além da construção de resenhas sobre seus trabalhos.

Os dados da BBC Music são disponibilizados embutidos em páginas HTML, portanto sua utilização só é possível através da utilização de scripts para extração das informações ou através de serviços que disponibilizem as triplas de maneira separada. Para isso, a própria BBC disponibiliza uma API (Application Programming Interface) para desenvolvedores de aplicações, além de dois SPARQL Endpoints através das plataformas disponibilizadas pelas empresas OpenLink e Talis.

Apesar das formas de acesso, a BBC não disponibiliza um sítio web para a divulgação de metadados sobre a base, vocabulários utilizados ou os assuntos tratados pelos dados, por exemplo. Da mesma forma, o CKAN provê apenas informações sobre acesso aos dados e um exemplo de recurso RDF presente na base.

[5]http://www.bbc.co.uk/music

Devido à pouca informação disponível tentamos utilizar os SPARQL Endpoints como fonte para nossas análises. É importante notar que nossa avaliação ficou completamente prejudicada por questões de *timeout* (tempo de processamento excedido) na realização de consultas a ambos os Endpoints. Mesmo em consultas simples com tempo limite para *timetout* de 150 segundos não conseguimos obter uma resposta. Por este motivo, não pudemos concluir nossas análises, podendo apenas fazer consultas pontuais como a necessária para a verificação de URIs derreferenciáveis. Os dados gerais da BBC Music são apresentados no Quadro 4.4

Durante a verificação da existência de URIs derreferenciáveis notamos que grande parte das URIs descritas pela BBC são, na verdade, URIs sob o domínio de outras bases de dados tais quais a Freebase[6] e a Geo Linked Data[7]. De toda forma, todas as URIs parecem ser derreferenciáveis visto que as descritas com o namespace http://api.talis.com/stores/bbc-backstage/ retornam informações sobre o recurso quando acessadas e as que atendem pelo namespace http://www.bbc.co.uk/programmes/ redirecionam para páginas web que falam sobre o recurso em questão.

Quadro 4.4: Dados Gerais da BBC Music

Página no CKAN	http://ckan.net/dataset/bbc-music
SPARQL Endpoint	http://lod.openlinksw.com/sparql/
	http://api.talis.com/stores/bbc-backstage/services/sparql
Namespace	http://api.talis.com/stores/bbc-backstage/
	http://www.bbc.co.uk/programmes/

4.2.2.3 Data Incubator: Music Brainz

A Data Incubator: Music Brainz é uma base de dados criada a partir da conversão dos dados do projeto Music Brainz de uma base relacional para o formato de triplas.

[6] http://rdf.freebase.com/
[7] http://geo.linkeddata.es

O projeto Music Brainz[8] visa a criação de uma base de dados no domínio da música com informações sobre artistas e seus trabalhos. O projeto está disponível na Web e foi criado de forma colaborativa.

A Data Incubator possui, segundo o CKAN, um sítio web próprio para o projeto, mas este sítio esteve indisponível em diversas tentativas de acesso que fizemos por meses seguidos. Sendo assim, tentamos extrair o máximo possível de informação direto do CKAN.

Segundo as informações encontradas, a base de dados disponibiliza um SPARQL Endpoint utilizando a plataforma Talis, o que nos trouxe o mesmo problema de *timeout* relatado com o uso da BBC Music. Por este motivo nossa avaliação ficou bastante restrita.

Ainda no CKAN, verificamos que há um link para a descrição da base de dados utilizando VoiD mas, ao acessarmos o link, a descrição também estava indisponível.

Partindo à análise utilizando o Endpoint verificamos uma característica similar à base anterior que é a descrição de recursos identificados com URIs as quais a base de dados não é responsável pelo namespace. Dentre estas URIs pudemos identificar por exemplo recursos descritos pela DBPedia que são derreferenciáveis e recursos usando URIs próprias da base de dados (sob o namespace http://musicbrainz.dataincubator .org) que não são derreferenciáveis. Algumas informações sobre a Data Incubator: Music Brainz estão presentes no Quadro 4.5.

Quadro 4.5: Dados Gerais da Data Incubator: Music Brainz

Sítio Web	http://musicbrainz.dataincubator.org/
SPARQL Endpoint	http://api.talis.com/stores/musicbrainz/services/sparql
Namespace	http://musicbrainz.dataincubator.org/

[8]http://musicbrainz.org/

4.2.2.4 Linked Brainz

A Linked Brainz é uma base que também foi criada a partir do projeto Music Brainz. A base de dados possui um sítio web próprio além de uma wiki com diversas informações sobre a forma como o projeto tem sido executado. O Quadro 4.6 apresenta algumas informações gerais sobre esta base de dados.

Quadro 4.6: Dados Gerais da Linked Brainz

Sítio Web	http://linkedbrainz.c4dmpresents.org
SPARQL Endpoint	http://linkedbrainz.c4dmpresents.org/snorql/ http://linkedbrainz.c4dmpresents.org/talis/snorql/
Namespace	http://data.linkedmdb.org/resource/movie/
Descritores	Dublin Core
	Friend of a Friend
	Music Ontology
	Open Vocab
	RDF
	RDFS
	SKOS
Número de Classes	13

Avaliando a dimensão de ***Acessibilidade*** verificamos que os dados da Linked Brainz podem ser acessados tanto via SPARQL Endpoint quanto através do download das triplas RDF. O sítio web do projeto indica dois SPARQL Endpoints, um que se encontra indisponível e outro que permite consulta aos dados sem grandes problemas com *timeout*. Apesar dos SPARQL Endpoints, decidimos realizar as análises a partir do arquivo RDF com todas as triplas existentes na base.

As URIs descritas pela Linked Brainz adotam a URI http://musicbrainz.org/ como *namespace* fazendo com que todas as URIs sejam derreferenciáveis ao apontarem para a sítio web do projeto Music Brainz. Além deste namespace, a base também descreve entidades sob o namespace http://ontologi.es/place/ que também são derreferenciáveis.

Para avaliação da dimensão de ***Compreensibilidade***, verificamos que no sítio web do projeto não há a divulgação de metadados sobre a base nem sobre os vocabulários utilizados para descrição dos dados. Podemos encontrar as classes e propriedades utilizadas apenas se analisarmos diretamente o arquivo de regras para transformação dos dados de sua fonte original para a base em Linked Data ou verificando, via consulta SPARQL, os próprios recursos que descrevem as classes e propriedades.

Os vocabulários utilizados pela base, e verificados por meio do nosso processo de avaliação, são: Music Ontology (RAIMOND et al., 2007), RDF, SKOS (Simple Knowledge Organization System) (MILES; BECHHOFER, 2009), Open Vocab[9], RDFS, Dublin Core e Friend of a Friend.

Na avaliação do significado das classes verificamos que apenas parte delas (11 em 13) tem seus nomes descritos utilizando *rdfs:label* e nenhuma utiliza *rdfs:comment*. Percebemos ainda que as classes que utilizam rdfs:label são as únicas que realmente estão descritas na base de dados como sendo do tipo *owl:Class*, as outras duas, a saber http://purl.org/ontology/mo/SoloMusicArtist e http://purl.org/ontology/mo/MusicGroup são utilizadas como valor da propriedade *rdf:type* mas não estão efetivamente descritas na base de dados.

Os recursos que utilizam *rdfs:label* para a descrição de seus nomes pertencem apenas a um conjunto restrito de classes (vide Quadro B.1 no Apêndice B), as outras, quando representando Entidades Nomeadas, utilizam propriedades alternativas para descrição de seus nomes.

Passando à análise dos indicadores da dimensão de ***Validade dos Documentos*** começamos por não ter disponíveis dados sobre quais os vocabulários e, consequentemente, que classes e propriedades deveriam ser esperadas na base, sendo assim, dois

[9] http://open.vocab.org/

dos indicadores não puderam ser avaliados. Com relação ao indicador que informa a ausência de erros de sintaxe, efetuamos o carregamento do arquivo RDF disponível no sítio web do projeto no gerenciador de triplas OpenLink Virtuoso sem nenhum problema.

Por último, passamos à análise da ***Riqueza nos Dados***, quando verificamos quais as classes que se referem a Entidades Nomeadas onde a relação completa pode ser vista no Qaudro B.1. Das 13 classes avaliadas verificamos que 10 delas identificam Entidades Nomeadas enquanto o número de relacionamentos de domínio é, em média, de 1,07 relacionamentos de domínio por classe. Durante esta avaliação verificamos que a propriedade http://purl.org/ontology/mo/publishing_location tem como valor URIs que identificam recursos originalmente externos. Apesar disso, estes mesmos recursos são descritos (de forma duplicada) internamente à base, sendo assim reconhecemos a propriedade como um relacionamento de domínio

4.2.2.5 MusicBrainz (DBTune.org)

A MusicBrainz é uma das bases disponibilizadas pelo DBTune.org e, assim como as duas bases de dados anteriores, utiliza o sítio web MusicBrainz como fonte de seus dados. Esta base foi gerada utilizando a tecnologia D2R (BIZER; CYGANIAK, 2006) que mapeia os dados de uma base de dados relacional para o formato RDF no momento em que uma consulta é realizada.

O sítio web do projeto disponibiliza o endereço de um SPARQL Endpoint e uma navegação pelos recursos da base. Através da navegação pelos recursos pudemos verificar que as URIs são derreferenciáveis. Por outro lado, o SPARQL Endpoint não funcionou nem mesmo utilizando uma consulta exemplo do próprio Endpoint. Como a base não apresenta o arquivo RDF para download não pudemos fazer qual-

quer outro tipo de avaliação visto que a avaliação manual (navegando pelos links) é trabalhosa e não vai de encontro ao objetivo deste trabalho.

Algumas informações gerais sobre a MusicBrainz (DBTune.org) são apresentadas no Quadro 4.7.

Quadro 4.7: Dados Gerais da Music Brainz (DBTune.org)

Sítio Web	http://dbtune.org/musicbrainz/
SPARQL Endpoint	http://dbtune.org/musicbrainz/sparql

4.2.2.6 Magnatune RDF Server (DBTune.org)

A Magnatune RDF Server é uma base de dados publicada como parte do projeto DBTune.org e tem como fonte de seus dados o sítio web da Magnatune. Magnatune[10] é uma produtora de artistas independentes e tem um sítio web para venda das músicas.

O sítio web da base de dados disponibiliza alguns exemplos de recursos anotados com RDF, bem como o link para o SPARQL Endpoint e para o arquivo RDF de toda a base. Assim como a MusicBrainz (DBTune.org) esta base foi gerada utilizando o D2R para mapeamento dos dados.

Tentamos acessar as informações sobre as classes existentes na base utilizando o SPARQL Endpoint. Como o endereço disponibilizado não permite consultas via browser utilizamos um endereço alternativo, também disponível no sítio web, para realização das consultsa atestando que o Endpoint está funcional.

Para encerrar a avaliação da dimensão de ***Acessibilidade*** optamos por avaliar a

[10]http://magnatune.com/

existência de URIs derreferenciáveis através da navegação pelos recursos dados como exemplo no próprio sítio web da base de dados. Sendo assim, verificamos que as URIs descritas pela base retornam informações adicionais quando acessadas via HTTP, logo são derreferenciáveis.

O sítio web da base não disponibiliza informações de metadados e também não explicita quais os vocabulários utilizados para anotação dos dados. Sendo assim, alguns indicadores de qualidade (***Validade nos Documentos***) não puderam ser avaliados por falta de informações.

Partimos então à avaliação da ***Compreensibilidade*** através de consultas SPARQL. As classes e propriedades utilizadas podem ser vistas no Apêndice C (Quadro C.1). Os vocabulários utilizados para descrição dos dados são: RDF, RDFS, Music Ontology, Dublin Core, Friend of a Friend, Time Ontology[11], Timeline Ontology[12], BIO[13] e Event Ontology[14].

Nenhuma das classes utilizadas são descritas como recursos RDF na própria base, assim, não apresentam rdfs:label ou rdfs:comment. No caso das instâncias, todas as classes tem suas instâncias descritas utilizando rdfs:label porém nem sempre o valor desta propriedade possibilita a compreensão do que o recurso representa. Um exemplo claro é o uso de rdfs:label para instâncias da classe http://purl.org/NET/c4dm/timeline.owl#RelativeTimeLine onde o valor da propriedade é o texto "timeline" seguido por uma sequência de números. O mesmo ocorre para instâncias das classes http://purl.org/ontology/mo/Performance, http://purl.org/ontology/mo/Signal e http://www.w3.org/TR/owl-time/Interval.

[11] http://www.w3.org/TR/owl-time/
[12] ttp://motools.sourceforge.net/timeline/timeline.html
[13] http://vocab.org/bio/0.1/.html
[14] http://motools.sourceforge.net/event/event.html

Segundo o que pudemos levantar, a Magnatune RDF Server conta com a presença de 7 classes sendo 3 delas classes que identificam Entidades Nomeadas, possuindo em média 1,14 relacionamentos de domínio por classe.

Para finalizar a avaliação da Magnatune efetuamos o download do arquivo RDF disponível no sítio web da base e o carregamos em uma base de triplas, o que não resultou em qualquer erro de sintaxe. O Quadro 4.8 apresenta algumas informações sobre esta base de dados.

Quadro 4.8: Dados Gerais da Magnatune RDF Server (DBTune.org)

Sítio Web	http://dbtune.org/magnatune/
SPARQL Endpoint	http://dbtune.org/magnatune/sparql http://dbtune.org/magnatune/store/user/query
Namespace	http://dbtune.org/magnatune/
Descritores	BIO
	Dublin Core
	Event Ontology
	Friend of a Friend
	Music Ontology
	Open Vocab
	RDF
	RDFS
	SKOS
	Time Ontology
	Timeline Ontology
Número de Classes	7

4.2.2.7 Jamendo RDF Server (DBTune.org)

Mais uma integrante da DBTune.org, Jamendo RDF Server é uma base de dados que tem como fonte para seus dados o sítio web Jamendo[15], cuja principal finalidade é o compartilhamento de músicas sob a licença Creative Commons. Esta base dis-

[15]http://jamendo.com/

ponibiliza um arquivo RDF com todas as triplas da base e um SPARQL Endpoint para consulta aos dados.

Analisando a dimensão de ***Compreensibilidade***, verificamos que nem no sítio web do projeto nem no CKAN são disponibilizados metadados sobre a base ou informações sobre os vocabulários utilizados. Como o SPARQL Endpoint oferece um bom tempo de resposta, pensamos em utilizá-lo como fonte para as avaliações, porém, após carregarmos os dados do RDF em uma base de triplas local, verificamos que o conteúdo do RDF é diferente do conteúdo apresentado pelo SPARQL Endpoint. Por este motivo, optamos por utilizar apenas o conteúdo do RDF, visto que ele não muda com o tempo enquanto o conteúdo do Enpoint independe de nosso controle.

A partir de nossa versão local da base, verificamos que não há qualquer erro de sintaxe no arquivo. A próxima etapa foi avaliar se a base efetivamente disponibiliza URIs derreferenciáveis para identificação de seus recursos. Esta análise foi feita a partir de uma pequena amostra das URIs disponibilizadas pelo Endpoint.

Todos os vocabulários utilizados para descrição dos dados têm informações disponíveis na web. Os vocabulários são: Music Ontology, Time Ontology, Timeline Ontology, Friend of a Friend, RDF, RDFS, OWL, Dublin Core e Tag Ontology[16]. Como não há nenhuma informação sobre as classes e propriedades direto na base então não há uso de *rdfs:label* e *rdfs:comment*. Com relação aos recursos, nenhum deles utiliza *rdfs:label* para fornecimento de um nome compreensível por humanos.

Além disso, para avaliação da dimensão de ***Riqueza nos Dados***, verificamos que a base de dados conta com 11 classes onde 3 delas identificam Entidades Nomeadas (Quadro D.1 no Apêndice D). A base conta ainda com 0,54 relacionamentos de domínio, em média, por classe. Durante o levantamento de informações sobre

[16]http://www.holygoat.co.uk/owl/redwood/0.1/tags/

os relacionamentos de domínio verificamos que os recursos que são instâncias da classe http://www.w3.org/2006/time#Interval possuem uma URI como valor da propriedade http://purl.org/NET/c4dm/timeline.owl#onTimeLine sendo que esta URI não é descrita na base de dados.

Algumas informações sobre a Jamendo RDF Server são apresentadas no Quadro 4.9.

Quadro 4.9: Dados Gerais da Jamendo RDF Server (DBTune.org)

Sítio Web	http://dbtune.org/jamendo/
SPARQL Endpoint	http://dbtune.org/jamendo/sparql/
	http://dbtune.org/jamendo/store/user/query/
Namespace	http://dbtune.org/jamendo/
Descritores	Dublin Core
	Friend of a Friend
	Music Ontology
	RDF
	RDFS
	Tag Ontology
	Time Ontology
	Timeline Ontology
	OWL
Número de Classes	11

4.2.3 Análise Comparativa

A partir da avaliação das bases, podemos ter uma visão parcial da atual situação das bases de dados presentes na nuvem de LOD. Esta avaliação indicou algumas dificuldades que deverão ser enfrentadas pelas aplicações que visam consumir os dados destas bases.

A primeira dificuldade reside na acessibilidade das bases atuais. Devido à baixa capacidade de acesso às bases, seja através do download do arquivo RDF, seja através de SPARQL Endpoints que não funcionam corretamente, nem sequer podemos fazer

uso da informação. Esta característica foi frequente em diversas das bases analisadas e prejudicou até mesmo a avaliação das mesmas, onde a dimensão de Validade de Documentos nem sequer foi avaliada.

A segunda dificuldade está na descoberta de quais são os vocabulários utilizados para a anotação dos dados presentes na base. Sem essa informação precisamos de algoritmos que não contem com determinadas classes ou vocabulários, ou que façam mineração nos dados para descobrir as classes e propriedades utilizadas, tal qual fizemos no processo de avaliação.

Outra dificuldade é a diversidade nos SPARQL Endpoints e o recebimento de uma mensagem de tempo de execução excedido como resposta. Bases de dados em Linked Data tendem a ter um número muito grande de triplas, o que inviabiliza o uso de grandes arquivos RDF por cada aplicação consumidora. Sendo assim, a inexistência ou precariedade no funcionamento dos SPARQL Endpoints dificulta em muito a utilização destas bases de dados.

Alguns parâmetros de qualidade que utilizamos não são fundamentais, sobretudo para a criação de aplicações para Resolução de Entidades Nomeadas. Porém, indicadores como a utilização de *rdfs:label* ou a existência de descrição para os vocabulários auxiliam não só no cálculo da precisão das aplicações como também na posterior utilidade dos textos anotados. Sem uma anotação compreensível, o objeto final da aplicação de Resolução de Entidades Nomeadas torna-se indecifrável e consequentemente inútil visto que a anotação tem por objetivo dar significado à entidade que está sendo mencionada no texto.

Por fim, com base nas dimensões de qualidade levantadas, em seus indicadores e nos resultados da avaliação de cada uma das bases de dados (Quadro 4.10), pudemos escolher ao menos duas bases para a experimentação de nossa ferramenta. Para es-

colha das bases, descartamos todas aquelas que não disponibilizam o arquivo RDF pois precisamos de dados que não mudem com o tempo a fim de que nossos experimentos possam ser reproduzidos a qualquer tempo. Sendo assim, as bases melhor avaliadas foram a Linked Movie Database e a Linked Brainz devido ao alto número de classes identificando Entidades Nomeadas e a quantidade de relacionamentos de domínio. Como a Linked Brainz possui um número de triplas extremamente difícil de processar com o computador de que dispomos para este trabalho, decidimos trocá-la pela Jamendo.

4.3 Trabalhos Relacionados

Passado o período inicial de grande entusiasmo pela publicação de novas bases de dados em Linked Data, a comunidade científica resolve agora passar à avaliação das bases quanto à sua qualidade.

Muita discussão tem sido feita na web (FLEMMING; HARTIG, 2010) (FERRIS, 2010) a fim de levantar as principais características que caracterizam uma base de dados de boa qualidade. Uma dessas discussões (FLEMMING; HARTIG, 2010) deu origem às dimensões de Acessibilidade, Compreensibilidade e Validade dos Documentos que utilizamos neste trabalho, além de alguns de seus indicadores. A dimensão de Riqueza nos Dados e seus indicadores, bem como dois indicadores da dimensão de Validade dos Documentos foram idealizados por nós visando especialmente a tarefa de Resolução de Entidades Nomeadas.

Seguindo as discussões na Web, alguns trabalhos (MARTIN, 2012)(HOGAN et al., 2012) listam as características básicas de qualidade de bases de dados em geral enquanto outros (FüRBER; HEPP, 2011) acreditam que a qualidade de uma base de dados só pode ser medida a partir de uma aplicação para os dados.

Neste trabalho fizemos a avaliação de um conjunto de bases de dados em Linked Data visando a realização da tarefa de Resolução de Entidades Nomeadas da melhor forma possível. Como fruto da avaliação, tivemos um conjunto de bases aptas para, com um nível mínimo de qualidade, serem utilizadas para a realização de nossa tarefa.

Quadro 4.10: Quadro Comparativo com os Resultados da Avaliação das Bases de Dados em Linked Data

Indicadores	Linked MDB	BBC Music	Data Inc: Music Brainz	Linked Brainz	Music Brainz	Magna-tune	Ja-men-do
Dimensão: Acessibilidade							
Uso de URIs derreferenciáveis	Sim	Sim	Parcial	Sim	Sim	Sim	Sim
Disponibilização da base em arquivo RDF	Sim	Não	Não	Sim	Não	Sim	Sim
Disponibilização de SPARQL Endpoint	Sim	Sim*	Sim*	Sim	Sim*	Sim	Sim
Dimensão: Compreensibilidade							
Existência de descrição dos vocabulários utilizados	Não	-	-	Sim	-	Sim	Sim
Uso de rdfs:label para nomes de classes e propriedades	Não	-	-	Parcial	-	Não	Não
Uso de rdfs:label para nomes de recursos	Parcial	-	-	Parcial	-	Parcial	Não
Uso de rdfs:comment na descrição de classes e propriedades	Não	-	-	Não	-	Não	Não
Presença de metadados sobre a base de dados	Texto	Não	Não	Texto	Texto	Texto	Texto
Dimensão: Validade dos Documentos							
Ausência de erros de sintaxe	Não	-	-	Sim	-	Sim	Sim
Uso exclusivo de classes e propriedades previamente definidas	-	-	-	-	-	-	-
Uso de todas as classes e propriedades previamente definidas	-	-	-	-	-	-	-
Dimensão: Riqueza nos dados							
Número de relacionamentos de domínio por classe	0,96	-	-	1,07	-	1,14	0,54
Número de classes que identificam Entidades Nomeadas	34	-	-	10	-	3	3
* Apesar de possuir SPARQL Endpoint apresenta timeout para consultas simples.							

5 RECONHECIMENTO DE PROPRIEDADES-NOME

O primeiro passo para a utilização de uma base de dados na Resolução de Entidades Nomeadas é a identificação de que estruturas representam Entidades Nomeadas e seus nomes na base de dados.

Em uma base em Linked Data, todos os atributos de uma entidade são representados através de propriedades específicas. Sendo assim, todo recurso que representa uma Entidade Nomeada deve explicitar seu nome através do uso de um conjunto de propriedades destinadas a este fim. A estas propriedades daremos o nome de ***propriedades-nome***.

As propriedades-nome podem variar de uma base de dados para outra, dependendo apenas dos Vocabulários utilizados. Devido à grande heterogeneidade na descrição de dados em Linked Data e ao tamanho de alguns dos descritores com centenas de classes e propriedades, como é o caso da DBPedia, por exemplo (AUER et al., 2007), o levantamento manual de todas as propriedades-nome pode tornar-se inviável.

Com o objetivo de minimizar o trabalho manual no levantamento de propriedades-nome, desenvolvemos o componente de Reconhecimento de Propriedades-nome. Seu

objetivo é reconhecer automaticamente, através de mineração e análise dos dados, todas as propriedades-nome existentes em uma base de dados em Linked Data. Este componente permite ainda identificar que classes representam Entidades Nomeadas, ou seja, classes que possuem uma ou mais propriedades-nome para cada uma de suas instâncias.

O componente de Reconhecimento de Propriedades-Nome recebe como entrada uma base de dados em Linked Data e retorna como resultado o conjunto de classes que identificam Entidades Nomeadas e as suas respectivas propriedades-nome (Figura 5.1).

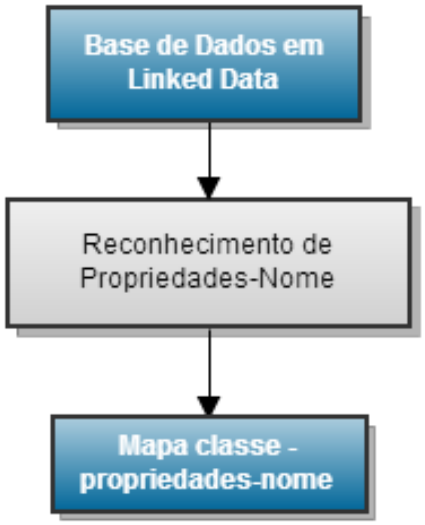


Figura 5.1: Entradas e Saídas esperadas para o componente de Reconhecimento de Propriedades-Nome

Para construção deste componente criamos um método para coleta dos dados e um conjunto de heurísticas para seleção das propriedades-nome. Todos os detalhes de desenvolvimento serão discutidos na seção seguinte e ao final faremos uma avaliação das heurísticas utilizando as bases de dados escolhidas na etapa de Seleção das Bases em Linked Data (Capítulo 4).

5.1 Método

O método desenvolvido para o componente de Reconhecimento de Propriedades-nome tem como objetivo receber o menor número possível de informações por parte do usuário. Dessa forma, temos de descobrir todo o possível sobre a base de dados em Linked Data utilizando apenas seus dados.

Este método utiliza pouquíssima informação semântica visto que a base de dados pode utilizar qualquer tipo de vocabulário. Nem toda base de dados em Linked Data tem explícita em sua semântica qualquer informação que possa nos auxiliar em nossa tarefa, tais quais o nome ou a identificação propriamente dita de que um recurso representa uma Entidade Nomeada. Outra informação que nos poderia ser útil é a indicação de todas as classes que identificam entidades nomeadas mas esta também muda de um vocabulário para outro e mesmo dentre bases que utilizam um mesmo vocabulário.

A única informação que utilizamos acerca do vocabulário utilizado é o fato de que a propriedade *rdfs:label* não é 100% confiável na busca por nomes (PEREIRA; SILVA; VIVACQUA, 2012). Sendo assim, sempre que houver dúvida entre a *rdfs:label* e outra propriedade optamos por utilizar a outra propriedade como propriedade-nome.

Nosso método seleciona todas as propriedades que possuem como valor um literal, analiza um conjunto de características destes literais e utiliza uma heurística para escolher quais das propriedades são propriedades-nome de acordo com a análise feita. As características verificadas para cada valor literal são:

- tipo do literal.
- porcentagem de palavras (tokens) iniciadas por letra maiúscula.

- quantidade de caracteres no texto.

Neste trabalho, apenas cadeias de caracteres são consideradas nomes, por isso a verificação do tipo do literal é importante. Além disso, nomes são identificados por nomes próprios, ou seja, cadeias de caracteres iniciados por letra maiúscula. Apesar de parecer simples, existem ainda algumas considerações a serem feitas. Se verificamos apenas a primeira letra do valor da propriedade, estamos igualando nomes, siglas e parágrafos inteiros pois todos eles iniciam-se por uma letra maiúscula. Por este motivo analisamos a porcentagem de palavras (tokens) iniciadas por letra maiúscula. Este valor é plenamente configurável, portanto neste trabalho iremos considerar apenas os textos cujo valor desta característica seja de pelo menos 50% a fim de excluir grande parte dos textos que não se referem a nomes ou siglas.

A terceira característica verificada permite que possamos excluir siglas, quando indesejadas, e textos muito grandes, como veremos mais à frente.

Para o método de Reconhecimento de Propriedades-nome utilizamos o Algoritmo 1 modificando apenas as heurísticas utilizadas. Neste algoritmo p_{valor} representa o literal que está presente como objeto para a propriedade p.

Para decisão de quais são as propriedades-nome desenvolvemos uma heurística inicial e, a partir dela, desenvolvemos outras três através de pequenos ajustes. As heurísticas criadas são: Ingênua, Ingênua Parametrizada, Multivalorada e Multivalorada com Threshold. Cada uma possui um valor de score e um critério associados que indicarão a pontuação de uma propriedade e se ela deve ser identificada como uma propriedade-nome ou não.

Algoritmo 1 Reconhecimento de Propriedades-nome

LD $\leftarrow$ base de dados em Linked Data
criterio$_{heuristica}$ $\leftarrow$ critério para aceitar uma propriedade como propriedade-nome
listaClasse $\leftarrow$ $\{classe_1, classe_2, \ldots, classe_n\}$, $\forall classe_i \in$ LD
for cada classe c $\in$ listaClasse **do**
  listaPropriedade $\leftarrow$ $\{propriedade_1, \ldots, propriedade_m\}$, $\forall propriedade_i$ que descreve c
  for cada propriedade p $\in$ listaPropriedade **do**
    colete todas as características de p_{valor}
    calcule o score$_{heuristica}$(p,c)
  end for
  selecione p como uma propriedade-nome se score$_{heuristica}$(p,c) atende a criterio$_{heuristica}$
end for

5.1.1 Heurística Ingênua

A Heurística Ingênua é a mais simples e é a que deu origem a todas as outras. Para cálculo do score utilizaremos apenas as duas primeiras características levantadas sobre os literais na base de dados. O score de uma propriedade p para uma determinada classe c é dado pelo número de ocorrências de nomes próprios como valores de p em recursos da classe c. O valor do score é calculado segundo a Fórmula (5.1) onde e_c é uma instância da classe c.

$$score_{ing}(p, c) = \sum_{\forall e_c} \begin{cases} 1, & \text{se } p_{valor} = \text{nome próprio} \\ 0, & \text{caso contrário} \end{cases} \tag{5.1}$$

Esta heurística visa selecionar a propriedade com maior score em cada classe, assumindo apenas propriedades com valor de score maior que zero. O maior score por classe é então o critério de seleção utilizado em conjunto com esta heurística.

Uma das características desta forma de seleção é o fato de que uma única propriedade

é identificada como propriedade-nome para cada classe presente na base de dados. Se todas as propriedades relacionadas a uma dada classe apresentam score igual a 0 então esta classe não possui propriedade-nome relacionada e, consequentemente, não será tida como uma classe que identifica Entidades Nomeadas.

5.1.2 Heurística Ingênua Parametrizada

A Heurística Ingênua Parametrizada visa melhorar o resultado da Heurística Ingênua através da adição de algumas restrições.

Como a Heurística Ingênua visa apenas a identificação de uma única propriedade-nome por classe, é importante que a propriedade represente realmente um nome e não um texto ou uma sigla (que também pode ser um nome mas, em geral, não é o nome principal). Deste modo utilizamos o número de caracteres dos literais presentes no valor de cada propriedade como forma de eliminar valores indesejados.

Textos muito longos tendem a se referir a descrições, portanto utilizamos um parâmetro que delimita o máximo (max) de caracteres permitido para que uma cadeia de caracteres seja reconhecida como um nome. Da mesma forma, textos muito pequenos têm a tendência a representar siglas, o que nos levou a criar um parâmetro de mínimo (min) de caracteres.

O cálculo do score da Heurística Ingênua Parametrizada é uma modificação do score da Heurística Ingênua, sendo calculado segundo a Fórmula (5.2).

$$score_{param}(p, c) = \sum_{\forall e_c} \begin{cases} 1, & \text{se } p_{valor} = \text{nome próprio, tamanho}(p_{valor}) \in [min, max] \\ 0, & \text{caso contrário} \end{cases} \tag{5.2}$$

O critério de seleção de propriedades-nome é o mesmo que o utilizado para a Heurística Ingênua, ou seja, a propriedade com o maior valor de score para cada classe. Mesmo que os parâmetros *max* e *min* excluam da contagem do score alguns nomes válidos, ainda assim a propriedade escolhida é aquela que tem o maior número de nomes válidos, podendo ser reconhecida como uma propriedade-nome.

5.1.3 Heurística Multivalorada

A Heurística Multivalorada tem por objetivo dar um passo na melhoria da Heurística anterior. O objetivo agora é tentar identificar mais de uma propriedade como propriedade-nome por classe.

O cálculo desse novo score exclui o uso do parâmetro *min* pois a partir de agora podemos ter várias propriedades-nome e, portanto, assumir que siglas também são possíveis nomes para Entidades Nomeadas. O cálculo do score foi modificado para o apresentado pela Fórmula (5.3).

$$score_{multi}(p, c) = \sum_{\forall e_c} \begin{cases} 1, & \text{se } p_{valor} = \text{nome próprio, tamanho}(p_{valor}) \leq max \\ 0, & \text{caso contrário} \end{cases} \tag{5.3}$$

A maior variação com relação à Heurística anterior é no critério de seleção onde agora todas as propriedades com valor de score maior que 0 são identificadas como propriedades-nome. Mais uma vez, classes sem uma propriedade-nome associada não identificam Entidades Nomeadas.

5.1.4 Heurística Multivalorada com Threshold

Apesar de retornar mais propriedades-nome, a Heurística Multivalorada é um mero relaxamento das restrições impostas pela Heurística Ingênua Parametrizada, sendo capaz de retornar um grande número de propriedades erradas se considerarmos que o valor do parâmetro *max* não foi muito bem escolhido.

A Heurística Multivalorada com Threshold foi então criada a fim de minimizar os possíveis erros da Heurística Multivalorada. Seu score é calculado não mais com base no número de ocorrências mas sim na porcentagem de ocorrências de nomes em instâncias de uma determinada classe. Ou seja, se uma classe identifica Entidades Nomeadas e uma determinada propriedade identifica o nome das instâncias desta classe, então esta propriedade deve aparecer na descrição de grande parte, se não em todas, as instâncias da classe. Este novo score é dado de acordo com a Fórmula (5.4) onde $\mid e_c \mid$ é o número total de instâncias da classe c.

$$score_{threshold}(p, c) = score_{multi}(p, c) / \mid e_c \mid \tag{5.4}$$

Na construção deste método tomamos o cuidado de não dar um peso maior para uma propriedade ou outra. Em casos onde uma propriedade se refere a um nome alternativo, a tendência é que essa propriedade apareça mais de uma vez por instância caso a Entidade Nomeada representada tenha mais de um nome alternativo. Portanto, passaremos a contar apenas uma única ocorrência de cada propriedade por instância em cada classe.

Como critério de seleção utilizamos um valor de Threshold que será o score mínimo que uma propriedade deve ter para ser considerada uma propriedade-nome. Caso seja de nosso desejo selecionar apenas as propriedades-nome principais então este

valor deve ser próximo dos 100%. Caso desejemos identificar propriedades que representem siglas ou nomes alternativos para uma mesma Entidade Nomeada então o valor de threshold deve ser menor.

5.2 Avaliação

Tendo descrito o método e as heurísticas, passamos então à avaliação de sua aplicação utilizando bases de dados em Linked Data. Nosso objetivo aqui é verificar a real aplicabilidade do método além de identificar a melhor situação para uso das diferentes heurísticas.

Utilizamos as bases de dados Linked Movie Database e Jamendo, selecionadas no Capítulo 4, para nosso processo de avaliação. O primeiro passo foi analisar ambas as bases a fim de criar um Gold Standard, ou seja, o conjunto de respostas dadas como corretas para o componente de Reconhecimento de Propriedades-Nome.

Nas próximas subseções detalharemos o processo de criação do Gold Standard e comentaremos a performance de cada heurística aplicada a cada uma das bases de dados. Ao final faremos uma breve comparação de todas as heurísticas desenvolvidas.

5.2.1 Gold Standard

O Gold Standard é o conjunto formado pelas respostas esperadas para o componente de Reconhecimento de Propriedades-nome, dada uma determinada base em Linked Data como entrada.

O processo de criação do Gold Standard seguiu os seguintes passos:

Passo 1. Identificação de todas as classes que possuem instâncias na base de dados em Linked Data

Passo 2. Para cada classe, identificamos todas as propriedades que têm um literal como valor.

Passo 3. Análise do significado de cada classe e propriedade.

Passo 4. Seleção das classes que identificam Entidades Nomeadas.

Passo 5. Seleção das propriedades-nome para cada classe que identifica Entidades Nomeadas.

Para identificar todas as propriedades-nome precisamos fazer o levantamento de todas as classes realmente existentes na base de dados (Passo 1). Depois disto, fizemos um levantamento de todas as propriedades utilizadas para descrever cada uma das classes, levando em conta apenas as propriedades que apresentam literais como seus valores (Passo 2).

De posse de todas as classes e propriedades relevantes, passamos à identificação do significado de cada um destes dados (Passo 3). A semântica dos dados em Linked Data é dada pelos vocabulários utilizados em sua descrição. Sendo assim, procuramos levantar todos os vocabulários utilizados e buscamos os significados relativos a cada item levantado. Caso a descrição do vocabulário não estivesse disponível, como aconteceu com a Linked Movie Database, fizemos a avaliação do significado utilizando um conjunto de instâncias de cada classe e os nomes de cada propriedade e classe.

Quadro 5.1: Propriedades-nome da base de dados Jamendo

Classe	Categoria	Propriedade-nome
mo:Record	preferencial	dc:title
mo:MusicArtist	preferencial	foaf:name
mo:Track	preferencial	dc:title

Como resultado do Passo 3 pudemos selecionar todas as classes que identificam Entidades Nomeadas (Passo 4) e todas as propriedades-nome (Passo 5). É importante ressaltar que os dois últimos passos acontecem quase que simultaneamente pois uma classe que identifica Entidade Nomeada é exatamente aquela que possui pelo menos uma propriedade-nome.

Durante o processo de seleção das propriedades-nome separamo-las em três categorias: preferencial, alternativa e sigla.

As propriedades-nome da categoria ***preferencial*** são aquelas que indicam os nomes preferenciais das Entidades Nomeadas. Sua característica é estar presente em todas as instâncias de uma classe, uma única vez por instância.

Dentre as propriedades-nome da classe ***alternativa*** estão aquelas que se referem a possíveis nomes pelos quais as Entidades Nomeadas são conhecidas. Elas podem aparecer uma ou mais vezes em uma mesma instância mas não aparecem necessariamente em todas as instâncias de uma dada classe.

As propriedades-nome na categoria de ***siglas*** são aquelas cujos valores são textos pequenos com maior ocorrência de letras maiúsculas. Estas propriedades seguem as mesmas características das propriedades-nome alternativas.

O resultado do nosso Gold Standard com todas as classes que identificam Entidades Nomeadas e suas respectivas propriedades-nome podem ser visto nos Quadros 5.1

e 5.2 para a Jamendo e a Linked Movie Database respectivamente. Note que não encontramos nenhuma propriedade-nome alternativa nas bases de dados avaliadas, embora este tipo de propriedade-nome seja comum em outras bases.

Quadro 5.2: Propriedades-nome da base de dados Linked Movie Database

Classe	Categoria	Propriedade-nome
movie:Actor	preferencial	movie:actor_name
movie:Cinematographer	preferencial	movie:cinematographer_name
movie:Country	preferencial	movie:country_name
	sigla	movie:country_fips_code
	sigla	movie:country_iso_alpha2
	sigla	movie:country_iso_alpha3
movie:Director	preferencial	movie:director_name
movie:Editor	preferencial	movie:editor_name
movie:Film	preferencial	dc:title
	preferencial	rdfs:label
movie:FilmArtDirector	preferencial	movie:film_art_director_name
movie:FilmAwardsCeremony	preferencial	movie:film_awards_ceremony_name
movie:FilmCastingDirector	preferencial	movie:film_casting_director_name
movie:FilmCharacter	preferencial	movie:film_character_name
movie:FilmCollection	preferencial	movie:film_collection_name
movie:FilmCompany	preferencial	movie:film_company_name
movie:FilmCostumeDesigner	preferencial	movie:film_costume_designer_name
movie:FilmCrewGig	preferencial	movie:film_crew_role
movie:FilmCrewMember	preferencial	movie:film_crewmember_name
movie:FilmCritic	preferencial	movie:film_critic_name
movie:FilmDistributor	preferencial	movie:film_distributor_name
movie:FilmFeaturedSong	preferencial	movie:film_featured_song_name
movie:FilmFestival	preferencial	movie:film_festival
movie:FilmFestivalEvent	preferencial	movie:film_festival_event_name
movie:FilmFormat	preferencial	movie:film_format_name
movie:FilmGenre	preferencial	movie:film_genre_name
movie:FilmJob	preferencial	movie:film_job_name
movie:FilmLocation	preferencial	movie:film_location_name
movie:FilmProductionDesigner	preferencial	movie:film_production_designer_name
movie:FilmSeries	preferencial	movie:film_series_name
movie:FilmSetDesigner	preferencial	movie:film_set_designer_name

movie:FilmStoryContributor	preferencial	movie:film_story_contributor_name
movie:FilmTheorist	preferencial	movie:film_theorist_name
movie:MusicContributor	preferencial	movie:music_constributor_name
movie:Producer	preferencial	movie:producer_name
movie:ProductionCompany	preferencial	movie:production_company
movie:Writer	preferencial	movie:writer_name
foaf:Person	preferencial	movie:actor_name
	preferencial	movie:editor_name
	preferencial	movie:director_name
	preferencial	movie:cinematographer_name
	preferencial	movie:film_crewmember_name

5.2.2 Jamendo

A base de dados Jamendo é uma base relativamente pequena contando com apenas 11 classes. Destas, somente 3 classes identificam Entidades Nomeadas com uma única propriedade-nome associada a cada uma. A seguir, discutiremos o resultado da aplicação das 4 heurísticas desenvolvidas a fim de verificar o comportamento de cada uma.

A Heurística Ingênua foi capaz de reconhecer todas as três propriedades-nome previstas. Por outro lado, a propriedade *mo:text* da classe *mo:Lyrics* foi retornada como resposta erroneamente (Quadro 5.3). Esta propriedade tem como seu valor a letra completa de uma determinada música, onde grande parte das palavras são iniciadas por letra maiúscula.

Para análise da aplicação da Heurística Ingênua Parametrizada utilizamos os parâmetros $min = 4$ e $max = 100$. Mesmo com a adição destas restrições, a propriedade

Quadro 5.3: Resultado das Heurísticas Ingênua e Ingênua Parametrizada para a base Jamendo

Classe	Propriedades Retornadas
mo:Lyrics	mo:text
mo:MusicArtist	foaf:name
mo:Record	dc:title
mo:Track	dc:title

mo:text continuou sendo retornada como propriedade-nome (Quadro 5.3). Apesar do score ter diminuído significativamente (de cerca de 6800 para cerca de 900) ainda assim o score possui valor maior que 0, permitindo então que a propriedade fôsse identificada como propriedade-nome e, por consequência, a classe *mo:Lyrics* fôsse retornada como identificando Entidades Nomeadas. Percebemos ainda que a adição dos parâmetros não causou modificação nas propriedades corretamente identificadas.

Dado o erro apresentado com o parâmetro de $max = 100$, tentamos diminuir este valor gradativamente a fim de definir um valor que pudesse eliminar a propriedade *mo:text* dos resultados. Diminuindo de 10 em 10 descobrimos que, mesmo chegando ao valor de $max = 20$, a propriedade ainda é retornada com um score de cerca de 900. Visto que é bastante incomum uma letra de música possuir menos de 20 caracteres resolvemos analisar os valores da propriedade a fim de identificar o que poderia estar causando este resultado. Em uma análise superficial conseguimos encontrar 117 recursos cujo valor da propriedade *mo:text* é o texto "instrumental", além de casos onde o valor é o nome do artista responsável pela música. No caso do texto "instrumental" percebemos que este valor é inserido sempre que não há letra de música, ou seja, em vez de optar por não inserir um recurso da classe *mo:Lyrics*, o autor resolveu colocar o texto "instrumental" como valor da propriedade *mo:text* para indicar que a música não possui letra.

O tipo de dado apresentado pela base de dados Jamendo para a propriedade *mo:text*

pode ser visto como um dado de baixa qualidade. Este tipo de dado afetou negativamente o nosso método e pode ser considerado como um problema com relação à qualidade da base de dados e não com relação ao método ou à heurística utilizada.

A seguir, aplicamos o método com a Heurística Multivalorada mantendo o valor $max = 100$. Além dos resultados obtidos pela Heurística Ingênua Parametrizada, o método retornou duas outras propriedades errôneas: *mo:biography* e *dc:description*. A primeira propriedade tem como seu valor o texto da biografia de um determinado artista, enquanto a segunda representa uma descrição dada a uma determinada gravação de um disco. A fim de verificar se o problema residia no método ou na base, mais uma vez restringimos o valor de *max* a 20 sendo que ambos os textos das propriedades deveriam, idealmente, conter mais de 20 caracteres. Ainda assim as propriedades foram coletadas, embora tenham apresentado um score muito baixo (8 e 18).

Quadro 5.4: Resultado da Heurística Multivalorada para a base Jamendo

Classe	Propriedades Retornadas
mo:Lyrics	mo:text
mo:MusicArtist	foaf:name
	mo:biography
mo:Record	dc:title
	dc:description
mo:Track	dc:title

Para análise da Heurística Multivalorada com Threshold utilizamos 4 valores diferentes de Threshold para ver como o método iria se comportar. Os valores de Threshold utilizados foram: 0,4, 0,6, 0,8 e 0,95. Para os valores 0,4 e 0,6 o método obteve 100% de acerto encontrando todas as propriedades-nome. Quando aumentamos para 0,8, a propriedade *dc:title* deixa de ser retornada como propriedade-nome para a classe *mo:Track*. Isto acontece porque o número de ocorrências onde o valor da propriedade é um nome próprio é inferior a 80% do número de instâncias presentes na classe. Ou seja, apesar desta propriedade estar presente em todas as

instâncias, nem sempre seu valor é o esperado (um nome próprio). Ao aumentarmos o valor do Threshold pra 0,95 as outras propriedades caem no mesmo caso e o último método acaba por não retornar nenhuma propriedade-nome.

Mais uma vez identificamos a má qualidade dos dados como um dos problemas para o uso da Heurística Multivalorada com Threshold. Algumas vezes o valor das propriedades não é escrito com letra maiúscula como é característico de um nome próprio, enquanto outras vezes o nome da música, por exemplo, consta como "2" ou "3" e não como o nome real da música. Estes valores acabam por diminuir o score de todas as heurísticas e, no caso da Multivalorada com Threshold, acabam por inviabilizar a identificação de algumas propriedades-nome.

Outra característica interessante da base de dados, mas que não interfere drásticamente nos nossos resultados, é o fato de que a base possui músicas com nomes realmente diferentes como "!" ou ",".

Quadro 5.5: Resultado da Heurística Multivalorada com Threshold para a base Jamendo

Classe	Propriedades Retornadas	Valor de Threshold
mo:MusicArtist	foaf:name	0.4/0.6/0.8
mo:Record	dc:title	0.4/0.6/0.8
mo:Track	dc:title	0.4/0.6

5.2.3 Linked Movie Database

A Linked Movie Database é uma base de dados com 53 classes onde 34 identificam Entidades Nomeadas. Dentre as classes que identificam Entidades Nomeadas (Quadro 5.2) notamos que a classe *Film Collection* possui uma única instância, mesmo assim resolvemos mantê-la dentre as classes sob análise a fim de verificar como cada heurística irá lidar com ela.

A aplicação de nosso método em conjunto com a Heurística Ingênua retornou 32 propriedades-nome das 42 existentes e mais 19 outras propriedades-nome erroneamente. Uma das respostas erradas foi a identificação da *rdfs:label* para a classe *Person* devido ao fato de que ela aparece com frequência maior que qualquer uma das propriedades-nome para esta classe.

Uma característica da base de dados que também resultou em erro foi a existência de propriedades cujo valor é realmente o nome de uma Entidade Nomeadas mas não o nome da entidade descrita pelo recurso. Uma propriedade deste tipo é a *movie:film_regional_release_date_film_release_region* em recursos da classe *Film Regional Release Date*. Esta classe representa a data do lançamento de um filme em um determinado local e a propriedade tem como valor o nome deste local. Ou seja, a propriedade realmente tem como valor o nome de uma Entidade Nomeada mas não a propriedade nomeada a qual descreve. Uma possível solução para este problema seria a troca desta propriedade por uma que represente o relacionamento entre o recurso descrito e um recurso da classe *Location*, por exemplo.

Uma característica no uso da Heurística Ingênua é o fato de que o resultado pode variar quando um conjunto de propriedades apresenta o mesmo score. Em uma execução, o método reconheceu a propriedade *movie:country_iso_alpha3* como propriedade-nome para a classe *Country*, em outra ele reconhece a *movie:country_name*. Isto ocorre pela falta de restrições do método, permitindo que uma propriedade representando uma sigla possa ser escolhida em vez da propriedade-nome preferencial. Como a escolha da propriedade que será avaliada primeiro pode variar de acordo com o SPARQL Endpoint, a propriedade escolhida, neste caso, pode variar. Outro ponto interessante é que o score da propriedade *rdfs:label* empatou com as propriedades-nome corretas em muitas classes, mas devido ao nosso critério de desempate a eficiência do algoritmo não foi afetada.

Em seguida aplicamos o método com a Heurística Ingênua Parametrizada com $min = 4$ e $max = 100$. Desta vez o número de propriedades-nome corretamente identificadas caiu para 19 enquanto o número de falsos positivos subiu para 32. Enquanto a propriedade-nome para a classe *Country*, que possui muitas siglas, foi corretamente identificada devido aos parâmetros, outras classes foram prejudicadas. Alguns nomes possuem 4 caracteres enquanto o respectivo valor para a propriedade *rdfs:label* no mesmo recurso é constituído pelo nome de 4 caracteres mais o nome da classe, o que fez com que a propriedade correta obtivesse um score menor que a *rdfs:label* aumentando em muito o número de falsos positivos.

O uso da Heurística Multivalorada permitiu o reconhecimento de todas as propriedades-nome mas também retornou um número muito maior de falsos positivos. Os 83 falsos positivos foram encontrados devido aos fatores já identificados nas heurísticas anteriores.

Por último, avaliamos o uso da Heurística Multivalorada com Threshold com os mesmos valores utilizados para avaliação da base de dados anterior: 0,4, 0,6, 0,8 e 0,95. Com o valor de 0,4, o método reconheceu 36 propriedades-nome retornando 66 falso positivos. Mais uma vez houve grande influência da *rdfs:label* e das propriedades que deveriam ser relacionamentos. Com 0,6 o método reconheceu o mesmo número de propriedades-nome com apenas 63 falsos positivos. Aumentando o valor do Threshold para 0,8 foram reconhecidas 35 propriedades-nome e 59 falsos positivos, e com o valor em 0,95 foram identificadas 33 propriedades-nome e o número de falsos positivos caiu para 52.

A única classe que não poderia ter suas propriedades-nome corretamente identificadas para valores de Threshold muito altos é a classe *Person*. Esta classe tem uma característica muito particular onde todas as suas instâncias são também instâncias de uma das seguintes classes: *Actor*, *Cinematographer*, *Director*, *Editor* e *Film Crew*

Member. A classe *Person* herda as propriedades-nome das classes que compartilha recursos com ela, portanto, ela possui cinco propriedades-nome que não aparecem concomitantemente em um único recurso. Sendo assim, por participar de uma fração pequena dos recursos da classe, cada propriedade-nome não será retornada por esta última heurística se o valor de threshold for maior que a porcentagem de recursos em que a propriedade aparece.

Quadro 5.6: Propriedades-nome corretamente identificadas por cada método para a Linked Movie Database. I = Heurística Ingênua, IP = Heurística Ingênua Parametrizada, M = Heurística Multivalorada e MT = Heurística Multivalorada com Threshold

Classe	Propriedade	I	IP	M	MT			
					0.4	0.6	0.8	0.95
movie:Actor	movie:actor_ name	X		X	X	X	X	X
movie:Cinematographer	movie:cinematographer_name	X	X	X	X	X	X	X
movie:Country	movie:country_name	X	X	X	X	X	X	X
	movie:country_fips_code			X	X	X	X	X
	movie:country_iso_alpha2			X	X	X	X	X
	movie:country_iso_alpha3			X	X	X	X	X
movie:Director	movie:director_name	X		X	X	X	X	X
movie:Editor	movie:editor_name	X	X	X	X	X	X	X
movie:Film	dc:title	X	X	X	X	X	X	X
	rdfs:label			X	X	X	X	X
movie:FilmArtDirector	movie:film_art_director_name	X	X	X	X	X	X	X
movie:FilmAwards Ceremony	movie:film_awards_ceremony_name	X	X	X				
movie:FilmCasting Director	movie:film_casting_director_name	X		X	X	X	X	X
movie:FilmCharacter	movie:film_character_name	X		X	X	X	X	X
movie:FilmCollection	movie:film_collection_name	X	X	X	X	X	X	X

movie:FilmCompany	movie:film_company_name	X	X	X	X	X	X	X
movie:FilmCostume Designer	movie:film_costume_designer_name	X	X	X	X	X	X	X
movie:FilmCrewGig	movie:film_crew_role			X	X	X	X	X
movie:FilmCrew Member	movie:film_crewmember_name	X	X	X	X	X	X	X
movie:FilmCritic	movie:film_critic_name	X	X	X	X	X	X	X
movie:FilmDistributor	movie:film_distributor_name	X		X	X	X	X	X
movie:FilmFeatured Song	movie:film_featured_song_name	X	X	X	X	X	X	X
movie:FilmFestival	movie:film_festival_name	X	X	X	X	X	X	X
movie:FilmFestival Event	movie:film_festival_event_name	X	X	X				
movie:FilmFormat	movie:film_format_name	X	X	X	X	X	X	
movie:FilmGenre	movie:film_genre_name	X		X	X	X	X	X
movie:FilmJob	movie:film_job_name	X		X	X	X	X	X
movie:FilmLocation	movie:film_location_name	X		X	X	X	X	X
movie:FilmProduction Designer	movie:film_production_designer_name	X	X	X	X	X	X	
movie:FilmSeries	movie:film_series_name	X		X	X	X	X	X
movie:FilmSetDesigner	movie:film_set_designer_name	X	X	X	X	X	X	X
movie:FilmStory Contributor	movie:film_story_contributor_name	X	X	X	X	X	X	X
movie:FilmTheorist	movie:film_theorist_name	X	X	X	X	X	X	X
movie:Music Contributor	movie:music_contributor_name	X		X	X	X	X	X
movie:Producer	movie:producer_name	X		X	X	X	X	X

movie:Production Company	movie:production_company_name	X		X	X	X	X	X
movie:Writer	movie:writer_name	X		X	X	X	X	X
foaf:Person	movie:actor_name			X	X	X		
	movie:editor_name			X				
	movie:director_name			X				
	movie:cinematographer_name			X				
	movie:film_crewmember_name			X				

5.2.4 Análise Comparativa

Cada heurística apresenta características diversas e a escolha do melhor método depende das características de cada base de dados.

Ao retornar uma única propriedade-nome por classe que identifica Entidades Nomeadas, as heurísticas Ingênuas nem sempre identificam todas as propriedades-nome existentes na base. Estas heurísticas priorizam a precisão em vez do recall.

A heurística Multivalorada conseguiu reconhecer todas as propriedades-nome em ambas as bases de dados avaliadas porém foi também a que retornou o maior número de falsos positivos. A heurística Multivalorada com Threshold manteve um bom recall mas com uma precisão melhor que a Multivalorada. Estas heurísticas são indicadas para aplicações onde o fator mais importante é o recall e não a precisão.

No caso de nossa aplicação todas as heurísticas podem ser utilizadas. Caso o objetivo seja o reconhecimento de nomes o mais correto possível, as heurísticas Ingênuas ou

a Multivalorada com alto valor de Threshold são as mais indicadas. Caso o objetivo seja apenas diminuir o espaço de busca por nomes em uma base muito grande, então as heurísticas Multivalorada e Multivalorada com um baixo valor de Threshold são mais interessantes.

Além das propriedades-nome, as heurísticas também são capazes de identificar as classes que identificam Entidades Nomeadas. Esta característica permite que as propriedades-nome possam ser selecionadas manualmente com maior facilidade pois diminuem a quantidade de dados a ser analisada.

Quadro 5.7: Resultado da aplicação do método de reconhecimento de propriedades-nome

	Jamendo			Linked Movie Database		
Heurísticas	Precisão	Recall	F1	Precisão	Recall	F1
Ingênua	0,75	1	0,8571	0,6274	0,7619	0,6882
Ingênua Parametrizada	0,75	1	0,8571	0,3725	0,4524	0,4086
Multivalorada	0,5	1	0,6667	0,3360	1	0,5030
Multivalorada com Threshold (0,4)	1	1	1	0,3529	0,9000	0,5070
Multivalorada com Threshold (0,6)	1	1	1	0,3636	0,9000	0,5180
Multivalorada com Threshold (0,8)	1	0,6667	0,8	0,3723	0,8750	0,5224
Multivalorada com Threshold (0,95)	0	0	0	0,3882	0,8684	0,5366

5.3 Trabalhos Relacionados

Todos os trabalhos em Entity Linking utilizando bases em Linked Data costumam, em algum momento, identificar os nomes das entidades que a base descreve. Alguns utilizam os nomes para a geração de Gazetteers enquanto outros os utilizam apenas no momento de efetuar o link do recurso da base com a menção no texto. Como a

Quadro 5.8: Resultado do reconhecimento de classes que identificam Entidades Nomeadas

	Jamendo			Linked Movie Database		
Heurísticas	Positivo	Falso Positivo	Falso Negativo	Positivo	Falso Positivo	Falso Negativo
Ingênua	3	1	0	34	17	0
Ingênua Parametrizada	3	1	0	34	17	0
Multivalorada	3	1	0	34	17	0
Multivalorada com Threshold (0,4)	3	0	0	32	16	2
Multivalorada com Threshold (0,6)	3	0	0	32	15	2
Multivalorada com Threshold (0,8)	2	0	1	32	15	2
Multivalorada com Threshold (0,95)	0	0	3	30	12	4

maioria das aplicações no ramo é de caráter comercial não temos informações sobre seu processo de busca por nomes, sendo assim pudemos avaliar apenas a DBPedia Spotlight (MENDES et al., 2011) e a AIDA (HOFFART et al., 2011).

A DBPedia Spotlight identifica o nome dos recursos de duas formas. A primeira inclui utilizar apenas o *rdfs:label* para extração do nome de todos os recursos, sem identificar se eles realmente são Entidades Nomeadas ou não. Esta característica facilita a identificação de conceitos no texto mas também permite que pequenas descrições possam ser retornadas como nomes. A segunda forma de identificação de nomes ocorre através de textos extraídos de links da Wikipedia, fonte para os dados da DBPedia. Ou seja, um determinado recurso da DBPedia é representado por uma página da Wikipedia de onde todos os textos em links que apontam para

essa página são extraídos como possíveis nomes para o recurso.

Diferente da nossa abordagem, a DBPedia Spotlight assume a *rdfs:label* como propriedade-nome para todas as classes. Esta abordagem é suficiente para a aplicação, mas ela poderia ser expandida pelo uso de outras propriedades-nome presentes na base.

A AIDA utiliza a base de dados YAGO (SUCHANEK; KASNECI; WEIKUM, 2007) e a propriedade *yago:means* como fonte para nomes das entidades. Os trabalhos publicados não informam se outras propriedades são também utilizadas, visto que a YAGO apresenta propriedades como a *yago:hasPreferredName* que também indica nomes.

6 GERAÇÃO DE DICIONÁRIO DE NOMES

Para utilizar uma base de dados em Linked Data como fonte de Entidades Nomeadas, o próximo passo após a identificação das Entidades Nomeadas e seus nomes é a transformação da base de dados em um Gazetteer.

Gazetteer é o nome dado para o dicionários de nomes. Sua construção é feita utilizando os nomes como índices e os recursos que descrevem Entidades Nomeadas como os valores relacionados a cada índice. Cada Entidade Nomeada é indexada pelo nome que a identifica. Se uma entidade é conhecida por mais de um nome então ela aparecerá relacionada a um conjunto de nomes utilizados como índice.

Um Gazetteer é uma estrutura de dados que necessita de um serviço relacionado para garantir a consulta por nomes. A este serviço chamamos Serviço de Lookup. A cada nome dado como entrada, o serviço retornará a lista de entidades nomeadas identificadas por aquele nome no Gazetteer.

Em nossa arquitetura o Gazetteer e seu serviço de Lookup são gerados no componente de Geração de Dicionário de Nomes (Figura 6.1). O componente recebe uma base de dados em Linked Data como entrada e opcionalmente recebe também o

resultado do componente de Reconhecimento de Propriedades-Nome para restringir o espaço de busca por nomes. A saída esperada para este componente é um mapa formado por nomes como índices e URIs para recursos da base em Linked Data como valores, ou seja, o Gazetter propriamente dito. Além disto, o componente é responsável por gerar o serviço de lookup que também será retornado como resultado.

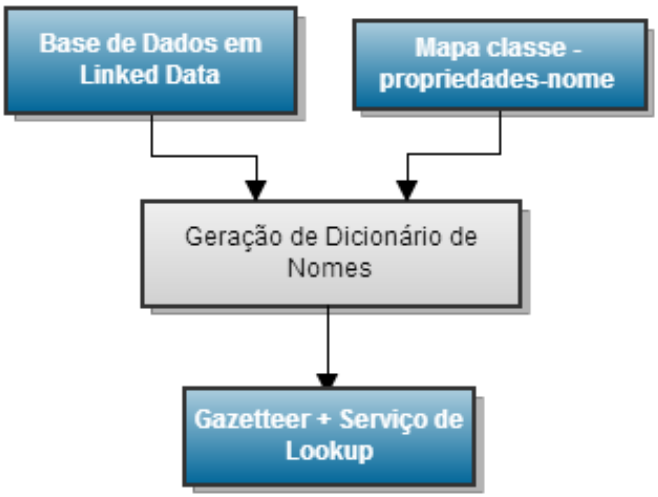


Figura 6.1: Entradas e Saídas esperadas para o componente de Geração de Dicionário de Nomes

Para construção deste componente avaliamos a utilização de duas tecnologias diferentes e de dois métodos para aquisição de nomes. Nas próximas seções abordaremos em detalhes os métodos para aquisição de nomes (Método Geral e Método Limitado a Propriedades-nome), ambas as tecnologias para geração do Gazetteer e do serviço de lookup (utilizando SPARQL e Lucene) e ao final faremos a avaliação dos Gazetteers gerados a partir dos métodos e tecnologias apresentados.

6.1 Métodos para Busca de Nomes

O primeiro passo na transformação de uma base de dados qualquer em um Gazetteer é a identificação de quais estruturas representam as Entidades Nomeadas e seus nomes.

Como nosso objetivo é lidar com uma base em Linked Data sem uma estrutura de descrição definida, não partimos de um conjunto pré-definido de classes e propriedades relacionadas a Entidades Nomeadas e a seus nomes. Sendo assim, o primeiro método para construção do Gazetteer visa a utilização de toda a base como fonte para nomes, e a esse método chamamos método Geral.

Apesar do método Geral permitir o trabalho com bases genéricas de forma mais direta, utilizar toda a base como fonte para nomes pode resultar em processamento desnecessário além de permitir uma grande quantidade de erros. A partir desta ideia, desenvolvemos o método Limitado a propriedades-nome que utiliza apenas propriedades-nome como fonte para nomes.

6.1.1 Geral

O princípio do qual partimos ao desenvolver este método é o de que a entrada para um serviço de lookup será sempre um nome a ser procurado. Estando este nome na base de dados, conseguimos descobrir o conjunto de Entidades Nomeadas a que ele se refere procurando por todos os recursos que possuem este nome como valor de uma de suas propriedades.

Segundo o algoritmo proposto para implementação deste método (Algoritmo 2), percorremos todos os recursos que possuem alguma propriedade cujo valor é um

literal. Caso o valor da propriedade seja o nome que está sendo buscado então o recurso anotado por esta propriedade provavelmente representa a Entidade Nomeada que buscamos. Sendo assim, a URI do recurso é dada como uma das respostas à busca.

Algoritmo 2 Método Geral de Geração de Gazetteers

```
LD ← base de dados em Linked Data
P ← conjunto de todas as propriedades cujo valor é um literal
nome ← nome que se quer procurar
for cada recurso r ∈ LD que é descrito por p ∈ P do
    if valor de p = nome then
        retorne como resposta a URI que identifica r
    end if
end for
```

Como exemplo podemos dizer que estamos procurando pelo nome "Margin of Safety" na base de dados Jamendo. Se um dos recursos na base possui este nome como valor de uma de suas propriedades, então há uma grande chance de que este recurso represente a Entidade Nomeada que procuramos. Sendo assim, a URI do recurso será retornada como uma das respostas na execução do algoritmo.

Este algoritmo tem alguns problemas inerentes à sua criação. O primeiro deles é o tempo de processamento que aumenta conforme o crescimento da base de dados, independente se os novos dados referem-se ou não a Entidades Nomeadas. Além disso, um texto de descrição, por exemplo, nunca será o nome de uma entidade. Mesmo assim o método irá comparar este texto com o nome que está sendo buscado em todas as execuções do algoritmo. Esta característica faz com que o algoritmo tome um tempo desnecessário de processamento.

Outro problema identificado é que nem sempre o texto que aparece como valor de uma propriedade é realmente o nome da Entidade Nomeada representada pelo recurso onde aparece. Um exemplo disto foi visto na avaliação das bases de dados

em Linked Data (Capítulo 4) onde recursos da classe Performance (na Linked Movie Database) possuem nomes de atores como o valor de uma de suas propriedades. Se o nome de um desses atores é dado como entrada então o respectivo recurso, instância de Performance, será retornado como resposta erroneamente.

6.1.2 Limitado a propriedades-nome

A fim de reduzir os problemas causados pelo método Geral, desenvolvemos o método Limitado a propriedades-nome. Este novo método tem por finalidade limitar o espaço de busca a um conjunto de propriedades que realmente identificam os nomes de recursos que representam Entidades Nomeadas.

O método Limitado a propriedades-nome utiliza um algoritmo similar ao método anterior, com a diferença de que P passa a ser agora o conjunto de propriedades-nome relacionadas a cada classe.

Segundo o algoritmo desenvolvido (Algoritmo 3), o valor de uma propriedade só é verificado se ela pertence à lista de propriedades-nome da classe da qual o recurso é uma instância. Com isso, o espaço de busca fica restrito ao real espaço de nomes para Entidades Nomeadas existente na base de dados.

Algoritmo 3 Método de Geração de Gazetteers Limitado a propriedades-nome

LD ← base de dados em Linked Data
P_{nome} ← conjunto de propriedades-nome
nome ← nome que se quer procurar
for cada recurso $r \in$ LD que é descrito por $p \in P_{nome}$, quando r e p estão relacionados à mesma classe **do**
    if valor de p = nome **then**
        retorne como resposta a URI que identifica r
    end if
end for

Neste método podemos permitir que o usuário informe o conjunto de propriedades-nome que deseja considerar ou podemos utilizar o resultado do componente de Reconhecimento de Propriedades-Nome como fonte para este dado.

6.2 Tecnologias para Criação do Serviço de Lookup

Nesta seção apresentamos dois métodos utilizando tecnologias diferentes para a geração do Gazetteer propriamente dito e dois mecanismos necessários para implementação do serviço de lookup. A primeira tecnologia utilizada é a linguagem de consulta SPARQL. Ao Gazetteer gerado a partir dela damos o nome de SPARQL Gazetteer que realiza o mapeamento de consultas por nomes para consultas SPARQL. O segundo Gazetteer gerado é o Lucene Gazetteer que utiliza a biblioteca Lucene como tecnologia. Esta biblioteca é fortemente utilizada para indexação e busca em documentos textuais. Neste caso, criamos uma representação da base de dados no modelo de documentos utilizado pela Lucene e mapeamos a consulta por nomes para uma busca nestes documentos.

6.2.1 SPARQL Gazetteer

O SPARQL Gazetteer foi nossa primeira tentativa de transformar uma base em Linked Data em um Gazetteer. Nosso objetivo em sua construção foi utilizar somente tecnologias próprias a bases de dados em Linked Data. Segundo Berners-Lee (2006), toda base de dados que segue as premissas de Linked Data deve disponibilizar algum meio para realização de consultas utilizando a linguagem SPARQL, que é a linguagem padrão de consulta. Por este motivo, o SPARQL Gazetteer foi construído assumindo a existência de um SPARQL Endpoint para realização das consultas SPARQL.

Mesmo conhecendo as propriedades que serão utilizadas na procura por nomes, devemos ainda identificar como essa busca será realizada. Para construção deste Gazetteer levantamos três diferentes tipos de busca que podem ser utilizadas:

- Busca simples por um literal como valor de uma propriedade;
- Busca utilizando expressões regulares;
- Busca por texto completo (full-text search).

Cada busca é constituída por uma ou mais consultas SPARQL seguidas por um pós-processamento que identifica se o resultado é válido.

A **Busca simples** prevê a verificação do valor de uma propriedade utilizando a cláusula FILTER. Esta é uma busca de fácil processamento e é geralmente suportada por qualquer tipo de SPARQL Endpoint. Um exemplo de consulta SPARQL que realiza este tipo de busca pode ser visto no Quadro 6.1.

Quadro 6.1: Consulta SPARQL utilizada na Busca Simples

```
SELECT ?uri
WHERE {
    ?uri ?propriedade ?valor.
    FILTER(isLiteral(?valor)).
    FILTER(?valor = "Michael Jackson").
}
```

O problema na consulta utilizada pela busca simples é que nem sempre a base de dados retorna uma resposta, mesmo que o nome esteja presente na base. O problema com esta busca foi detectado quando tentamos buscar o nome "Margin of

Safety" na base de dados Jamendo utilizando o gerenciador de triplas Openlink Virtuoso[1]. Este texto é o nome de um grupo musical realmente existente na base de dados mas notamos que a busca simples não retornava resultado. Isto ocorreu porque em algumas bases o literal é identificado como sendo do tipo string (cadeia de caracteres) e em outras não. A busca da forma como apresentamos retorna apenas valores literais que estejam identificados como sendo do tipo string (http://www.w3.org/2001/XMLSchema#string).

Uma forma de contornar o problema foi transformar o valor da propriedade e o nome que se quer procurar em string (Quadro 6.2) antes de compará-los.

Quadro 6.2: Consulta SPARQL aperfeiçoada utilizada na Busca Simples

```
SELECT ?uri
WHERE {
    ?uri ?propriedade ?valor.
    FILTER(isLiteral(?valor)).
    FILTER(str(?valor) = str("Margin of Safety")).
}
```

A segunda forma de busca é a **Busca por Expressão Regular**. Neste novo modelo utilizamos expressões regulares através do parâmetro regex da linguagem SPARQL para casar o valor da propriedade (Quadro 6.3) com o nome que estamos procurando. O problema com este tipo de busca é que nem todos os SPARQL Endpoints permitem o uso de regex, além do que, o uso deste parâmetro tende a deixar a busca mais complexa podendo afetar o desempenho em termos de tempo.

O último modelo de busca desenvolvido foi a **Busca por Texto Completo**. Algumas bases de dados permitem um método diferenciado de indexação para literais

[1] http://virtuoso.openlinksw.com/

Quadro 6.3: Consulta SPARQL utilizada na Busca por Expressão Regular

```
SELECT ?uri
WHERE {
    ?uri ?propriedade ?valor.
    FILTER(isLiteral(?valor)).
    FILTER(regex(?valor,"Michael Jackson")).
}
```

permitindo o que chamamos de full-text search, um método otimizado para buscas textuais. A consulta SPARQL neste modelo (Quadro 6.4) utiliza uma propriedade identificada como *bif:contains* que relaciona o campo e o valor que se quer verificar.

Quadro 6.4: Consulta SPARQL utilizada na Busca por Texto Completo

```
SELECT ?uri
WHERE {
    ?uri ?propriedade ?valor.
    ?valor bif:contains '"Michael Jackson"'.
}
```

O problema com a Busca por Texto Completo é que grande parte dos SPARQL Endpoints não disponibiliza o tipo de consulta SPARQL que necessitamos, tornando a sua utilização desta busca limitada a um pequeno conjunto de bases.

O SPARQL Gazetteer permite ainda a utilização de ambos os métodos levantados anteriormente, tanto o Geral quanto o Limitado a propriedades-nome. Neste caso, o SPARQL Gazetteer usa originalmente o método Geral visto que ele procura em toda a base de dados. O método Limitado a propriedades-nome pode ser utilizado para restringir o escopo da busca modificando a consulta SPARQL (Quadro 6.5), ou o resultado retornado pelo método, através de um pós-processamento dos resultados

retornados pela consulta.

Quadro 6.5: Consulta SPARQL usando método Limitado a propriedades-nome

```
SELECT ?uri
WHERE {
    ?uri ?propriedade ?valor.
    (...))
    OPTIONAL{
    FILTER(?propriedade = <http://xmlns.com/foaf/0.1/name>)}.
    OPTIONAL{
    FILTER(?propriedade = <http://purl.org/dc/elements/1.1/title>)}.
}
```

6.2.2 Lucene Gazetteer

O SPARQL Gazetteer é totalmente dependente da existência de um SPARQL Endpoint. A princípio isto não parece ser um problema, mas como nossa avaliação das bases de dados mostrou (Capítulo 4), os atuais Endpoints apresentam problemas de tempo de limite excedido para consultas muito simples. Como alternativa criamos o Lucene Gazetteer.

O Lucene Gazetteer foi construído utilizando a Lucene[2], uma biblioteca para busca em texto. A ideia deste Gazetteer é a mesma do SPARQL Gazetteer utilizando full-text search, com a diferença de que a Lucene é uma biblioteca especialmente desenvolvida para esta função.

O primeiro passo na construção do Lucene Gazetteer foi a remodelagem dos dados em Linked Data de forma que a Lucene pudésse processá-los, pois a biblioteca utiliza

[2]http://lucene.apache.org/

um formato diferente do RDF. A Lucene trabalha com o que ela chama internamente de documento. Cada documento conta com um ou mais campos onde cada campo é dividido em duas partes: nome e valor. Alguns campos de cada documento são selecionados para compôr os índices a serem utilizados no processo de busca.

A modelagem dos dados em Linked Data para a Lucene aconteceu da seguinte forma:

1. Criação de um documento para cada recurso na base em Linked Data
2. Para cada documento é adicionado um campo com o nome URI cujo valor será o identificador do recurso representado pelo documento.
3. Criação de um campo para cada propriedade que descreve o recurso. O nome do campo é o nome da propriedade e o valor será o valor da propriedade para o dado recurso.

No Quadro 6.6 apresentamos um exemplo de documento Lucene criado a partir de um recurso da base de dados Jamendo.

Quadro 6.6: Dados do documento Lucene referente ao recurso http://dbtune.org/jamendo/artist/1003 da Jamendo

Campo	Valor
URI	http://dbtune.org/jamendo/artist/1003
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://purl.org/ontology/mo/MusicArtist
http://xmlns.com/foaf/0.1/based_near	http://sws.geonames.org/3031359/
http://xmlns.com/foaf/0.1/name	“Cicada"
http://xmlns.com/foaf/0.1/homepage	http://www.cicada.fr.st
http://xmlns.com/foaf/0.1/img	http://img.jamendo.com/artists/h/hattrickman.jpg
http://xmlns.com/foaf/0.1/made	http://dbtune.org/jamendo/record/1086
http://xmlns.com/foaf/0.1/made	http://dbtune.org/jamendo/record/8309

Ao contrário do SPARQL Gazetteer que contava com servidores externos onde eram efetuadas as consultas SPARQL, o Lucene Gazetteer efetua a duplicação da base de dados em ambiente local. Esta nova abordagem tende a ser mais rápida devido às características da Lucene e não necessita de um SPARQL Endpoint externo toda vez que efetuamos a busca por um novo nome. Por outro lado, este Gazetteer exige um maior espaço de armazenamento e pode ficar desatualizado em comparação com o SPARQL Endpoint.

O Lucene Gazetteer exige ainda uma quantidade maior de tempo na primeira vez em que é executado pois é na primeira execução que será feita a indexação de todos os dados da base. Nas execuções seguintes fazemos apenas o procedimento de busca que ocorre em um tempo consideravelmente menor.

Com o objetivo de minimizar também o espaço de armazenamento utilizado, deveremos armazenar apenas os dados que serão utilizados pela busca. Na utilização do Gazetteer queremos identificar todas as URIs relacionadas a um nome dado como entrada. Sendo assim, devemos ter apenas as propriedades cujo valor é um literal (ou as propriedades-nome, dependendo do método) como índice para busca e as URIs como valores a serem retornados. Com essa visão, não há necessidade de armazenamento de qualquer outro dado existente na base de dados visto que não haverá qualquer proveito para o Lucene Gazetteer.

6.3 Avaliação

Podemos gerar oito diferentes Gazetteers a partir do cruzamento dos métodos e tecnologias apresentados. Nesta seção faremos uma avaliação de como cada um destes Gazetteers se comporta.

O processo de avaliação que fizemos foi bem simples. Como é inviável a verificação manual de todos os dados no Gazetteer em um tempo hábil, visto que cada um contém milhares de nomes, selecionamos um conjunto de nomes a fim de verificar se as URIs retornadas por cada Gazetteer são realmente URIs de Entidades Nomeadas com os nomes dados como entrada.

O primeiro grupo de nomes foi extraído de biografias existentes na própria base de dados Jamendo. O segundo grupo foi extraído de textos de trívia de alguns filmes listados como Top 250 no sítio web do IMDB[3]. Todos os nomes de ambos os grupos estão realmente relacionados a uma Entidade Nomeada representada na bases de dados e cada grupo contém 100 nomes. O primeiro grupo foi utilizado na busca por entidades na base Jamendo e o segundo na base Linked Movie Database.

Além da verificação da corretude na resposta, medimos também o tempo de processamento gasto por cada Gazetteer. Note que o teste de tempo não foi um teste exaustivo e tende apenas a medir a grandeza do tempo gasto. É importante notar ainda que, no caso do Lucene Gazetteer, estamos medindo apenas o tempo de processamento de cada consulta e não o tempo de indexação da base.

Em Gazetteers que utilizaram o método Limitado a propriedades-nome, a lista completa de propriedades-nome (a mesma utilizada como Gold Standard para a avaliação dos métodos de Reconhecimento de Propriedades-Nome) foi dada como entrada, a fim de que o erro no componente de reconhecimento de propriedades-nome não viesse a influenciar no desempenho do Gazetteer.

Os experimentos foram executados com o uso de dois computadores. O primeiro é um computador com um processador Intel i3 com 4 Gb de memória, responsável pelo Lucene Gazetteer e pela execução das consultas aos Gazetteers. O segundo

[3]http://www.imdb.com/chart/top

computador é um Intel Core 2 Duo com 2 Gb de memória responsável apenas pelo SPARQL Endpoint. Esta divisão visa espelhar um ambiente real onde existe um servidor externo responsável pelo SPARQL Endpoint e onde o computador local é quem efetua a consulta e a busca usando o Lucene Gazetteer.

É importante salientar que ao aplicarmos o SPARQL Gazetteer com o método Limitado a propriedades-nome optamos por realizar um pós-processamento ao invés de modificarmos a consulta SPARQL. Além disso, utilizamos o valor 10 para o parâmetro que indica o número de resultados a serem retornados por uma busca utilizando a Lucene, pois assumimos que é razoável encontrarmos até dez entidades com o mesmo nome. Este valor pode ser maior ou menor impactando nos resultados retornados (como veremos posteriormente).

Os Quadros 6.7 e 6.8 apresentam os resultados dos diferentes Gazetteers para as bases Jamendo e Linked Movie Database respectivamente. Nelas apresentamos o tempo total gasto na consulta pelos 100 nomes, a quantidade total de URIs retornadas para estas consultas e a taxa de acerto dentre estas URIs retornadas. Uma taxa de acerto de 100% indica que todas as URIs dadas como resposta estão corretas.

Utilizando dados de ambas as bases, o método utilizando Busca por texto completo não foi capaz de encontrar nomes que continham acentos ou traço, tais como a gravação “Tu bala me hará un favor” para a base Jamendo e a atriz “Carrie-Anne Moss” da base Linked Movie Database. O mesmo ocorreu no caso dos acentos com o método de Busca por Expressão Regular. Para a língua inglesa esta característica pode não ser um grande problema mas para idiomas como o português, o espanhol e o francês o reconhecimento de acentos é indispensável.

Analisando o tempo de execução, o SPARQL Endpoint em sua versão mais comum (sem full-text search) mostrou ser um empecilho para a utilização do Gazetteer em

Quadro 6.7: Resultado dos diversos Gazetteers aplicados à base de dados Jamendo

Gazetteer	Método	Tempo Total (ms)	URIs Retornadas	Taxa de Acerto
SPARQL (simples)	Geral	1390322	94	100%
SPARQL (simples)	Limitado a propriedades-nome	1001964	94	100%
SPARQL (expressão regular)	Geral	1409546	93	100%
SPARQL (expressão regular)	Limitado a propriedades-nome	1019365	93	100%
SPARQL (texto completo)	Geral	326089	93	100%
SPARQL (texto completo)	Limitado a propriedades-nome	1048	93	100%
Lucene	Geral	639	94	100%
Lucene	Limitado a propriedades-nome	144	94	100%

tempo real pois chegou a durar mais de 1 hora com a consulta simples e mais de 3 horas com a consulta utilizando expressões regulares. Em contraposição, todos os métodos otimizados para busca textual apresentaram um ótimo desempenho com relação ao tempo, chegando a resolução de 100 nomes em cerca de 1 segundo ou menos.

Com relação à taxa de acerto, a utilização do método Limitado a Propriedades-Nome foi fundamental para o desempenho dos Gazetteers. Este desempenho era o esperado visto que com este método limitamos o espaço de busca apenas aos textos que realmente representam nomes. Pudemos perceber ainda que houve uma taxa de acerto de 100% para todos os Gazetteers gerados a partir da base de dados

Quadro 6.8: Resultado dos diversos Gazetteers aplicados à base de dados Linked Movie Database

Gazetteer	Método	Tempo Total (ms)	URIs Retornadas	Taxa de Acerto
SPARQL (simples)	Geral	5741925	3457	5,58%
SPARQL (simples)	Limitado a propriedades-nome	5756830	193	100%
SPARQL (expressão regular)	Geral	12464043	3457	5,58%
SPARQL (expressão regular)	Limitado a propriedades-nome	12468853	193	100%
SPARQL (texto completo)	Geral	8733	3436	5,59%
SPARQL (texto completo)	Limitado a propriedades-nome	6483	192	100%
Lucene	Geral	4410	685	28,03%
Lucene	Limitado a propriedades-nome	1323	193	100%

Jamendo. Este fato ocorreu porque a base de dados tem poucas propriedades cujo valor é um literal e todas as propriedades que continham os nomes que procuramos são realmente propriedades-nome.

Uma última observação na avaliação é que, devido ao fato de que o Lucene Gazetteer exige que indiquemos o número máximo de resultados, algumas URIs corretas podem ser deixadas de fora. Este tipo de problema ocorreu quando utilizamos o método Geral com a Linked Movie Database. Outras URIs errôneas acabaram sendo melhor ranqueadas deixando uma URI correta de fora.

A partir dos dados de avaliação, os Gazetteers que demonstraram ser mais promisso-

res foram: o SPARQL Gazetteer com Busca por texto completo e método Limitado a propriedades-nome caso seja priorizado o uso de um SPARQL Endpoint em detrimento de um processamento local; e o Lucene Gazetteer também com o método Limitado a propriedades-nome por oferecer excelentes resultados no menor tempo.

6.4 Trabalhos Relacionados

Gazetteers foram durante muito tempo o gargalo no processo de Reconhecimento de Entidades Nomeadas pois costumavam ser gerados manualmente (KARKALETSIS et al., 1999) devido à pouca disponibilidade de conteúdo em meio digital. Com a popularização da internet e a participação dos usuários como geradores de conteúdo, a criação de Gazetteers tornou-se cada vez mais rápida. Esta rapidez se deu devido à capacidade de transformação de bases textuais já publicadas na Web em Gazetteers.

Dentre os Gazetteers mais comuns podemos citar aqueles que são construídos sobre bases de dados georreferenciados como é o caso do GeoNames[4] e do Alexandria Digital Library (HILL; FREW; ZHENG, 1999), que utiliza dados de fontes oficiais dos Estados Unidos, por exemplo.

Além das bases de dados georreferenciadas, outras bases de dados passaram a ser utilizadas como fonte para a geração de Gazetteers, como veremos a seguir.

Sporleder et al. (2006) estudou a viabilidade do reconhecimento de entidades em campos textuais presentes em bases de dados relacionais. Para este reconhecimento os autores utilizaram a própria base de dados como Gazetteer.

Cucerzan (2007) utiliza a estrutura da Wikipédia para formar um Gazetteer. Neste

[4]http://www.geonames.org/

trabalho, o autor extrai os nomes das entidades: dos títulos das diversas páginas da Wikipedia, dos títulos das páginas de redirecionamento, das páginas de desambiguação e dos links em outras páginas da Wikipédia. Com o Gazetteer em mãos, ele será usado, junto a outras características da base de dados, no processo de desambiguação de Entidades Nomeadas.

Mendes et al. (2011) e Hoffart et al. (2011) utilizam bases em Linked Data para a geração de Gazetteers. O primeiro utiliza a DBPedia, em conjunto com dados da Wikipedia, como fonte para um Gazetteeer a ser utilizado para a tarefa de Entity Linking, enquanto o segundo utiliza a base de dados YAGO para o mesmo propósito.

A diferença do nosso trabalho para os trabalhos de Mendes et al. (2011) e Hoffart et al. (2011) é que permitimos a utilização de toda a base de dados como fonte para os nomes ou um conjunto de propriedades-nome ao invés de utilizarmos uma única propriedade. O uso de outras propriedades-nome mantém ou melhora o número de acertos com relação a estes trabalhos. Além disso, nosso trabalho permite o uso de outras bases de dados além da DBPedia e a YAGO.

7 RECONHECIMENTO DE MENÇÕES A ENTIDADES NOMEADAS

Nos componentes apresentados nos capítulos anteriores definimos como a base de dados em Linked Data será utilizada para o processo de busca por Entidades Nomeadas. O próximo passo é encontrar os nomes destas Entidades Nomeadas citados no texto.

O Reconhecimento de Menções a Entidades Nomeadas consiste em identificar em um texto em Linguagem Natural quais os trechos que fazem menção a Entidades Nomeadas e quais os recursos que representam as entidades que podem estar sendo mencionadas. Para isso, criamos um algoritmo para identificação das menções no texto utilizando o Gazetteer gerado pelo componente de Geração de Dicionário de Nomes.

O componente de nossa arquitetura responsável pelo Reconhecimento de Menções a Entidades Nomeadas (Figura 7.1) recebe como entrada o texto que se quer processar e um Gazetteer para busca de nomes e identificação das entidades candidatas a anotar estes nomes. Como saída o componente prevê um mapa com a identificação de todas as menções no texto e a lista de identificadores (URIs) para as possíveis

entidades citadas por cada menção.

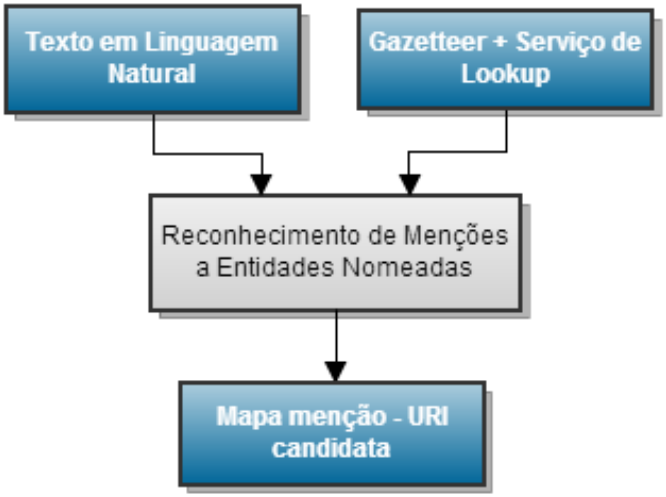


Figura 7.1: Entradas e Saídas esperadas para o componente de Reconhecimento de Menções a Entidades Nomeadas

Nas próximas seções apresentaremos o algoritmo criado para implementação deste componente e faremos uma avaliação de seu resultado utilizando as bases de dados Jamendo e Linked Movie Database.

7.1 Algoritmo

Para o reconhecimento de menções a Entidades Nomeadas no texto utilizamos um algoritmo simples baseado nos nomes existentes em um Gazetteer. O algoritmo divide-se em duas etapas principais:

1. Divisão do texto em tokens.

2. Busca por nomes.

A primeira etapa utiliza um tokenizador para dividir o texto em tokens. O tokenizador é um algoritmo que utiliza as principais características de um idioma para separar os diversos símbolos do texto em tokens. Estes tokens são palavras, pontuação e outras estruturas específicas de cada idioma. O nosso objetivo ao utilizá-lo é preparar o texto para ser processado, tornando cada palavra uma estrutura independente. O texto é dado ainda como uma sequência de caracteres para o tokenizador e, após processado, sai como uma lista sequencial de tokens (Figura 7.2). A essa lista sequencial de tokens damos o nome de texto tokenizado.

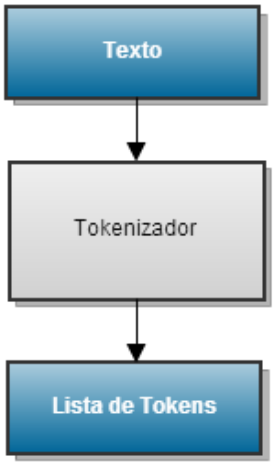


Figura 7.2: Entradas e Saídas esperadas para o Tokenizador

Este algoritmo prevê que o idioma do texto foi identificado previamente e que existe um tokenizador próprio para este idioma. Caso o tokenizador não exista, a opção é utilizar o tokenizador que utiliza o espaço como forma de separação dos tokens.

Em posse do texto tokenizado podemos então passar à busca por nomes. O algoritmo é baseado em uma janela deslizante que vai passando pelo texto a fim de identificar os trechos que são, na verdade, nomes de Entidades Nomeadas.

A janela inicial é formada por n tokens. Esses n tokens são transformados novamente em uma cadeia de caracteres. Caso a cadeia possua mais de 1 caracter então

ela será submetida ao Gazetteer a fim de verificar se constitui um nome. Se o Gazetteer informar que o nome não foi encontrado, então diminuímos a janela em 1 token, eliminando o último da sequência, e repetimos este processo. O processo de diminuição da janela continua até que a sequência forme um nome reconhecido pelo Gazetteer ou até que a janela fique vazia. Se a janela torna-se vazia, aumentamo-la para o tamanho inicial n e a deslizamos uma posição para a frente a partir do ponto inicial, recomeçando todo o processo. Se um nome for reconhecido, aumentamos a janela para o tamanho inicial n e deslizamos até a posição imediatamente posterior ao nome encontrado, recomeçando a busca por nomes. O pseudo-código fica de acordo com o Algoritmo 4 (supondo o tamanho da janela igual a 5).

O tamanho da janela depende muito do número de tokens que um nome pode ter e isto varia muito de um contexto para outro. Uma sugestão é a utilização do próprio Gazetteer para determinar qual o menor número de tokens necessário para reconhecer qualquer nome. De outra forma um número suficientemente grande pode ser utilizado.

Como exemplo da aplicação do algoritmo, iremos indicar os primeiros passos no processamento do texto *"The Michael Jackson was a great singer."*. Passando pelo processo inicial de tokenização o texto fica da seguinte forma:

[The, Michael, Jackson, was, a, great, singer, .]

O Gazetteer de nosso exemplo reconhece apenas os textos "Michael Jackson" e "Jackson" como nomes. Utilizaremos ainda o tamanho de janela igual a 3. As iterações do algoritmo acontecem da seguinte maneira:

- O índice está na posição inicial referenciando o token na posição 1 do texto tokenizado (Figura 7.3). Assumindo que escolhemos o tamanho da janela

Algoritmo 4 Reconhecimento de Menções a Entidades Nomeadas

```
textoTokenizado[1..n] ← lista sequencial de tokens
tamanhoJanela ← 5
indice ← 1
while indice < (n+1) do
    if indice + tamanhoJanela > n then
        tamanhoJanela ← n - indice
    end if
    nome ← textoTokenizado[indice, indice + (tamanhoJanela - 1)]
    if tamanho(nome) > 1 caracter then
        uris ← gazetteer()
        if uris está vazia then
            if tamanhoJanela > 1 then
                tamanhoJanela ← tamanhoJanela - 1
            else
                tamanhoJanela ← 5
                indice ← indice + 1
            end if
        else
            retorna textoTokenizado[indice, indice + tamanhoJanela - 1] como res-
posta
            indice ← indice + tamanhoJanela
        end if
    else
        tamanhoJanela ← 5
        indice ← indice + 1
    end if
end while
```

igual a 3, o texto que estará contido na janela será o texto composto pelos três primeiros tokens.

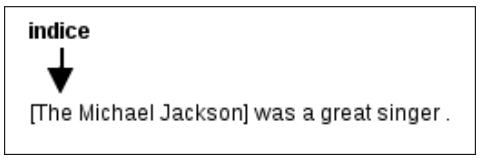


Figura 7.3: Estado da Primeira Iteração do Algoritmo de Reconhecimento de Menções a Entidades Nomeadas.

- Transformamos a sequência de tokens dentro da janela em uma sequência de caracteres e submetemos o texto "The Michael Jackson" para o Gazetteer.
- O Gazetteer não identifica o nome, retornando uma lista vazia como resposta.
- Como o nome não foi encontrado, diminuímos o tamanho da janela em 1 ficando agora com o texto "The Michael" dentro da janela (Figura 7.4). Submetemos o novo texto ao gazetteer e recebemos novamente uma lista vazia como resposta.

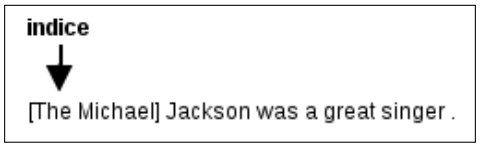


Figura 7.4: Segunda Iteração do Algoritmo de Reconhecimento de Menções a Entidades Nomeadas.

- Repetimos o processo diminuindo mais uma vez a janela e submetendo o texto "The" ao gazetteer. (Figura 7.5)
- O Gazetteer retorna mais uma vez uma lista vazia mas desta vez ao diminuirmos a janela, ela passa a não contêr tokens. Sendo assim, voltamos a janela ao seu tamanho inicial de 3, atualizamos o índice para a posição 2 e deslizamos o início da janela para a posição apontada pelo índice. (Figura 7.6)

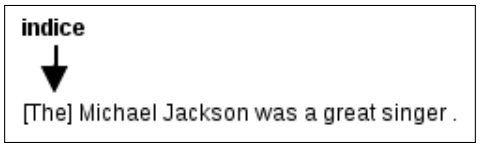


Figura 7.5: Terceira Iteração do Algoritmo de Reconhecimento de Menções a Entidades Nomeadas.

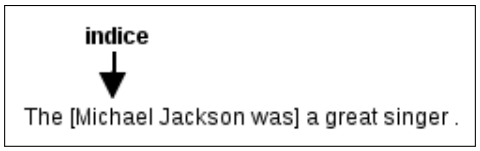


Figura 7.6: Quarta Iteração do Algoritmo de Reconhecimento de Menções a Entidades Nomeadas.

- O texto dentro da janela agora é "Michael Jackson was". Mais uma vez o nome não é encontrado, diminuindo o tamanho da janela e deixando agora o texto "Michael Jackson" (Figura 7.7).

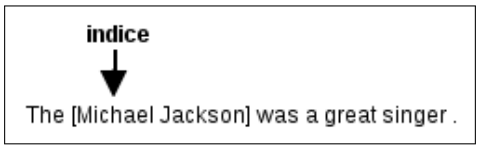


Figura 7.7: Quinta Iteração do Algoritmo de Reconhecimento de Menções a Entidades Nomeadas.

- O texto é novamente submetido ao Gazetteer, mas desta vez uma lista de URIs é retornada como resposta indicando que o nome foi encontrado. Sendo assim, o nome e a lista de URIs são retornados como resposta do algoritmo e o algoritmo continua a busca por outros nomes.

- Neste momento a janela não diminui, pois o nome foi encontrado. O tamanho da janela retorna ao seu tamanho original igual a 3, o índice que estava na posição 2 pula para a primeira posição após o nome encontrado, passando a valer 4. O texto que agora está contido dentro da janela é "was a great" (Figura 7.8).

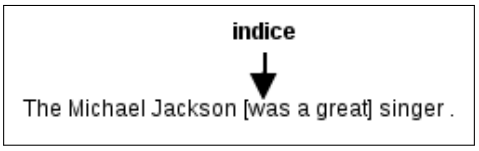


Figura 7.8: Sexta Iteração do Algoritmo de Reconhecimento de Menções a Entidades Nomeadas.

- O Gazetteer outra vez não reconhece o nome. A busca continua até que o texto dentro da janela seja igual a ".". Quando o nome não é encontrado, a janela fica vazia e o algoritmo é encerrado por não haver mais texto a ser processado.

Ao final da execução do algoritmo para Reconhecimento de Menções a Entidades Nomeadas temos um conjunto de trechos textuais, cada qual relacionado a um conjunto de URIs que podem estar sendo referenciadas por aquela menção. Nesta etapa ainda não sabemos quais são as Entidades Nomeadas realmente citadas no texto mas já temos um conjunto de Entidades Candidatas. Estas entidades devem passar ainda por um processo de desambiguação posterior.

7.2 Avaliação

Como forma de avaliar a eficácia do nosso algoritmo, recolhemos um conjunto de textos que serão submetidos ao algoritmo criado e verificamos se os trechos textuais identificados como nomes são realmente nomes no contexto do texto.

Reconhecemos como corretos todos os trechos textuais reconhecidos como nomes que: referem-se ao primeiro nome de uma pessoa, referem-se ao último nome de uma pessoa, representam o nome completo que está citado no texto e não apenas uma parte. Por exemplo, "Jimi", "Hendrix" e "Jimi Hendrix" são nomes válidos por serem trechos que fazem referência ao primeiro nome de um artista, ao último nome

ou ao nome completo pelo qual é conhecido. O mesmo não ocorre quando temos no texto o nome "Yellow Cactus" (um estúdio musical) mas apenas parte do nome ("Cactus", por exemplo) sendo reconhecida.

A avaliação foi feita a partir de 66 biografias extraídas da propriedade http://purl.org/ontology/mo/biography presente em recursos da base de dados Jamendo. Estas biografias foram submetidas ao Reconhecimento de Entidades Nomeadas utilizando a base de dados Jamendo como fonte para o Gazetteer. A segunda parte da avaliação contou com 28 textos de trívia (comentários curtos) do sítio web do IMDB[1]. Para esta segunda base de textos utilizamos a Linked Movie Database para o Gazetteer. É importante notar que todos os textos do IMDB estão em inglês enquanto para o Jamendo temos 62 textos em inglês, 3 em espanhol e 1 em português.

Antes da execução do algoritmo utilizamos o tokenizador disponível na biblioteca OpenNLP (APACHE, 2010) para tokenizar todos os textos.

Como Gazetteer utilizamos os dois melhor avaliados: o SPARQL Gazetteer com busca por texto completo e limitado a propriedades-nome; e o Lucene Gazetteer limitado a propriedades-nome. Em ambos os casos executamos o algoritmo em sua forma comum e o executamos também assumindo que todo nome deve iniciar por uma letra maiúscula. Nesta segunda versão, um trecho textual é submetido para verificação do Gazetteer apenas se a sua primeira letra está em maiúsculo, senão o algoritmo prossegue assumindo que aquele trecho não é um nome.

Assumimos que é razoável que um nome tenha até dez tokens, sendo assim utilizamos um tamanho de janela igual a 10.

[1] http://www.imdb.com/

7.2.1 Jamendo

A primeira avaliação foi feita utilizando as biografias da base Jamendo e a própria Jamendo como fonte dos dados para o Gazetteer. O resultado da avaliação pode ser visto na Tabela 7.1. Nesta tabela indicamos o Gazetteer utilizado, o tempo médio de processamento de cada biografia, o número médio de menções identificadas em cada uma e o número médio de menções corretamente identificadas.

Quadro 7.1: Resultado da Aplicação do Algoritmo de Reconhecimento de Menções a Entidades Nomeadas às Biografias da base Jamendo

Gazetteer	Apenas nomes próprios	Tempo médio execução	Média de resultados	Média de resultados corretos
SPARQL	Não	45571 ms	42,32	3,71 (8,77%)
SPARQL	Sim	3711 ms	8,41	3,58 (42,57%)
Lucene	Não	271 ms	24,54	3 (12,22%)
Lucene	Sim	159 ms	7,02	2,89 (41,17%)

A precisão da busca por nomes é muito dependente do Gazetteer utilizado. Como o nosso Lucene Gazetteer foi construído diferenciando maiúsculas e minúsculas, bem como presença e ausência de acentos, faz-se necessário que o nome no texto seja exatamente igual ao nome na base de dados não podendo trocar uma letra minúscula por maiúscula ou vice-versa. Enquanto isso, o SPARQL Gazetteer permite uma busca que independe se as letras são maiúsculas ou minúsculas.

A busca utilizando apenas nomes próprios auxiliou muito na porcentagem de resultados corretos, porém impediu que nomes iniciados por letras minúsculas fossem reconhecidos. Em caso de escrita informal é comum que certas regras da escrita sejam deixadas de lado, sendo assim seria interessante que não houvesse a necessidade de que um nome tivesse a primeira letra maiúscula para ser reconhecido.

O grande problema no uso da Jamendo como base de dados para anotação do texto

é que a base contém diversos nomes considerados incomuns. Textos como “Before”, “After”, “It”, dentre outros, são encontrados como nomes para entidades existentes na base. A Jamendo apresenta ainda sinais de pontuação corretamente identificados como nomes para músicas, por exemplo, o que é uma característica singular desta base. Apesar desta característica, este último caso não será reconhecido por nosso algoritmo, visto que submetemos aos Gazetteers apenas nomes com mais de 1 caracter.

7.2.2 Linked Movie Database

Utilizamos os textos de trívia extraídos do sítio web IMDB para a nossa segunda avaliação com os Gazetteers gerados a partir da Linked Movie Database. Os resultados podem ser encontrados no Quadro 7.2. Nesta tabela indicamos o Gazetteer utilizado, o tempo médio de processamento de cada texto, o número médio de menções identificadas em cada um e o número médio de menções corretamente identificadas.

Quadro 7.2: Resultado da Aplicação do Algoritmo de Reconhecimento de Menções a Entidades Nomeadas aos textos do IMDB utilizando a Linked Movie Database

Gazetteer	Apenas nomes próprios	Tempo médio execução	Média de resultados	Média de resultados corretos
SPARQL	Não	44552 ms	7,61	6,28 (82,52%)
SPARQL	Sim	3426 ms	7,04	6,14 (87,22%)
Lucene	Não	1324 ms	7,64	6,28 (82,20%)
Lucene	Sim	407 ms	7,04	6,18 (87,78%)

Mais uma vez o reconhecimento apenas de trechos textuais iniciados por letra maiúscula auxiliou na precisão dos nomes encontrados no texto. O que eliminou, por acaso, alguns resultados como “novel” ou “producer” que não são verdadeiramente nomes mas que encontram-se na base de dados descritos como nomes de recursos que fazem referência a Entidades Nomeadas.

Esta base tem, no geral, nomes melhor construídos e que não se assemelham a pontuação, artigos e pronomes. Da mesma forma, os textos assemelham-se aos nomes apresentados pelo Gazetteer sem diferenciação em maiúsculas e minúsculas. Por este motivo, os diferentes Gazetteers obtiveram resultados muito próximos, modificando apenas o tempo de execução que foi aceitável em todos os casos avaliados.

7.2.3 Análise Comparativa

Como o objetivo do componente de Reconhecimento de Menções a Entidades Nomeadas é mapear quais os trechos textuais que se referem a Entidades Nomeadas utilizando o Gazetteer como base de conhecimento, todas as variações do algoritmo gerado para este componente obtiveram sucesso.

Apesar do sucesso apresentado, notamos que a eficácia do algoritmo depende muito do Gazetteer utilizado e, sobretudo, da qualidade dos dados que estão presentes na base de dados. Se os dados são de baixa qualidade, o algoritmo irá reconhecer uma série de palavras como nomes de forma errônea, enquanto pode ou não reconhecer um verdadeiro nome dependendo da construção do Gazetteer. No caso do Lucene Gazetteer, alguns nomes não puderam ser reconhecidos pois ele utiliza exatamente a forma como o nome está na base de dados, impedindo variações de letras maiúsculas e minúsculas. Já o SPARQL Gazetteer obteve grande sucesso neste quesito, porém em um tempo de execução consideravelmente maior utilizando ambas as bases de dados.

De uma forma geral, o algoritmo em suas diferentes variações permitiu o mapeamento de entidades candidatas a anotarem o texto, tendo melhores resultados ao utilizar a Linked Movie Database do que a Jamendo devido às características das bases de dados. De toda forma necessitamos de mais um passo para resolver quais das

entidades candidatas realmente devem ser utilizadas para anotação do texto. Este último passo será feito pelo componente de Desambiguação de Entidades Nomeadas.

Embora necessitemos de mais dados para avaliação, pudemos perceber que nossa abordagem é plenamente aceitável para uso com outros idiomas tais quais português, francês e espanhol, além do próprio inglês que foi nosso foco principal. Esta característica se dá por não utilizarmos ferramentas muito dependentes de idioma. A única ferramenta que tem algum tipo de influência dada pelo idioma é o tokenizador e mesmo assim podemos utilizar um tokenizador genérico que utilize apenas os espaços e pontuação como separação dos tokens.

7.3 Trabalhos Relacionados

Existem diversos trabalhos na área de Entity Linking que trabalham na busca por entidades nomeadas candidatas a anotarem um determinado texto. Dentre os principais trabalhos podemos identificar os que usam a Wikipedia como fonte para as entidades nomeadas e os que utilizam bases de dados em Linked Data.

Um dos trabalhos pioneiros na área de Entity Linking utilizando a Wikipédia foi o trabalho de Cucerzan (2007). Nele, o autor apresenta um algoritmo que utiliza regras de capitalização e estatísticas para identificar o que é uma menção e com qual identificador da Wikipédia essa menção pode ser anotada. Depois de seu trabalho surgiram ainda diversos outros, tais quais: Milne e Witten (2008) que utilizam técnicas de aprendizado de máquina para reconhecimento das menções e escolha das entidades candidatas; Kulkarni et al. (2009) que utilizam o texto ao redor de uma menção para decidir quais as entidades candidatas para aquela menção; e Dredze et al. (2010) que utilizam um conjunto de regras para identificar a menção no texto, permitindo identificação de variações nos nomes.

Dentre os trabalhos que utilizam bases de dados em Linked Data podemos citar o DBPedia Spotlight, o AIDA e o NERSO.

O DBPedia Spotlight (MENDES et al., 2011) utiliza o LingPipe Exact Dictionary-Based Chunker(ALIAS-I,), um algoritmo que trabalha de forma muito semelhante ao algoritmo que criamos aqui. Neste trabalho, os autores tentam casar o texto com os nomes expressos no Gazetteer, que no caso do DBPedia Spotlight é o DBPedia Lexicalization (MENDES; JAKOB; BIZER, 2012). Se o nome for encontrado, uma série de URIs relacionadas àquele nome são retornadas como URIs candidatas. Além disso o DBPedia Spotlight permite que o usuário indique um número de suporte, ou seja, o número mínimo de links que a respectiva Entidade Nomeada deve ter na Wikipédia para que seja aceita como uma entidade nomeada candidata.

A diferença no uso do LingPipe com relação ao nosso trabalho é que ele permite uma série de outras configurações. Uma delas permite a verificação de uma mesma palavra em mais de um nome. Por exemplo, no texto "George Michael Jackson" ele pode reconhecer "George Michael" como uma entidade nomeada e "Michael Jackson" também, ou seja, a palavra Michael foi testada mais de uma vez em busca de nomes. Em nosso método, ao encontrar o nome "George Michael" excluimos a palavra "Michael" do resto da busca.

O AIDA (HOFFART et al., 2011) utiliza o Stanford NER para reconhecimento das menções no texto. Cada menção é cruzada com os nomes presentes em um Gazetteer gerado a partir da YAGO.

O NERSO (HAKIMOV; OTO; DOGDU, 2012) utiliza o mesmo algoritmo que utilizamos neste trabalho. Ele utiliza uma janela que desliza pelo texto a fim de encontrar os nomes que estão no Gazetter, escolhendo uma janela de tamanho igual a 4. Seu Gazetteer utiliza a DBPedia como fonte para os nomes.

A diferença do nosso trabalho para todos os trabalhos levantados é que utilizamos bases de dados em Linked Data como fonte para os dados sem que precisemos efetivamente modificar os dados ou utilizar uma fonte alternativa para identificação das menções.

8 DESAMBIGUAÇÃO DE ENTIDADES NOMEADAS

Na maioria dos idiomas, sobretudo na língua inglesa, podemos identificar uma série de palavras polissêmicas, ou seja, que apresentam significados diversos dependendo do contexto em que aparecem (KONCHADY, 2006, p. 8-9). Da mesma forma que palavras comuns do idioma, nomes também podem ser tidos como polissêmicos porque podem fazer referência a mais de uma Entidade Nomeada dependendo do contexto em que se apresentam.

O componente de Reconhecimento de Menções a Entidades Nomeadas foi responsável por identificar menções (nomes) no texto e as entidades candidatas a anotar cada uma destas menções. A seguir temos de identificar, dentre todas as entidades candidatas, qual a que está sendo realmente referenciada por cada menção. A este processo de resolução chamamos Desambiguação de Entidades Nomeadas.

O processo de desambiguação é necessário sempre que uma menção textual pode estar fazendo referência a mais de uma Entidade Nomeada. Este processo consiste em identificar, através do contexto dado pelo texto, qual a real entidade que está sendo citada.

Em nossa arquitetura, o componente de Desambiguação de Entidades Nomeadas (Figura 8.1) recebe como entrada o mapa de entidades candidatas para cada menção textual e a base de dados em Linked Data que foi utilizada para identificar estas entidades candidatas. A saída esperada para este componente é um novo mapa com as menções junto aos identificadores da Entidades Nomeadas realmente identificadas por elas.

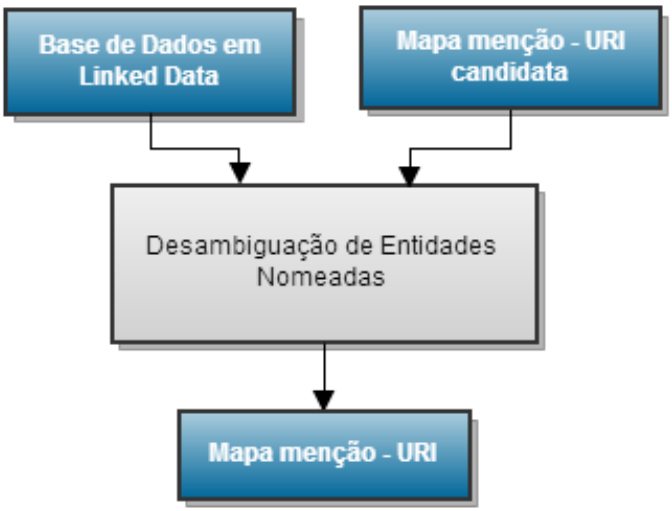


Figura 8.1: Entradas e Saídas esperadas para o componente de Desambiguação de Entidades Nomeadas

O método que utilizamos para implementação deste componente foi totalmente baseado no trabalho de Han, Sun e Zhao (2011). Neste trabalho, os autores utilizam a Wikipédia como base de conhecimento para Desambiguação de Entidades Nomeadas através da comparação entre o contexto expresso pelo texto e o contexto dado pelas relações entre as páginas da Wikipédia. Nossa abordagem é uma adaptação deste trabalho a bases de dados em Linked Data.

Na seção 8.1 detalhamos as principais ideias do trabalho de Han, Sun e Zhao (2011) e explicamos como esta abordagem pode ser modificada para uso com bases em

Linked Data na seção 8.2. Em seguida, na seção 8.3, explicamos o passo-a-passo no processamento do texto a fim de efetivamente realizar a desambiguação, apontando sempre as diferenças adotadas por nosso trabalho. Na seção 8.4 apresentamos os resultados da avaliação do método. Ao final, na seção 8.5, apresentamos os trabalhos relacionados ao processo de Desambiguação de Entidades Nomeadas.

8.1 Identificação de Entidades através da Análise do Contexto

O trabalho de Han, Sun e Zhao (2011) parte de duas premissas principais:

1. O contexto de uma entidade citada no texto é dado por todas as outras entidades citadas no mesmo texto.
2. A vizinhança de uma entidade na base de dados indica o contexto em que aquela entidade se insere.

Para a primeira premissa, os autores indicam um exemplo onde os nomes "Michael Jordan", "NBA" e "Space Jam" aparecem juntos em um mesmo texto. O objetivo é saber se o nome "Michal Jordan" faz referência a um jogador de basquete, a um professor de Berkeley ou a um ator americano. Os autores partem da ideia de que todas as palavras que aparecem juntas em um texto formam um contexto que dá significado a todas elas. Neste caso, poderíamos identificar que o nome faz referência a um jogador de basquete se temos conhecimento de que este jogador atua na NBA, que é a liga americana de basquete, e atuou no filme Space Jam.

A ideia expressa por esta primeira premissa foi apresentada inicialmente por Saussure (apud BIEMANN, 2012) e é base para a grande maioria dos trabalhos em de-

sambiguação em textos (Vide Seção 8.5).

Para exemplificar o que queremos dizer, iremos apresentar um dos textos que serão posteriormente utilizados em nosso processo de avaliação. Este texto foi extraído de trívias do site IMDB e trata de informações acerca do filme "The Gladiator" e está expresso no Quadro 8.1.

Quadro 8.1: Texto de Exemplo para o processo de desambiguação. Extraído de http://www.imdb.com/title/tt0172495/trivia

> *"Richard Harris, who plays Marcus Aurelius, was originally set to play Commodus in The Fall of the Roman Empire (but left the film due to artistic differences with director Anthony Mann and was replaced by Christopher Plummer)."*

Mesmo um leitor que não tenha conhecimento sobre o filme é capaz de identificar as entidades presentes no texto e a que tipo de classe elas pertencem. Esta característica ocorre justamente pelo contexto que as palavras dão umas às outras. Um leitor poderia facilmente identificar que o ator "Richard Harris" atuou como o personagem "Marcus Aurelius" e foi cogitado para atuar como o personagem "Commodus" no filme "The Fall of the Roman Empire". Pode ainda notar que "Richard Harris" e "Anthony Mann" possuem algumas diferenças de opinião artística e que "Christopher Plummer" foi quem finalmente atuou como o personagem "Commodus". Sendo assim, se precisássemos definir qual das pessoas chamadas "Richard Harris" em todo o mundo está realmente sendo citada pelo texto, tentaríamos descobrir qual delas atuou como um personagem cujo nome é "Marcus Aurelius", por exemplo.

É então neste ponto que utilizamos a segunda premissa. Mesmo que tenhamos identificado que "Richard Harris" é um ator, não temos como saber quem é esta pessoa dentro de uma lista de pessoas reconhecidas por este mesmo nome, a menos que tenhamos algum conhecimento prévio. Em termos computacionais este conhecimento prévio será dado pela base de dados utilizada. Quanto mais informação a base de

dados puder nos prover, melhor será o processo de desambiguação.

No trabalho de Han, Sun e Zhao (2011) a base de dados utilizada foi a Wikipédia. Neste trabalho, cada página comum da Wikipédia foi tida como representando uma entidade cujo nome é dado pelo título da página. Assim, relacionamentos entre duas entidades são caracterizados por citações a uma entidade no texto da página de outra. Em outras palavras, se existe uma página na Wikipédia para o jogador "Michael Jordan" e existe uma citação a ele no texto da página da entidade "NBA" então isto indica que existe um relacionamento entre as entidades "Michael Jordan" e "NBA" na Wikipédia.

Outra característica, dada pela primeira premissa, é que, se duas entidades aparecem no mesmo texto então existe algum tipo de relacionamento de contexto entre as duas. Dessa forma, se duas entidades aparecem no texto de uma terceira, esta terceira entidade indica que as outras duas estão em um mesmo contexto. Ou seja, se as palavras "NBA" e "Michael Jordan" aparecem no texto de descrição da entidade "Space Jam" então existe um relacionamento entre "NBA" e "Michael Jordan" dado pela entidade "Space Jam".

No caso da Wikipédia, não temos como identificar diretamente o tipo de relação semântica existente entre duas entidades. De qualquer forma, podemos identificar o quão relacionadas duas entidades são pelo número de relacionamentos que possuem entre si e que possuem através de outras entidades diretamente relacionadas a ambas (WITTEN; MILNE, 2008). Mesmo que uma entidade na base de dados não esteja diretamente relacionada a outra, a existência de entidades que interligam as duas representa uma medida de proximidade semântica que indica que as duas entidades aparecem em um mesmo contexto. Um exemplo para isto está apresentado na Figura 8.2 onde mostramos as entidades relacionadas a "Automobile" (automóvel) e "Global Warming" (aquecimento global) na Wikipédia em inglês. Note que todas

as entidades que se relacionam a ambas as entidades estão inseridas em um mesmo contexto, que trata de combustíveis e poluição, enquanto entidades que não estão ligadas às duas indicam contextos diversos em que apenas uma das entidades se insere.

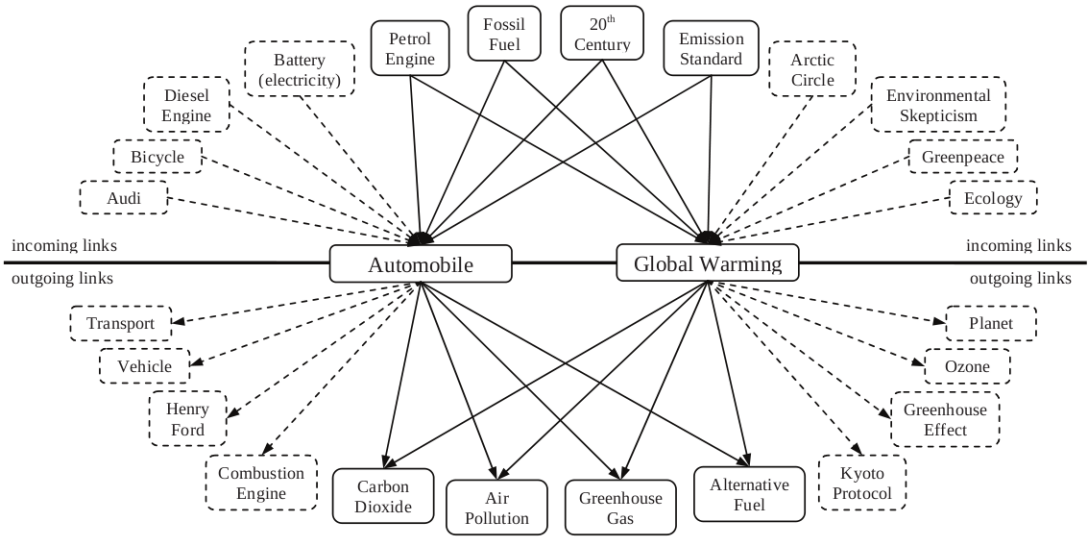


Figura 8.2: Vizinhança das entidades *Automobile* e *Global Warming* construída através de dados extraídos da Wikipédia em inglês. Imagem extraída de (WITTEN; MILNE, 2008)

8.2 Identificação do contexto em bases de dados em Linked Data

O trabalho de Han, Sun e Zhao (2011) pode ser adaptado para qualquer base de dados onde exista alguma forma de identificar as entidades e os relacionamentos que expressam o contexto em que estas entidades aparecem.

Ao contrário da Wikipédia, bases de dados em Linked Data contam com a explicitação formal dos relacionamentos entre entidades, além da identificação do tipo destes relacionamentos e seus significados. Para utilizar a abordagem com bases em Linked

Data precisamos adaptá-la às características deste tipo de base de dados, além de verificar se os relacionamentos são efetivamente relacionamentos que expressam o contexto de existência das entidades.

Em termos gerais, podemos identificar dois tipos de relacionamento: os ***relacionamentos funcionais*** e os ***relacionamentos de domínio***.

Os relacionamentos funcionais tratam de informações de estruturação da base, como relações classe-instância e relações de equivalência entre recursos. Dentre propriedades que expressam este tipo de relacionamento podemos citar *rdf:type*, *owl:sameAs* e *rdfs:seeAlso*. Relacionamentos deste tipo nos dão informação sobre a base de dados mas não são interessantes em nossa abordagem de desambiguação.

Relacionamentos de domínio são aqueles que definem a vizinhança de uma entidade, ou seja, são os que expressam um padrão de ligações que indica o contexto de existência de uma dada entidade. Este segundo tipo de relacionamento é o que efetivamente utilizamos na abordagem de Desambiguação de Entidades Nomeadas.

Para melhor evidenciar a possibilidade do uso de bases em Linked Data com um número razoável de relacionamentos de domínio, mostraremos um exemplo utilizando a Linked Movie Database com cerca de 1 relacionamento de domínio por classe.

Selecionamos a entidade cujo nome é "Marcus Aurelius" no texto do Quadro 8.1. Verificamos que existem duas entidades com este nome na base de dados (um ator e um personagem). Nosso objetivo é verificar se o contexto dado pela base de dados em Linked Data para uma entidade reflete o contexto dado pelo texto em linguagem natural. Assim, verificaremos a vizinhança (direta e através dos vizinhos) de cada uma das entidades candidatas à procura das outras entidades citadas no texto. Se outras entidades do texto aparecem relacionadas exclusivamente a uma

das duas entidades candidatas, então podemos concluir que o contexto dado pela base é coerente com o contexto dado pelo texto.

O resultado da construção da vizinhança das duas entidades candidatas está demonstrado na Figura 8.3. Nesta figura, os nós que indicam as duas entidades candidatas aparecem em preto junto com sua vizinhança e os vizinhos de seus vizinhos, ou seja, as entidades que relacionam duas entidades indicando seu contexto, como no método de Han, Sun e Zhao (2011). As outras entidades que estão sendo referenciadas no texto aparecem em listrado e, como podemos notar, aparecem, de fato, relacionadas a apenas uma das entidades candidatas. Esta entidade é aquela que está sendo realmente citada no texto e deverá ser o resultado retornado pelo processo de desambiguação para o nome "Marcus Aurelius".

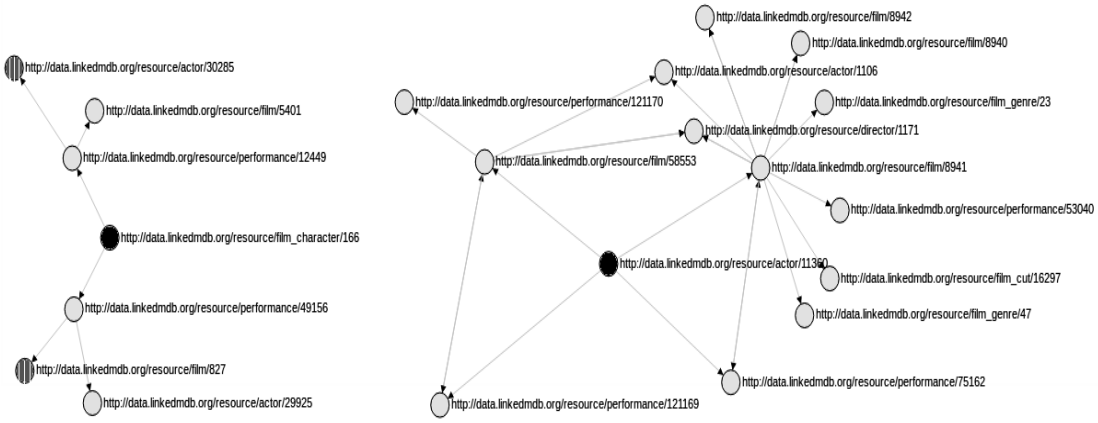


Figura 8.3: Vizinhança das entidades candidatas a anotar o texto *Marcus Aurelius*. As entidades candidatas aparecem em preto e outras entidades citadas no mesmo texto aparecem em listrado.

8.3 Método

Neste componente veremos a base de dados em Linked Data como um grafo representado apenas pelos relacionamentos entre recursos. Utilizaremos, portanto, a

seguinte definição:

Definição 8.1 *Uma **base de dados em Linked Data** do ponto de vista do relacionamento entre seus recursos pode ser definida como um grafo* $G_{LD} = (V_{LD}, A_{LD})$ *onde* $V_{LD} = \{v_i \mid v_i =$ *recurso na base em Linked Data*$\}$ *e* $A_{LD} = \{(a_i, a_j) \mid a_i, a_j \in V_{LD};\ a_i \neq a_j\}$. *Cada membro de* A_{LD} *é um par ordenado que pode ser visto como uma aresta direcionada que representa o relacionamento entre dois recursos presentes na base de dados.*

Para melhor compreensão do algoritmo a ser apresentado precisamos ainda de algumas definições adicionais. Uma delas é a definição do Grafo de Contexto do Texto (G_{MEc}) dado pelas menções e entidades candidatas a anotar cada menção. A segunda definição é o Grafo de Contexto da base em Linked Data (G'_{LD}). Por último, temos o Grafo de Referência que é a união do grafo de contexto do texto com o grafo de contexto da base de dados em Linked Data. Este último grafo é construído durante a execução do método e é o que efetivamente permite a realização da desambiguação.

Definição 8.2 *O **Grafo de Contexto do Texto** é dado por* $G_{MEc} = (V_{MEc}, A_{MEc})$ *onde* $V_{MEc} = V_M \cup V_{Ec}$. $V_M = \{ m_i \mid m_i =$ *menção identificada no texto*$\}$ *e* $V_{Ec} = \{ e_{ci} \mid e_{ci} =$ *entidade candidata a anotar* $m_i\}$ *com* $V_M \cap V_{Ec} = \emptyset$. *O conjunto* A_{MEc} *é formado por arestas que ligam as menções identificadas no texto às suas entidades candidatas, assim,* $A_{MEc} = \{(a_i, a_j) \mid a_i \in V_m;\ a_j \in V_{Ec}\ \}$.

Definição 8.3 *O **Grafo de Contexto da Base de Dados em Linked Data** é dado por* $G'_{LD} = (V'_{LD}, A'_{LD})$. *Os vértices desse grafo são exatamente as entidades candidatas a anotar as menções do texto, ou seja,* $V'_{LD} = V_{Ec} \subseteq V_{LD}$. *As arestas representam as relações semânticas de contexto existentes entre estas entidades*

candidatas, sendo assim, cada aresta (a_i, a_j) possui uma aresta (a_j, a_i) contrária representando a mesma relação. $A'_{LD} = X \cup Y$ *onde* X *representa o conjunto de relacionamentos existentes =* $\{(a_i, a_j) \mid a_i, a_j \in V_{Ec}; a_i \neq a_j ; (a_i, a_j) \in A_{LD} \vee (a_i,x) , (x,a_j) \in A_{LD} \}$ *e* Y *representa as arestas em sentido oposto=*$\{(a_i, a_j) \mid (a_j, a_i) \in X\}$.

Definição 8.4 *O* ***Grafo de Referência*** $G = (V, A)$ *é dado por* $G = G_{MEc} \cup G'_{LD}$.

O método consiste em um processo de desambiguação global, ou seja, efetuamos a desambiguação de todas as entidades ao mesmo tempo, utilizando todas as menções detectadas no texto. O processo pode ser divido em 5 etapas (HAN; SUN; ZHAO, 2011):

1. Geração do Grafo de Referência Inicial.
2. Cálculo de Similaridade Semântica entre Entidades Candidatas.
3. Cálculo do Grau de Importância das Menções.
4. Cálculo das Probabilidades de Propagação das Evidências.
5. Inferência Coletiva.

Na primeira etapa geramos um grafo que irá representar o contexto extraído do texto e o contexto dado pela base de dados. O contexto do texto é formado pelas menções e suas entidades candidatas, enquanto o contexto da base é formado pelos relacionamentos semânticos existentes entre as entidades candidatas. Na segunda

etapa calculamos a força de cada um dos relacionamentos, ou seja, o quão relacionadas duas entidades estão no contexto dado pela base. A seguir, calculamos a importância de cada uma das menções para o texto e atribuímos um valor a cada uma no grafo de referência inicial. Na etapa seguinte, calculamos a probabilidade de propagação desta importância para todas as entidades candidatas presentes no grafo para, na última etapa, executarmos um algoritmo de passeio aleatório que irá realizar a inferência coletiva de todas as entidades que são referenciadas por cada menção.

Cada etapa do processo de desambiguação será detalhada em uma das subseções seguintes. Em cada uma explicaremos o funcionamento original de cada etapa e identificaremos as adaptações para o uso de Linked Data, quando existente.

8.3.1 Geração do Grafo de Referência

O Grafo de Referência é, em suma, o grafo que representa as ligações entre todas as menções e as entidades candidatas a anotá-las, bem como todas as relações semânticas entre entidades candidatas. A primeira etapa do processo de desambiguação consiste em gerar este grafo a partir da base de dados em Linked Data e do Mapa menção-URIs candidatas dados como entrada.

A fim de exemplificar a geração de um grafo de referência, iremos mostrar o passo-a-passo executado a partir do texto sobre o Filme “The Gladiator” presente no Quadro 8.1. As menções previamente identificadas foram: “Richard Harris", “Marcus Aurelius", “Commodus", “The Fall of the Roman Empire”, “Anthony Mann"e “Christopher Plumm".

Cada uma das menções identificadas no texto gera um vértice $v_m \in V_M$ no grafo de

referência G (Figura 8.4). A partir daí, inserimos no grafo todos os vértices $v_{ec} \in V_{Ec}$ que representam entidades candidatas (Figura 8.5).

Figura 8.4: Grafo de Referência apenas com os vértices representando as menções a Entidades Nomeadas.

Neste ponto, já temos todo o conjunto de vértices do Grafo de Referência, faltando especificar que arestas irão compôr o grafo. O Grafo de Referência é composto por dois tipos de aresta:

- $a_{MC} = (a_i, a_j) \in V_{MEc}$, ligando menções a entidades candidatas.
- $a_{CC} = (a_i, a_j) \in A'_{LD}$, ligando entidades candidatas semanticamente relacionadas.

Cada aresta do grafo possui ainda uma medida relacionada à compatibilidade entre sua fonte e seu destino. Para as arestas a_{MC} temos a compatibilidade local menção-entidade e para as arestas a_{CC} temos a similaridade semântica.

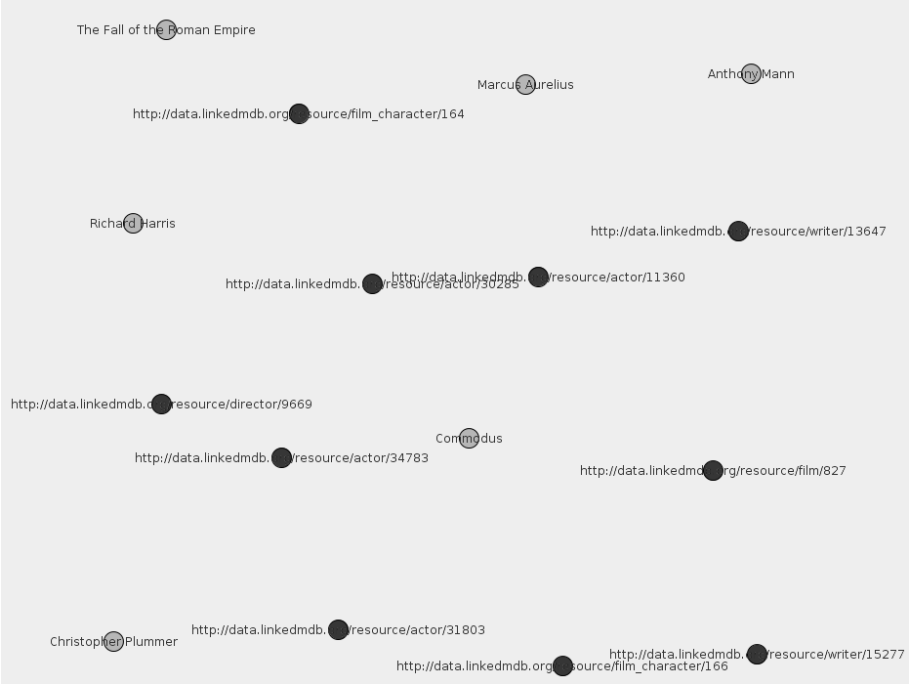


Figura 8.5: Grafo de Referência com todos os vértices (menções em cinza claro e entidades candidatas em cinza escuro).

8.3.2 Cálculo da Compatibilidade Menção-Entidade

A Compatibilidade Menção-Entidade CP(m,c) indica o quanto uma entidade candidata está apta a anotar uma determinada menção.

Originalmente o valor de CP(m,c) é dado pela coocorrência de termos no texto do entorno da menção e no texto de descrição da entidade candidata. Porém, como nem toda base de dados em Linked Data provê um texto de descrição para seus recursos, o uso desta medida torna-se inviável. Sendo assim, a **compatibilidade local menção-entidade CP(m,c)** será calculada atribuindo um valor igualitário a todas as entidades candidatas relacionadas a uma mesma menção, não dando qualquer privilégio a nenhuma das entidades candidatas (Fórmula (8.1)).

$$CP(m, c) = \frac{1}{|c|} \tag{8.1}$$

Na Fórmula (8.1) temos que $|c|$ é o número absoluto de entidades candidatas relacionadas à menção **m**. Sendo assim, o valor de CP para todas as arestas e_{mc} ligadas à menção "Richard Harris", por exemplo, será igual a $\frac{1}{2}$.

Adicionando este conjunto de arestas ao nosso grafo de referência ficamos com o grafo da Figura 8.6.

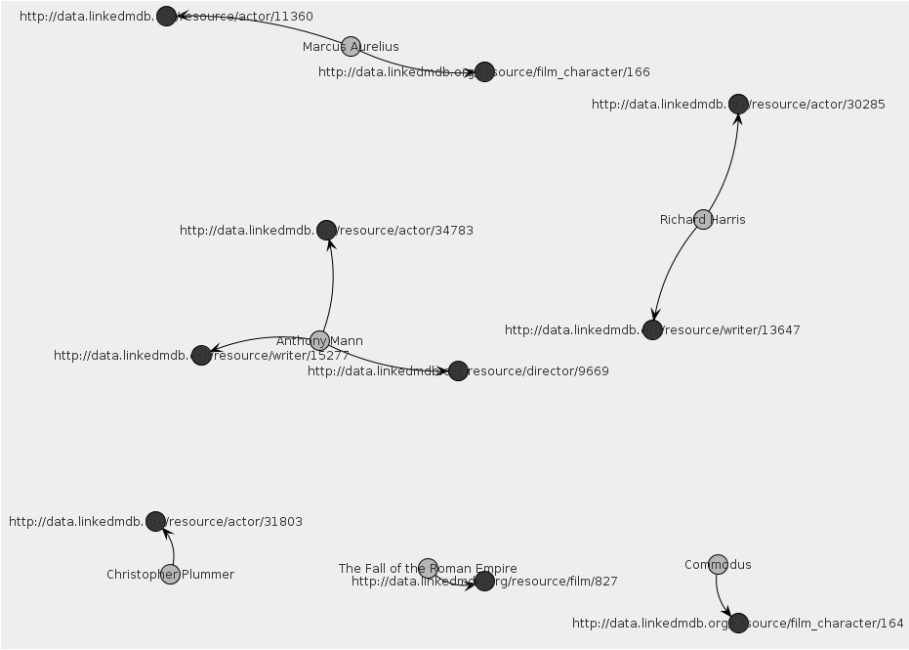


Figura 8.6: Grafo de Referência (menções em cinza claro e entidades candidatas em cinza escuro).

8.3.3 Cálculo de Similaridade Semântica Entre Entidades Candidatas

Além das arestas que ligam as menções às entidades candidatas, temos também as arestas que indicam a existência de relacionamentos de contexto entre as diversas

entidades candidatas. Como visto na Seção de Análise do Contexto das Entidades (Seção 8.1), duas entidades estão de alguma forma relacionadas semanticamente se existe um relacionamento direto entre ambas ou se compartilham pelo menos um vizinho.

Para cada par de vértices v $\in$ V_{Ec} em G com algum grau de similaridade semântica, criamos uma aresta (a_i, a_j) $\in$ X ligando ambas as entidades, o que irá gerar uma aresta no sentido oposto pertencente a Y.

O processo de verificação de similaridade entre duas entidades candidatas quaisquer segue o algoritmo 5.

Algoritmo 5 Verificação de Similaridade entre as Entidades Candidatas a anotar um texto

```
for v1 ∈ V_Ec do
    for v2 ∈ V_Ec e v2 ≠ v1 do
        if SR(v1,v2) > 0 then
            crie aresta a_CC de v1 para v2
        end if
    end for
end for
```

O cálculo da similaridade semântica entre entidades SR(v_1,v_2) (MILNE; WITTEN, 2008) pode ser dividido em três passos. Primeiro calculamos a distância semântica entre as duas entidades, normalizamos para um valor entre 0 e 1 e posteriormente subtraimos este valor de 1 para obtermos a similaridade SR. A fórmula (8.2) representa este cálculo, onde v_1 e v_2 são as duas entidades candidatas, V1 e V2 são os conjuntos de todas as entidades relacionadas a v_1 e v_2, respectivamente, e L é o conjunto com todos os recursos existentes na base de dados em Linked Data.

$$SR(v_1, v_2) = 1 - \frac{log(max(|V1|, |V2|)) - log(|V1 \cap V2|)}{log(|L|) - log(min(|V1|, |V2|))} \tag{8.2}$$

Todo valor de SR(v_1,v_2) maior que zero dará origem a uma aresta a_{CC} que liga v_1 a v_2. Se aplicarmos o algoritmo sobre o Grafo de Referência Inicial de nosso exemplo, sem qualquer tipo de adaptação à base de Linked Data, damos origem ao grafo apresentado pela Figura 8.7.

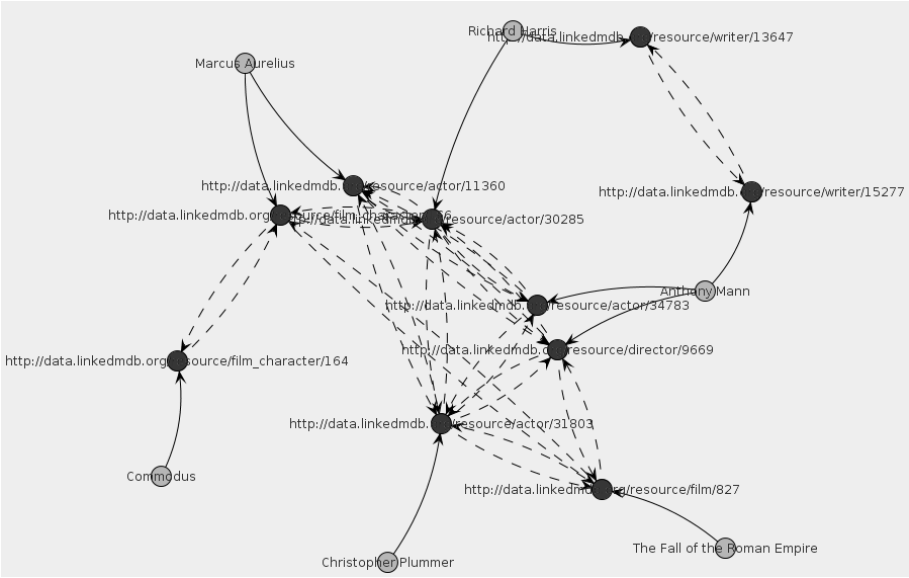


Figura 8.7: Grafo de Referência sem Adaptação (menções em cinza claro e entidades candidatas em cinza escuro).

Neste ponto podemos então identificar uma característica própria a bases de dados em Linked Data: **Bases de Dados em Linked Data possuem um conjunto de recursos que interligam grande parte das entidades presentes na base**. Este tipo de recurso será doravante chamado de **recurso de contexto da base** por ser responsável por indicar o assunto central da base e não o contexto em que as entidades aparecem. Em geral todas as classes entram nesta categoria de recurso pois elas indicam o assunto tratado na base e relacionam semanticamente todas as suas instâncias.

Além das classes que se utilizam do relacionamento funcional dado pela propriedade *rdf:type*, existem ainda outros recursos de contexto da base que utilizam relaciona-

mentos de domínio. Um exemplo deste tipo de recurso é aquele que representa a versão 3.0 da licença Creative Commons (http://creativecommons.org/licenses/by-nc-sa/3.0/) e que está relacionado a mais de 11000 recursos descritos pela base de dados Jamendo. Não é de se estranhar que esta licença represente um recurso de contexto da base em uma base de dados como a Jamendo, já que ela é responsável por descrever trabalhos musicais disponíveis sob licenças abertas.

Os recursos de contexto da base não auxiliam na identificação do contexto em que uma dada entidade aparece, pois eles estão relacionados a todos ou pelo menos a grande parte dos recursos presentes na base, unindo entidades em contextos individuais diferentes. Por este motivo, optamos por eliminar estes recursos no processo do cálculo de similaridade semântica.

Outro fato importante é que, de acordo com o cálculo de similaridade semântica original, poderiam existir arestas entre entidades candidatas a uma mesma menção. Como acreditamos que este tipo de relação semântica não contribui para o processo de desambiguação, optamos por não efetuar o cálculo para pares de entidades ligadas a uma mesma menção.

De acordo com estas informações, excluímos do processo de cálculo de similaridade semântica:

- Todos os recursos referentes a classes relacionadas através da propriedade rdf:type.
- Todos os recursos relacionados a mais de um determinado número de entidades (recursos de contexto da base).
- Todos os cálculos de similaridade entre entidades candidatas a uma mesma menção.

Aplicando o algoritmo ao texto de exemplo com estas novas restrições chegamos finalmente ao grafo apresentado pela Figura 8.8.

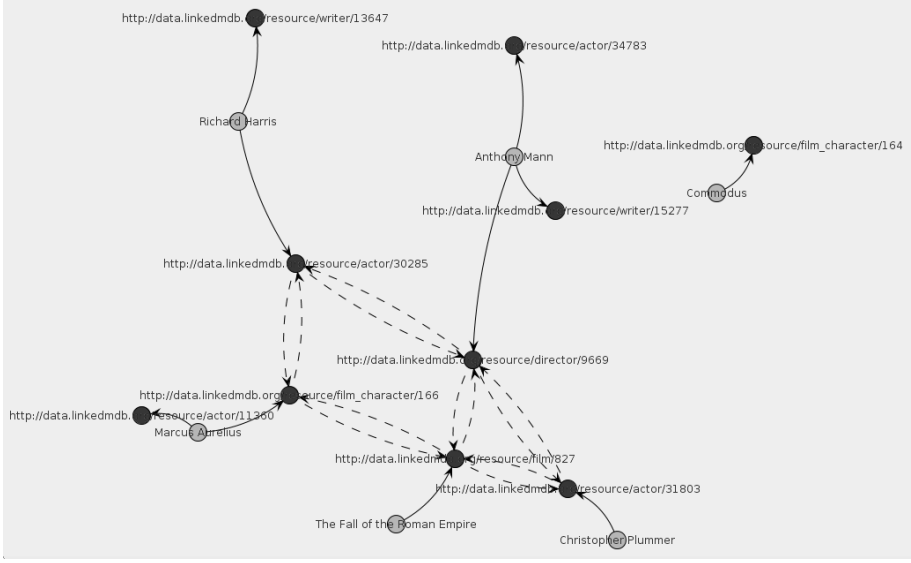


Figura 8.8: Grafo de Referência Completo (menções em cinza claro e entidades candidatas em cinza escuro).

8.3.4 Cálculo do Grau de Importância das Menções

A quarta etapa de nossa solução consiste em calcular o grau de importância de uma menção no texto a fim de identificar qual das menções terá maior peso no processo de desambiguação.

Uma determinada menção é importante para a indicação do contexto do texto quanto mais citada ela é. Se uma menção aparece muitas vezes então seu grau de importância deve ser maximizado, caso contrário sua importância tem de ser minimizada.

O cálculo de importância utilizado originalmente é o tf-idf, ou seja, uma medida que indica o quão importante é um termo por sua alta frequência no texto e baixa

frequência em outros textos do mesmo conjunto. Porém, esta medida só pode ser utilizada quando há um conjunto de textos a ser considerado. Como nosso objetivo é anotar um único texto, a importância de uma menção será dada por sua frequência no texto (Fórmula (8.3)).

$$Importancia(m) = \frac{|m|}{|M|} \tag{8.3}$$

No cálculo da Importância, $|m|$ representa o número de vezes em que uma mesma menção **m** aparece no texto e $|M|$ representa o valor total de menções existentes no texto.

8.3.5 Cálculo das Probabilidades de Propagação da Importância

Na etapa anterior calculamos a importância de uma menção. Nesta etapa iremos calcular a probabilidade de propagação desta importância para cada uma das entidades candidatas. Os valores que indicam a probabilidade de propagação serão inseridos no Grafo de Referência como pesos relacionados às arestas.

Para as arestas a_{MC} utilizamos o próprio valor CP como probabilidade de propagação de evidência (Fórmula (8.4)). Por ser um valor já normalizado não necessitamos de nenhum proceso de normalização.

$$P(a_{MC}) = P(v_m \rightarrow v_{ec}) = CP(m, c) \tag{8.4}$$

No caso das arestas a_{CC} a probabilidade de que a importância se propague de uma entidade para as demais é dado pelo valor de similaridade semântica SR entre as

entidades. Sendo assim, a probabilidade de propagação entre duas entidades é a normalização do valor de SR levando em conta todas as arestas que partem da mesma fonte. O cálculo é dado pela Fórmula (8.5) onde N_c é o conjunto de vizinhos de **c** quando a_{CC} = (c,x).

$$P(a_{CC}) = P(v_{ec_i} \rightarrow v_{ec_j}) = \frac{SR(c_i, c_j)}{\sum_{c \in N_{c_i}} SR(c_i, c)} \tag{8.5}$$

8.3.6 Inferência Coletiva

Com o Grafo de Referência completo, os valores iniciais de importância e as probabilidades de propagação da importância, nosso trabalho se resume a encontrar a entidade candidata, para cada menção **m**, que:

- Possui a mais forte relação de compatibilidade com a menção **m**.
- Possui a mais forte relação semântica com todas as entidades que representam as outras menções.

A primeira relação é diretamente refletida pela medida CP(m,c) que representa a compatibilidade entre a entidade candidata e a menção. A segunda relação é um pouco mais complicada pois não queremos a relação SR com apenas uma das entidades, mas a relação com todas as outras entidades candidatas, sobretudo as entidades que realmente irão anotar as menções. Esta relação será dada pela função $r_d(c)$ que pode ser identificada como o valor de evidência de que uma entidade candidata **c** pode anotar o texto do documento **d**.

O nosso problema passa a ser a identificação de qual a entidade candidata **c** que maximiza o valor da função dada pela Fórmula (8.6).

$$m.C = \underset{c}{argmax}\ CP(m,c) \times r_d(c) \tag{8.6}$$

Neste último passo do método iremos inferir coletivamente quais são todas as entidades **c** a anotar o texto. Para isso, iremos utilizar a definição dada por Han, Sun e Zhao (2011):

> **Definição 8.5** *Dado um Grafo de Referência G=(V,E) contendo n vértices, atribuimos um índice inteiro a cada vértice de 1 a $|V|$ e utilizamos este índice para representar o vértice. Desta forma podemos escrever a matriz de adjacências A do Grafo de Referência G onde A[i,j] ou A_{ij} é o peso da aresta que liga o vértice i ao vértice j.*
>
> ***s**: Vetor com as importâncias iniciais. Um vetor $n \times 1$ onde s_i = Importancia(m) se i corresponde a uma menção m.*
>
> ***r**: Vetor de evidências. Um vetor $n \times 1$ onde r^i é o valor de evidência de que o vértice i pode ser uma entidade presente no documento d (ou seja, $r_d(c)$ se o vértice i corresponde à entidade c) ou o valor de importância do vértice i (se o vértice i corresponde a uma menção).*
>
> ***T**: A matriz de propagação. Uma matriz $n \times n$ onde T_{ij} é a probabilidade de propagação da importância/evidência do vértice j para o vértice i.*

O vetor de importâncias iniciais **s** representa a importância inicial de todos os vértices, sendo sempre um número positivo para cada vértice representando uma menção no texto e 0 para cada vértice representando entidades candidatas.

O vetor de evidências **r** representa o valor de evidência que queremos efetivamente calcular. Cada vértice que representa um menção terá sempre o valor de evidência dado pelo valor de sua importância no texto, enquanto a evidência de um vértice representando uma entidade candidata é dado pela sua relação com a menção que pretende anotar e com a similaridade semântica com outras entidades candidatas. Nosso objetivo é, então, computar o valor do vetor de evidências **r**. Sendo assim, estabelecemos que seu valor inicial $\mathbf{r}^0$ é o valor do vetor de importâncias iniciais **s**:

$$r^0 = s \tag{8.7}$$

O passo seguinte é computar o vetor de evidências r através da propagação das evidências utilizando a matriz **T**. Esta matriz representa todas as relações existentes entre os diferentes vértices, seja a relação entre as menções e as suas respectivas entidades candidatas, seja a relação de similaridade existente entre as diferentes entidades candidatas. A forma recursiva para o cálculo do vetor **r** fica da seguinte forma:

$$r^{t+1} = T \times r^t \tag{8.8}$$

onde $\mathbf{r}^t$ é o vetor de evidências em um tempo t já conhecido.

Alguns vértices não possuem aresta de saída que faça com que sua evidência se propague, como é o caso das entidades candidatas que não se relacionam a qualquer outra entidade candidata. Neste caso faremos a inserção de um fator de realocação (λ). A cada iteração realocamos uma fração da evidência dada pelo vetor **s**. Com este passo impedimos que, em qualquer momento, todas as evidências fiquem zeradas, o que não contribuiria em nada para nosso método. O valor de **r** então passa a

ser dado por:

$$r^{t+1} = (1 - \lambda) \times T \times r^t + \lambda \times s \tag{8.9}$$

No cálculo de **r** levamos em conta as relações semânticas existentes entre as entidades (**T**), a compatibilidade local entre menções e entidades (**T**) e a importância relativa das menções no texto (**s**).

Após uma série de iterações, as entidades candidatas escolhidas para anotarem as menções são aquelas que obtiverem o maior valor de $\mathbf{CP(m,c)} \times \mathbf{r}_d\mathbf{(c)}$, o que, no nosso caso, são as entidades candidatas que obtiverem o maior valor de evidência **r** dentre as entidades candidatas para cada menção **m**.

8.4 Avaliação

A avaliação do método utilizado para o componente de Desambiguação de Entidades Nomeadas foi feita com base nos mesmos textos utilizados para a avaliação do componente de Reconhecimento de Menções a Entidades Nomeadas.

O Gold Standard utilizado para esta avaliação contou com a anotação (identificação das menções e escolha das entidades a anotar cada menção) manual dos 66 textos de biografia do sítio web Jamendo e dos 28 textos de trívia do sítio web IMDB. Os textos de biografia foram anotados com as entidades presentes na base de dados Jamendo e os textos de trívia foram anotados com a Linked Movie Database.

Todos os textos foram lidos e as menções foram identificadas manualmente. Após a identificação houve uma busca por cada menção nas respectivas bases de dados

em Linked Data. Apenas as entidades cujo nome foi exatamente o descrito no texto foram anotadas, ignorando maiúsculas e minúsculas. Quando apenas o primeiro nome de uma pessoa ou parte de um nome era citado sem que houvesse correlação completa com o nome na base de dados não houve anotação.

Tivemos de definir alguns parâmetros para a realização da desambiguação:

1. Algoritmo a ser utilizado para o processo de Reconhecimento de Menções a Entidades Nomeadas.
2. Lista com os recursos de contexto da base que não serão considerados no processo de cálculo da similaridade semântica entre entidades.
3. Escolha do fator de realocação λ.

Para o Reconhecimento de Menções a Entidades Nomeadas utilizamos o método construído a partir do Lucene Gazetteer limitado a propriedades-nome reconhecendo apenas nomes próprios, pois foi o método que obteve o menor tempo de processamento e a maior porcentagem de resultados corretos segundo nossa avaliação (Seção 6.3).

Para cada base de dados fizemos uma consulta SPARQL (Quadro 8.2) para saber o número de relacionamentos em que cada recurso faz parte. Pudemos notar que as bases são bem distintas. A Jamendo demonstrou ter recursos participantes dentre 1 a mais de 102000 relacionamentos enquanto os da Linked Movie Database participam dentre 1 a mais de 197000. A primeira base se caracteriza por entidades nomeadas relacionadas a um número pequeno de entidades (de 4 a 60) enquanto na segunda base as entidades nomeadas tendem a ter de dezenas a centenas de relacionamentos. Sendo assim, a partir de uma análise manual das bases de dados,

escolhemos os número 75 e 1100 como padrão de corte para o número de relacionamentos máximo que uma entidade pode ter nas bases Jamendo e Linked Movie Database, respectivamente, para ser considerada no cálculo de similaridade semântica.

Quadro 8.2: Consulta SPARQL para verificação do número de relacionamentos de cada recurso

```
SELECT?s COUNT(DISTINCT(?o)) AS ?num
WHERE {
    { ?s ?p ?o.
        FILTER(isIri(?o)).
    } UNION {
        ?o ?p ?s.
        FILTER(isIri(?s)).
    }
}ORDER BY DESC(?num)
```

Por fim, optamos por utilizar nesta avaliação o mesmo fator de realocação λ de 0,1 utilizado por Han, Sun e Zhao (2011).

Ao realizarmos a desambiguação, verificamos um resultado satisfatório mas que foi fortemente influenciado pelo Reconhecimento de Menções a Entidades Nomeadas. Isto ocorre pelo fato de que uma menção erroneamente reconhecida não poderá ser desambiguada pois ela não possui nenhuma entidade realmente relacionada a ela na base de dados. Grande parte dos erros do processo de desambiguação foram originados pelo algoritmo de reconhecimento de menções que reconheceu os textos corretamente mas eram menções que não estavam efetivamente relacionadas à uma das entidades da base de dados.

Outro ponto importante é que o tempo de processamento aumenta como o número de entidades candidatas identificadas para o texto, pois o cálculo de similaridade

semântica requer a realização de consultas SPARQL para cada par de entidades. Devido ao grande número de consultas, o tempo de processamento de um único texto foi alto levando quase 4 minutos, em média, para desambiguar um texto de trívia de cerca de 4 linhas.

Quadro 8.3: Resultado da Avaliação da Desambiguação com aplicação do algoritmo de Reconhecimento de Menções a Entidades Nomeadas

Base de Dados	Precisão		Recall		F-Measure (Total)	Tempo Médio
	Média	Desvio Padrão	Média	Desvio Padrão		
Jamendo	0,4228	0,3177	0,7480	0,3613	0,3795	60702 ms
Linked MDB	0,7652	0,2268	0,9937	0,0327	0,8179	223418 ms

8.5 Trabalhos Relacionados

Existem os mais diversos trabalhos na área de Desambiguação de Entidades Nomeadas. Portanto, daremos foco apenas aos métodos que utilizam bases de conhecimento para o processo de desambiguação, a fim de compará-los com nossa abordagem. Dentre estes métodos podemos definir dois tipos principais: o método local e o global.

O método local consiste em desambiguar uma menção por vez. Grande parte dos métodos de desambiguação local foram inspirados na definição de Saussure (apud BIEMANN, 2012) que diz que os sinais de linguagem (palavras) são unicamente determinados por seus relacionamentos com outros sinais. Ou seja, o significado de uma palavra em um texto é dado pela existência das outras palavras em seu entorno e, além disso, duas palavras com contextos semelhantes tendem a expressar significados semelhantes. Neste campo, podemos citar os trabalhos de Bunescu e Pasca (2006), Mihalcea e Csomai (2007), Cucerzan (2007), Fader et al. (2009) e Mendes et al. (2011) que utilizam a semelhança entre o texto no entorno das menções e o texto de descrição de entidades. Cucerzan (2007) utiliza ainda as categorias da Wikipédia

para melhorar o processo de desambiguação e Mendes et al. (2011) trabalham com Linked Data.

Ainda dentre os métodos de desambiguação local existem aqueles que se utilizam de diversas características das menções e das entidades para realizar a desambiguação utilizando aprendizado de máquina. Dentre estes trabalhos podemos citar Dredze et al. (2010), Zhang et al. (2010) e Zhou et al. (2010).

Ao contrário do método local, o método de desambiguação global, adotado por nós, resolve todas as menções ao mesmo tempo. A ideia por trás da desambiguação é o fato de que todas as entidades citadas no texto estão inseridas em um mesmo contexto, então as entidades candidatas que devem anotar o texto devem estar também em um mesmo contexto dentro da base de conhecimento utilizada. É comum que trabalhos com desambiguação global utilizem também as métricas levantadas pelos trabalhos de desambiguação local no que diz respeito ao entorno do texto da menção que se quer desambiguar.

Dentre os trabalhos que utilizam o método global e a Wikipédia como base de conhecimento podemos citar Gentile et al. (2009), Kulkarni et al. (2009), Hachey, Radford e Curran (2011) e Han, Sun e Zhao (2011). Gentile et al. (2009) e Han, Sun e Zhao (2011) utilizam diferentes funções representando a relação entre a menção e as entidades candidatas, Kulkarni et al. (2009) estudam a formação de cliques na base de conhecimento entre as entidades candidatas, enquanto Hachey, Radford e Curran (2011) estudam a viabilidade no uso de duas medidas de grafo para o processo de desambiguação.

Utilizando bases de dados em Linked Data verificamos os trabalhos de Hoffart et al. (2011) e Hakimov, Oto e Dogdu (2012). Hoffart et al. (2011) utilizam a popularidade das entidades na base de dados, a similaridade entre as menções e o nome das

entidades e a distância semântica entre entidades para o processo de desambiguação utilizando a base YAGO. Por fim Hakimov, Oto e Dogdu (2012) utilizam medidas de centralidade no grafo formado pelas menções existentes no texto.

Apenas para fim de comparação, Rizzo e Troncy (2011) efetuam a avaliação de cinco ferramentas utilizando textos de notícias da BBC e do New York Times onde quatro delas trabalham com Linked Data. Nesta avaliação, a AlchemyAPI (ALCHEMYAPI..., 2012) apresentou 0,4013 e 0,2069 de precisão, a DBPedia Spotligh apresentou 0,7677 e 0,0635, a OpenCalais[1] ficou com 0 em ambas as avaliações e a Zemanta[2] apresentou 0,8938 e 0,8950.

Existem ainda os trabalhos de Hachey et al. (2012) e Rizzo e Troncy (2011) que avaliam o desempenho dos métodos e ferramentas que utilizam a Wikipédia e Linked Data, respectivamente.

[1] http://opencalais.com
[2] http://www.zemanta.com

9 CONCLUSÃO

Iniciamos este trabalho apresentando o cenário de evolução da informação até o advento da Web 3.0 e motivando a união das áreas de Processamento de Linguagem Natural e Web Semântica na representação de conhecimento em um formato processável por máquina.

Ainda no Capítulo 1 apresentamos o problema a ser resolvido, a hipótese que pretendemos validar, nossa proposta de solução e separamos os objetivos no nosso trabalho em objetivos gerais e específicos.

A seguir, no Capítulo 2, apresentamos os principais tópicos para compreensão do nosso trabalho: Entidades Nomeadas, Gazetteer e a abordagem de Linked Data e suas tecnologias.

Do Capítulo 3 em diante apresentamos nossa proposta de solução separada em diversos componentes a fim de alcançar cada um dos objetivos específicos propostos. Ao final de cada capítulo apresentamos os trabalhos relacionados a cada assunto tratado no capítulo.

No Capítulo 3 fizemos um levantamento da metodologia adotada pelos trabalhos em Resolução de Entidades Nomeadas com Linked Data e identificamos a metodologia que iríamos utilizar. Com isso, alcançamos nosso primeiro objetivo de *identificação da metodologia a ser utilizada.* Além disso, separamos nossa solução em uma arquitetura formada por 5 componentes sequenciais.

O Capítulo 4 tratou do componente de Seleção de Bases de Dados em Linked Data, o primeiro componente de nossa arquitetura. Levantamos um conjunto de critérios de qualidade para que bases de dados em Linked Data possam ser utilizadas na tarefa de Resolução de Entidades Nomeadas e avaliamos um conjunto de bases da Nuvem de LOD. Desta avaliação concluímos que podemos utilizar bases em Linked Data para a tarefa proposta, sobretudo com nossa abordagem. Porém, verificamos ainda que o nível de qualidade de algumas bases ainda está muito aquém do desejado. Fato este corroborado pela análise feita durante a criação do Gold Standard desenvolvido para o componente de Reconhecimento de Propriedades-Nome. Neste capítulo atingimos nosso segundo objetivo específico: a *identificação de um conjunto de critérios de qualidade para bases em Linked Data visando a tarefa de Resolução de Entidades Nomeadas.*

No Capítulo 5 motivamos a necessidade de um método para o reconhecimento de propriedades-nome. Apresentamos nosso método e um conjunto de heurísticas para o reconhecimento de propriedades que guardavam o nome de Entidades Nomeadas como seus valores. Avaliamos o uso das diversas heurísticas em bases de dados em Linked Data selecionadas pelo componente anterior e verificamos que todas as heurísticas podem ser utilizadas. Cada uma deve ser escolhida de acordo com as características da base de dados. O desenvolvimento deste componente permitiu alcançarmos nosso terceiro objetivo específico: a *descoberta das estruturas que identificam as Entidades Nomeadas e seus nomes em bases de dados em Linked Data.* Estas estruturas são as classes e propriedades porém diferem de uma base de dados

para outra. Por isso, devemos criar métodos específicos para seu reconhecimento dependendo da base de dados dada como entrada.

A seguir, tratamos do desenvolvimento do componente de Geração de Dicionário de Nomes. No Capítulo 6, criamos um método capaz de transformar uma base de dados em Linked Data em um dicionário de nomes, também conhecido como Gazetteer. Geramos diferentes Gazetteers utilizando tecnologias diferentes e parâmetros diferentes. Percebemos através de um processo de avaliação, que o uso exclusivo de propriedades-nome aumenta muito a precisão do Gazetteer e que ambas as tecnologias SPARQL e Lucene podem ser utilizadas para geração do mesmo. Notamos ainda que os métodos otimizados para o processo de busca textual demoram um tempo consideravelmente menor que os métodos de consulta comum. Com este componente concluímos mais um objetivo específico, a *geração de um dicionário de nomes a partir de uma base de dados em Linked Data com uso de metadados genéricos.*

No Capítulo seguinte, desenvolvemos o componente de Reconhecimento de Menções a Entidades Nomeadas. Neste componente apresentamos um método para o reconhecimento de menções no texto utilizando o Dicionário de Nomes desenvolvido no componente anterior. Fizemos uma avaliação com uma lista de 200 nomes e verificamos que nosso método reconhece todo nome que esteja no Dicionário. Apesar disso, o método ainda carece de algumas características interessantes como o reconhecimento de nomes independente de letras maiúsculas ou minúsculas ou o reconhecimento parcial de nomes onde o cantor "Michael Jackson" poderia ser reconhecido apenas por "Michael", por exemplo. Com este componente concluímos o objetivo de *identificação de menções a entidades nomeadas no texto utilizando o dicionário de nomes gerado.*

Por último, no Capítulo 8, apresentamos nosso componente de Desambiguação de

Entidades Nomeadas. O método criado para este componente foi uma adaptação do método desenvolvido por Han, Sun e Zhao (2011), que utilizava a Wikipédia, para bases de dados em Linked Data. Fizemos uma avaliação do método em um conjunto de 94 textos manualmente anotados e verificamos que o método apresenta resultados aceitáveis. Notamos que a identificação de menções influencia muito na desambiguação pois uma menção erroneamente identificada não pode ser desambiguada. Ao mesmo tempo, vimos que a qualidade dos dados influencia também no processo de desambiguação. Quanto mais informação sobre relacionamentos de domínio existentes na base de dados, melhor é o processo de desambiguação. Este componente conclui nosso último objetivo específico ao realizarmos a *desambiguação de entidades nomeadas utilizando bases de dados em Linked Data.*

Com este trabalho e o resultado de nossas avaliações pudemos concluir que a hipótese de que *é possível utilizar uma base de dados no formato de Linked Data para a tarefa de Resolução de Entidades Nomeadas sem que precisemos saber, a priori, o significado das classes e propriedades dado pelos descritores utilizados pela base* é verdadeira. Pois conseguimos construir cada componente de nossa arquitetura sem que precisássemos levar em conta o significado de cada classe e propriedade previamente. Apenas verificamos seus significados durante o processo de avaliação de nossos resultados, mas não para a execução de nossos métodos. Ao final, durante a avaliação do componente de Desambiguação de Entidades Nomeadas, provamos que é possível criar uma ferramenta que resolva o problema de Resolução de Entidades Nomeadas utilizando bases de dados em Linked Data, ainda que de baixa qualidade. Em nossos experimentos obtivemos uma precisão média na anotação de textos de 42% utilizando a Jamendo e de 76% utilizando a Linked Movie Database.

Ao permitirmos que um número maior de bases sejam utilizadas e ao não levarmos em conta os significados das classes e propriedades cumprimos nosso objetivo geral. Conseguimos *a automatização do processo de resolução de entidades nomeadas*

utilizando bases de dados em Linked Data com uso de metadados genéricos.

A conclusão final de nosso trabalho é que demos um primeiro passo para que um número maior de bases de dados sejam utilizadas na resolução da tarefa de Resolução de Entidades Nomeadas. Ao comprovarmos que é possível a criação de ferramentas que utilizem bases de dados em Linked Data sem levar em conta o tipo de metadado utilizado, o problema de *inexistência de ferramentas de resolução de entidades nomeadas utilizando bases de dados em Linked Data que permitam a escolha da base pelo usuário* não será mais um problema a partir do momento em que nossos métodos sejam propriamente aplicados.

9.1 Trabalhos Futuros

Com a realização deste trabalho, demos um grande passo na utilização de bases de dados em Linked Data como bases de conhecimento para a Resolução de Entidades Nomeadas. Apesar disto, ainda há muito trabalho a ser feito.

Identificamos uma lista de trabalhos futuros que resultam deste trabalho:

- Desenvolvimento de métodos que descubram qual a melhor base de dados na nuvem de LOD a anotar um determinado texto.
- Automatização do processo de avaliação de bases de dados.
- Criação de heurísticas mais sofisticadas que levem em conta características já conhecidas de certas classes de Entidades Nomeadas para o reconhecimento de propriedades-nome.
- Utilização de um conjunto de classes e propriedades que já são comuns na

identificação de Entidade Nomeadas.

- Utilização de métodos para atualização automática do Lucene Gazetteer quando houver mudanças na base original.
- Reconhecimento de menções a nomes parciais, como o primeiro nome de um artista ou apenas seu sobrenome, por exemplo.
- Desambiguação de entidades que não casam realmente com uma entidade da base de dados. Ou seja, o nome é igual mas a entidade verdadeira não está realmente presente na base de dados.
- Utilização de nossa abordagem com textos e bases de dados em outros idiomas.
- Avaliação de nossa arquitetura utilizando bases em Linked Data em outros domínios do conhecimento.
- Comparação do resultado de nossa abordagem com ferramentas que já utilizam bases em Linked Data.
- Estudo de que componentes de nossa arquitetura podem melhorar as ferramentas já existentes.

REFERÊNCIAS

ALCHEMYAPI – Named Entity Extraction. 2012. Disponível em: <http://www.alchemyapi.com/api/entity>. Acesso em: abr. de 2012.

ALIAS-I. Lingpipe 4.0.0. Disponível em: <http://alias-i.com/lingpipe>. Acesso em: agosto de 2012.

ANDERSON, P. What is web 2.0. *Ideas, technologies and implications for education*, Joint Information Systems Committee Technology and Standards, v. 60, 2007.

APACHE. Apache opennlp. 2010. Disponível em: <http://opennlp.apache.org/>. Acesso em: agosto de 2012.

AUER, S. et al. Dbpedia: A nucleus for a web of open data. *The Semantic Web*, Springer, p. 722–735, 2007.

AUER, S. et al. Triplify: light-weight linked data publication from relational databases. In: ACM. *Proceedings of the 18th international conference on World wide web*. [S.l.], 2009. p. 621–630.

BERNERS-LEE, T. Linked data-design issues. *W3C*, ACM Press, v. 2009, n. 09/20, 2006.

BIEMANN, C. *Structure Discovery of Natural Language*. [S.l.]: Springer, 2012.

BIZER, C.; CYGANIAK, R. D2r server-publishing relational databases on the semantic web. In: *5th International Semantic Web Conference*. [S.l.: s.n.], 2006. p. 26.

BIZER, C.; HEATH, T.; BERNERS-LEE, T. Linked data-the story so far. *International Journal on Semantic Web and Information Systems (IJSWIS)*, IGI Global, v. 5, n. 3, p. 1–22, 2009.

BIZER, C.; JENTZSCH, A.; CYGANIAK, R. State of the lod cloud. 2011. Disponível em: <http://www4.wiwiss.fu-berlin.de/lodcloud/state/>. Acesso em: nov. de 2012.

BORTHWICK, A. et al. Nyu: Description of the mene named entity system as used in muc-7. In: *Proceedings of the Seventh Message Understanding Conference (MUC-7)*. [S.l.: s.n.], 1998. v. 6.

BREITMAN, K. *Web semântica: a internet do futuro*. [S.l.]: LTC, 2005.

BRICKLEY, D.; GUHA, R. V. Rdf vocabulary description language 1.0: Rdf schema. w3c recommendation 10 february 2004. 2004. Disponível em: <http://www.w3.org/TR/rdf-schema/>. Acesso em: agosto de 2012.

BRICKLEY, D.; MILLER, L. Foaf vocabulary specification 0.98. namespace document 9 august 2010. 2010. Disponível em: <http://xmlns.com/foaf/spec/>. Acesso em: agosto de 2012.

BUNESCU, R.; PASCA, M. Using encyclopedic knowledge for named entity disambiguation. In: *Proceedings of EACL*. [S.l.: s.n.], 2006. v. 6, p. 9–16.

CHEN, H. et al. Description of the ntu system used for met2. In: *Proceedings of 7th Message Understanding Conference*. [S.l.: s.n.], 1998.

CIMIANO, P. *Ontology learning and population from text: algorithms, evaluation and applications*. [S.l.]: Springer-Verlag New York Inc, 2006.

COWIE, J. Crl/nmsu: description of the crl/nmsu systems used for muc-6. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 6th conference on Message understanding*. [S.l.], 1995. p. 157–166.

CUCERZAN, S. Large-scale named entity disambiguation based on wikipedia data. In: *Proceedings of EMNLP-CoNLL*. [S.l.: s.n.], 2007. v. 2007, p. 708–716.

DCMI. Dublin core metadata element set, version 1.1. 2012. Disponível em: <http://dublincore.org/documents/dces/>. Acesso em: agosto de 2012.

DREDZE, M. et al. Entity disambiguation for knowledge base population. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 23rd International Conference on Computational Linguistics*. [S.l.], 2010. p. 277–285.

FADER, A. et al. Scaling wikipedia-based named entity disambiguation to arbitrary web text. In: *IJCAI-09 Workshop on User-contributed Knowledge and Artificial Intelligence (WikiAI09)*. [S.l.: s.n.], 2009.

FERRIS, B. Quality indicators for linked data datasets. 2010. Disponível em: <http://answers.semanticweb.com/questions/1072/quality-indicators-for-linked-data-datasets>. Acesso em: agosto de 2012.

FLEMMING, A.; HARTIG, O. Quality criteria for linked data sources. 2010. Disponível em: <goo.gl/deKrD>. Acesso em: agosto de 2012.

FüRBER, C.; HEPP, M. Towards a vocabulary for data quality management in semantic web architectures. In: ACM. *Proceedings of the 1st International Workshop on Linked Web Data Management*. [S.l.], 2011. p. 1–8.

GENTILE, A. et al. Graph-based semantic relatedness for named entity disambiguation. Demetra EOOD, 2009.

GOEL, K.; GUPTA, P. Introducing schema.org: Search engines come together for a richer web. 2011. Disponível em:

<http://googlewebmastercentral.blogspot.com.br/2011/06/introducing-schemaorg-search-engines.html>. Acesso em: abril de 2012.

GRISHMAN, R. Named entity task definition: Introduction. 1995. Disponível em: <http://cs.nyu.edu/cs/faculty/grishman/NEtask20.book_2.html>. Acesso em: jul. de 2012.

GRISHMAN, R.; SUNDHEIM, B. Message understanding conference-6: A brief history. In: *Proceedings of COLING*. [S.l.: s.n.], 1996. v. 96, p. 466–471.

HACHEY, B.; RADFORD, W.; CURRAN, J. Graph-based named entity linking with wikipedia. *Web Information System Engineering–WISE 2011*, Springer, p. 213–226, 2011.

HACHEY, B. et al. Evaluating entity linking with wikipedia. *Artificial intelligence*, Elsevier, 2012.

HAKIMOV, S.; OTO, S.; DOGDU, E. Named entity recognition and disambiguation using linked data and graph-based centrality scoring. In: ACM. *Proceedings of the 4th International Workshop on Semantic Web Information Management*. [S.l.], 2012. p. 4.

HAN, X.; SUN, L.; ZHAO, J. Collective entity linking in web text: a graph-based method. In: ACM. *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. [S.l.], 2011. p. 765–774.

HASSANZADEH, O.; CONSENS, M. Linked movie data base. In: *Proceedings of the 2nd Workshop on Linked Data on the Web (LDOW2009)*. [S.l.: s.n.], 2009.

HAYES, P. Rdf semantics. w3c recommendation 10 february 2004. 2004. Disponível em: <http://www.w3.org/TR/2004/REC-rdf-mt-20040210/>. Acesso em: agosto de 2012.

HILL, L.; FREW, J.; ZHENG, Q. Geographic names: The implementation of a gazetteer in a georeferenced digital library. Corporation for National Research Initiatives, 1999.

HOFFART, J. et al. Robust disambiguation of named entities in text. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. [S.l.], 2011. p. 782–792.

HOGAN, A. et al. An empirical survey of linked data conformance. *Web Semantics: Science, Services and Agents on the World Wide Web*, Elsevier, 2012.

IVAN. Resource description framework (rdf). 2004. Disponível em: <http://www.w3.org/2001/sw/wiki/RDF>. Acesso em: setembro de 2012.

KARKALETSIS, V. et al. Named-entity recognition from greek and english texts. *Journal of Intelligent & Robotic Systems*, Springer, v. 26, n. 2, p. 123–135, 1999.

KHALILI, A.; AUER, S. The rdfa content editor–from wysiwyg to wysiwym. In: *EEE Signature Conference on Computers, Software , and Aplications (COMPSAC 2012)*. [S.l.: s.n.], 2012.

KONCHADY, M. *Text Mining Application Programming*. [S.l.]: Charles River Media, 2006.

KRUPKA, G. Sra: Description of the sra system as used for muc-6. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 6th conference on Message understanding*. [S.l.], 1995. p. 221–235.

KULKARNI, S. et al. Collective annotation of wikipedia entities in web text. In: ACM. *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. [S.l.], 2009. p. 457–466.

MACBETH, S. Introducing schema.org: Bing, google and yahoo unite to build the web of objects. 2011. Disponível em: <http://www.bing.com/community/site_blogs/b/search/archive/2011/06/02/bing-google-and-yahoo-unite-to-build-the-web-of-objects.aspx>. Acesso em: abril de 2012.

MARTIN, P. Organizing linked data quality related methods. In: *2012 International Conference on Information and Knowledge Engineering (IKE12)*. [S.l.: s.n.], 2012.

MCNAMEE, P.; DANG, H. Overview of the tac 2009 knowledge base population track. In: *Text Analysis Conference (TAC)*. [S.l.: s.n.], 2009.

MENDES, P.; JAKOB, M.; BIZER, C. Dbpedia: A multilingual cross-domain knowledge base. In: CHAIR), N. C. C. et al. (Ed.). *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*. Istanbul, Turkey: European Language Resources Association (ELRA), 2012. ISBN 978-2-9517408-7-7.

MENDES, P. et al. DBpedia spotlight: shedding light on the web of documents. In: ACM. *Proceedings of the 7th International Conference on Semantic Systems*. [S.l.], 2011. p. 1–8.

MIHALCEA, R.; CSOMAI, A. Wikify!: linking documents to encyclopedic knowledge. In: ACM. *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. [S.l.], 2007. p. 233–242.

MIKHEEV, A.; MOENS, M.; GROVER, C. Named entity recognition without gazetteers. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*. [S.l.], 1999. p. 1–8.

MILES, A.; BECHHOFER, S. Skos simple knowledge organization system reference. w3c recommendation 18 august 2009. 2009. Disponível em: <http://www.w3.org/TR/skos-reference/>. Acesso em: agosto de 2012.

MILLER, L. Description of the saic dx syste m as used for muc-6. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 6th conference on Message understanding*. [S.l.], 1995. p. 193–206.

MILNE, D.; WITTEN, I. Learning to link with wikipedia. In: ACM. *Proceedings of the 17th ACM conference on Information and knowledge management*. [S.l.], 2008. p. 509–518.

NADEAU, D.; SEKINE, S. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, John Benjamins Publishing Company, v. 30, n. 1, p. 3–26, 2007.

PATEL-SCHNEIDER, P. F.; HAYES, P.; HORROCKS, I. Owl web ontology language semantics and abstract syntax. w3c recommendation 10 february 2004. 2004. Disponível em: <http://www.w3.org/TR/owl-semantics/>. Acesso em: agosto de 2012.

PEREIRA, B.; SILVA, J. da; VIVACQUA, A. Discovering names in linked data datasets. In: *Proceedings of the Workshop of Linked Entities (WOLE12)*. [S.l.: s.n.], 2012.

PILZ, A.; MOLZBERGER, L.; PAASS, G. Entity resolution by kernel methods. *Proc. Sabre TMS*, 2009.

PRUD'HOMMEAUX, E.; SEABORNE, A. Sparql query language for rdf. w3c recommendation 15 january 2008. 2008. Disponível em: <http://www.w3.org/TR/rdf-sparql-query/>. Acesso em: setembro de 2012.

RAIMOND, Y. et al. The music ontology. In: *Proceedings of the International Conference on Music Information Retrieval*. [S.l.: s.n.], 2007. p. 417–422.

RAO, D.; MCNAMEE, P.; DREDZE, M. Entity linking: Finding extracted entities in a knowledge base. p. 93–115, 2011.

RIZZO, G.; TRONCY, R. Nerd: evaluating named entity recognition tools in the web of data. In: *Workshop on Web Scale Knowledge Extraction (WEKEX'11), Bonn, Germany*. [S.l.: s.n.], 2011. p. 1–16.

RIZZO, G.; TRONCY, R. Nerd: A framework for unifying named entity recognition and disambiguation extraction tools. *EACL 2012*, p. 73, 2012.

SAUSSURE, F. de. *Course in General Linguistics*. [S.l.]: New York: McGraw-Hill, 1966.

SETH, S. Introducing schema.org: A collaboration on structured data. 2011. Disponível em: <http://www.ysearchblog.com/2011/06/02/introducing-schema-org-a-collaboration-on-structured-data/>. Acesso em: abr. de 2012.

SPORLEDER, C. et al. Identifying named entities in text databases from the natural history domain. In: *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC-06)*. [S.l.: s.n.], 2006. p. 1742–1745.

STUDER, R.; BENJAMINS, V.; FENSEL, D. Knowledge engineering: principles and methods. *Data & knowledge engineering*, Elsevier, v. 25, n. 1, p. 161–197, 1998.

SUCHANEK, F.; KASNECI, G.; WEIKUM, G. Yago: a core of semantic knowledge. In: ACM. *Proceedings of the 16th international conference on World Wide Web*. [S.l.], 2007. p. 697–706.

TURING, A. Computing machinery and intelligence. *Mind*, JSTOR, v. 59, n. 236, p. 433–460, 1950.

W3C. Ontologies - w3c. 2012. Disponível em: <http://www.w3.org/standards/semanticweb/ontology>. Acesso em: setembro de 2012.

WACHOLDER, N.; RAVIN, Y.; CHOI, M. Disambiguation of proper names in text. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the fifth conference on Applied natural language processing*. [S.l.], 1997. p. 202–208.

WIKIPEDIA. Orange - wikipedia, the free enclyclopedia. 2012. Disponível em: <http://en.wikipedia.org/wiki/Orange>. Acesso em: jul. de 2012.

WITTEN, I.; MILNE, D. An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In: *Proceeding of AAAI Workshop on Wikipedia and Artificial Intelligence: an Evolving Synergy, AAAI Press, Chicago, USA*. [S.l.: s.n.], 2008. p. 25–30.

YOSEF, M. et al. Aida: An online tool for accurate disambiguation of named entities in text and tables. *Proceedings of the VLDB Endowment*, v. 4, n. 12, 2011.

YU, S.; BAI, S.; WU, P. Description of the kent ridge digital labs system used for muc-7. In: *Proceedings of the Seventh Message Understanding Conference*. [S.l.: s.n.], 1998.

ZHANG, W. et al. Entity linking leveraging: automatically generated annotation. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 23rd International Conference on Computational Linguistics*. [S.l.], 2010. p. 1290–1298.

ZHOU, Y. et al. Resolving surface forms to wikipedia topics. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 23rd International Conference on Computational Linguistics*. [S.l.], 2010. p. 1335–1343.

Apêndice A

Quadro A.1: Classes e Propriedades utilizadas na descrição dos dados da Linked Movie Database. (dc = http://purl.org/dc/ terms/, dbpedia = http://dbpedia.org/property/, foaf = http://xmlns.com/foaf/0.1/, movie = http://data.linkedmdb.org/resource/movie/, oddlinked = http://data.linkedmdb.org/resource/oddlinker/ , rdf = http://www.w3.org/1999/02/22-rdf-syntax-ns#, rdfs = http://www.w3.org/2000/01/rdf-schema#, skos = http://www.w3.org/2004/02/skos/core# e owl = http://www.w3.org/2002/07/owl#)

Classe	Identifica EN	Propriedade	Relacionamento de Domínio
movie:actor	X	movie:actor_actorid	
		movie:actor_name	
		movie:actor_netflix_id	
		movie:actor_nytimes_id	
		movie:performance	X
		rdf:type	
		rdfs:label	
		owl:sameAs	
		foaf:page	
movie:cinematographer	X	movie:cinematographer_cinematographerid	
		movie:cinematographer_name	
		rdf:type	
		rdfs:label	

movie:cinematographer		foaf:page	
movie:content_rating		movie:content_rating_content_ratingid	
		movie:content_rating_country	
		movie:content_rating_film_rating_system	
		movie:content_rating_minimum_unaccompanied_age	
		movie:content_rating_name	
		movie:content_rating_system	X
		rdf:type	
		rdfs:label	
		foaf:page	
movie:content_rating_system		movie:content_rating_system_content_rating_systemid	
		movie:content_rating_system_jurisdiction	
		movie:content_rating_system_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:country	X	movie:country_areaInSqKm	
		movie:country_capital	
		movie:country_continent	
		movie:country_currency	
		movie:country_fips_code	
		movie:country_id	
		movie:country_iso_alpha2	
		movie:country_iso_alpha3	
		movie:country_iso_numeric	
		movie:country_languages	
		movie:country_name	
		movie:country_population	
		rdf:type	

movie:country		rdfs:label	
		owl:sameAs	
movie:director	X	movie:director_directorid	
		movie:director_name	
		rdf:type	
		rdfs:label	
		foaf:made	X
		foaf:page	
movie:dubbing_performance		movie:dubbing_performance_actor	
		movie:dubbing_performance_character	
		movie:dubbing_performance_dubbing_performanceid	
		movie:dubbing_performance_film	
		movie:dubbing_performance_language	
		movie:dubbing_performance_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:editor	X	movie:editor_editorid	
		movie:editor_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film	X	movie:actor	X
		movie:cinematographer	X
		movie:costume_designer	X
		movie:country	X
		movie:director	X
		movie:dubbing_performance	X
		movie:editor	X
		movie:executive_producer	X
		movie:featured_film_location	X
		movie:film_art_director	X

movie:film		movie:film_casting_director	X
		movie:film_collection	X
		movie:film_cut	X
		movie:film_featured_song	X
		movie:film_film_company_relationship	X
		movie:film_format	X
		movie:film_production_designer	X
		movie:film_regional_release_date	X
		movie:film_series	X
		movie:film_set_designer	X
		movie:film_story_contributor	X
		movie:film_subject	X
		movie:filmid	
		movie:genre	X
		movie:initial_release_date	
		movie:language	
		movie:location	
		movie:music_contributor	X
		movie:performance	X
		movie:personal_film_appearance	X
		movie:prequel	X
		movie:producer	X
		movie:production_company	X
		movie:rating	X
		movie:relatedBook	
		movie:runtime	
		movie:sequel	X
		movie:story_contributor	X
		movie:writer	X
		dbpedia:hasPhotoCollection	
		dc:date	
		dc:title	
		rdf:type	
		rdfs:label	

movie:film		owl:sameAs	
		skos:subject	X
		foaf:based_near	
		foaf:page	
movie:film_art_director	X	movie:film_art_director_id	
		movie:film_art_director_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_awards_ceremony	X	movie:film_awards_ceremony_film_awards_ceremony id	
		movie:film_awards_ceremony_id	
		movie:film_awards_ceremony_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_casting_director	X	movie:film_casting_director_film_casting_director_id	
		movie:film_casting_director_freebase_url	
		movie:film_casting_director_name	
		rdf:type	
		rdfs:label	
movie:film_character	X	movie:film_character_film_characterid	
		movie:film_character_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_collection	X	movie:film_collection_film_collectionid	
		movie:film_collection_name	
		rdf:type	
		rdfs:label	

movie:film_collection		foaf:page	
movie:film_company	X	movie:film_company_film_companyid	
		movie:film_company_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_costume_designer	X	movie:film_costume_designer_film_costume_designerid	
		movie:film_costume_designer_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_crew_gig	X	movie:film_crew_gig_film	X
		movie:film_crew_gig_film_crew_gigid	
		movie:film_crew_gig_film_job	X
		movie:film_crew_gig_id	
		movie:film_crew_gig_name	
		movie:film_crew_role	
		movie:film_crewmember	X
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_crewmember	X	movie:film_crewmember_film_crewmemberid	
		movie:film_crewmember_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_critic	X	movie:film_critic_film_criticid	
		movie:film_critic_name	

movie:film_critic		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_cut		movie:film_cut_film_cutid	
		movie:film_cut_note	
		movie:film_release_region	
		movie:runtime	
		movie:type_of_film_cut	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_distribution_medium		movie:film_distribution_medium_film_distribution_mediumid	
		movie:film_distribution_medium_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_distributor	X	movie:film_distributor_film_distributorid	
		movie:film_distributor_freebase_url	
		movie:film_distributor_name	
		rdf:type	
		rdfs:label	
movie:film_featured_song	X	movie:film_featured_song_film_featured_song_id	
		movie:film_featured_song_freebase_url	
		movie:film_featured_song_name	
		movie:film_featured_song_performed_by	
		rdf:type	
		rdfs:label	
movie:film_festival	X	movie:film_festival_event	X

movie:film_festival		movie:film_festival_film_festivalid	
		movie:film_festival_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_festival_event	X	movie:film_festival_event_closing_date	
		movie:film_festival_event_film_festival_event_id	
		movie:film_festival_event_freebase_url	
		movie:film_festival_event_name	
		movie:film_festival_event_opening_date	
		rdf:type	
		rdfs:label	
movie:film_festival_focus		movie:film_festival_focus_film_festival_focus_id	
		movie:film_festival_focus_freebase_url	
		movie:film_festival_focus_name	
		rdf:type	
		rdfs:label	
movie:film_festival_sponsor		movie:film_festival_sponsor_film_festival_sponsor_id	
		movie:film_festival_sponsor_freebase_url	
		movie:film_festival_sponsor_name	
		rdf:type	
		rdfs:label	
movie:film_film_company_relationship		movie:film_company	X
		movie:film_film_company_relationship_film_film_company_relationshipid	

movie:film_film_company_relationship		movie:film_film_company_relationship_freebase_url	
		movie:film_film_company_relationship_name	
		movie:film_film_company_relationship_role_service	
		rdf:type	
		rdfs:label	
movie:film_film_distributor_relationship		movie:film_distribution_medium	X
		movie:film_distributor	X
		movie:film_film_distributor_relationship_distributor	
		movie:film_film_distributor_relationship_film_cut	
		movie:film_film_distributor_relationship_film_distribution_medium	
		movie:film_film_distributor_relationship_film_film_distributor_relationshipid	
		movie:film_film_distributor_relationship_freebase_url	
		movie:film_film_distributor_relationship_name	
		movie:film_film_distributor_relationship_region	
		movie:film_film_distributor_relationship_year	
		movie:film_of_distributor	X
		rdf:type	
		rdfs:label	
movie:film_format	X	movie:film_format_film_formatid	
		movie:film_format_name	
		rdf:type	
		rdfs:label	
		foaf:page	

movie:film_genre	X	movie:film_genre_film_genreid	
		movie:film_genre_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_job	X	movie:film_job_film_jobid	
		movie:film_job_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_location	X	movie:film_location_film_locationid	
		movie:film_location_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_production_designer	X	movie:film_production_designer_film_production_designer_id	
		movie:film_production_designer_freebase_url	
		movie:film_production_designer_name	
		rdf:type	
		rdfs:label	
movie:film_regional_release_date		movie:film_regional_release_date_film_regional_debut_venue	
		movie:film_regional_release_date_film_regional_release_id	
		movie:film_regional_release_date_film_release_distribution_medium	
		movie:film_regional_release_date_film_release_region	
		movie:film_regional_release_date_freebase_url	

movie:film_regional_release_date		movie:film_regional_release_date_name	
		movie:film_regional_release_date_release_date	
		rdf:type	
		rdfs:label	
movie:film_screening_venue		movie:film_screening_venue_film_screening_venue_id	
		movie:film_screening_venue_freebase_url	
		movie:film_screening_venue_name	
		rdf:type	
		rdfs:label	
movie:film_series	X	movie:film_series_film_seriesid	
		movie:film_series_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_set_designer	X	movie:film_set_designer_film_set_designer_id	
		movie:film_set_designer_freebase_url	
		movie:film_set_designer_name	
		rdf:type	
		rdfs:label	
movie:film_story_contributor	X	movie:film_story_contributor_film_story_contributorid	
		movie:film_story_contributor_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_subject		movie:film_subject_film_subjectid	
		movie:film_subject_name	

movie:film_subject		rdf:type	
		rdfs:label	
		foaf:page	
movie:film_theorist	X	movie:film_theorist_film_theoristid	
		movie:film_theorist_name	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:music_contributor	X	movie:music_contributor_music_contributorid	
		movie:music_contributor_name	
		rdf:type	
		rdfs:label	
		owl:sameAs	
		foaf:page	
movie:performance		movie:film_character	X
		movie:performance_actor	
		movie:performance_character	
		movie:performance_film	
		movie:performance_name	
		movie:performance_note	
		movie:performance_performanceid	
		movie:performance_special_performance_type	X
		rdf:type	
		rdfs:label	
		foaf:page	
movie:personal_film_appearance		movie:personal_film_appearance_name	
		movie:personal_film_appearance_person	
		movie:personal_film_appearance_personal_film_appearanceid	

movie:personal_film_appearance		movie:personal_film_appearance_type	X
		movie:personal_film_appearance_type_of_appearance	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:personal_film_appearance_type		movie:personal_film_appearance_type_name	
		movie:personal_film_appearance_type_personal_film_appearance_typeid	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:producer	X	movie:producer_name	
		movie:producer_producerid	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:production_company	X	movie:production_company_name	
		movie:production_company_production_companyid	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:special_film_performance_type		movie:special_film_performance_type_name	
		movie:special_film_performance_type_special_film_performance_typeid	
		rdf:type	
		rdfs:label	
		foaf:page	
movie:writer	X	movie:writer_name	
		movie:writer_writerid	
		rdf:type	
		rdfs:SeeAlso	

movie:writer		rdfs:label	
		foaf:page	
oddlinker:interlink		movie:linkid	
		oddlinker:link_source	X
		oddlinker:link_target	
		oddlinker:link_type	
		oddlinker:linkage_run	X
		oddlinker:linkage_score	
		rdf:type	
		rdfs:label	
oddlinker:linkage_run		oddlinker:linkage_date	
		oddlinker:linkage_method	
		rdf:type	
foaf:Agent		movie:film_company_film_companyid	
		movie:film_company_name	
		movie:film_costume_designer_film_costume_designerid	
		movie:film_costume_designer_name	
		rdf:type	
		rdfs:label	
		foaf:page	
foaf:Person	X	movie:actor_actorid	
		movie:actor_name	
		movie:actor_netflix_id	
		movie:actor_nytimes_id	
		movie:cinematographer_cinematographerid	
		movie:cinematographer_name	
		movie:director_directorid	
		movie:director_name	
		movie:editor_editorid	
		movie:editor_name	
		movie:film_crewmember_film_crewmemberid	

foaf:Person		movie:film_crewmember_name	
		movie:performance	X
		rdf:type	
		rdfs:label	
		owl:sameAs	
		foaf:made	
		foaf:page	

Apêndice B

Quadro B.1: Classes e Propriedades utilizadas na descrição dos dados da Linked Brainz. (dc = http://purl.org/dc/terms/, foaf = http://xmlns.com/foaf/0.1/, geo = http://www.w3.org/2003/01/geo/wgs84_pos#, mo = http://purl.org/ontology/mo/, rdf = http://www.w3.org/1999/02/22-rdf-syntax-ns#, rdfs = http://www.w3.org/2000/01/rdf-schema#, skos = http://www.w3.org/2004/02/skos/core# e vocab = http://open.vocab.org/terms/)

Classe	Identifica EN	Propriedade	Relacionamento de Domínio
mo:Composition		mo:produced_work	X
		rdf:type	
mo:Label	X	skos:altLabel	
		foaf:name	
		vocab:sortLabel	
		rdfs:label	
		rdf:type	
mo:MusicalWork	X	mo:composed_in	X
		dc:title	
		rdfs:label	
		rdf:type	
mo:MusicArtist	X	foaf:name	
		vocab:sortLabel	
		foaf:made	X
		skos:altLabel	
		rdf:type	
mo:MusicGroup	X	foaf:name	

mo:MusicGroup		vocab:sortLabel	
mo:MusicGroup		foaf:made	X
		skos:altLabel	
		rdf:type	
mo:Record		mo:track	X
		mo:track_count	
		rdf:type	
mo:Release	X	mo:label	X
		mo:publishing_location	X
		mo:record	X
		foaf:maker	X
		mo:release_event	X
		dc:title	
		rdfs:label	
		rdf:type	
mo:ReleaseEvent		dc:date	
		rdf:type	
mo:Signal	X	mo:isrc	
		mo:puid	
		mo:duration	
		mo:duration_ms	
		foaf:maker	X
		dc:title	
		rdfs:label	
		rdf:type	
mo:SignalGroup	X	foaf:maker	X
		dc:title	
		rdfs:label	
		rdf:type	
mo:SoloMusicArtist	X	foaf:name	
		vocab:sortLabel	
		foaf:made	X
		skos:altLabel	
		rdf:type	
mo:Track	X	dc:title	
		rdfs:label	
		mo:published_as	X
		mo:track_number	
		rdf:type	

geo:Country	X	rdfs:label	
		rdf:type	

Apêndice C

Quadro C.1: Classes e Propriedades utilizadas na descrição dos dados da Magnatune (DBTune.org). (bio = http://purl.org/vocab/bio/0.1/, dc = http://purl.org/dc/elements/1.1/, event = http://purl.org/NET/c4dm/event.owl#, foaf = http://xmlns.com/foaf/0.1/, mo = http://purl.org/ontology/mo/, rdf = http://www.w3.org/1999/02/22-rdf-syntax-ns#, rdfs = http://www.w3.org/2000/01/rdf-schema#, time = http://www.w3.org/TR/owl-time e timeline = http://purl.org/NET/c4dm/timeline.owl#)

Classe	Identifica EN	Propriedade	Relacionamento de Domínio
timeline:RelativeTimeLine		rdf:type	
		rdfs:label	
mo:MusicArtist	X	dc:description	
		bio:olb	
		rdf:type	
		rdfs:label	
		foaf:based_near	
		foaf:homepage	
		foaf:img	
		foaf:name	
mo:Performance		event:place	
		mo:performer	X
		mo:recorded_as	X
		rdf:type	
		rdfs:comment	

mo:Performance		rdfs:label	
mo:Record	X	dc:title	
		mo:publishing_location	
		mo:track	X
		rdf:type	
		rdfs:label	
		foaf:maker	X
mo:Signal		mo:published_as	X
		mo:time	X
		rdf:type	
		rdfs:label	
mo:Track	X	dc:created	
		dc:title	
		mo:available_as	
		mo:paid_download	
		mo:track_number	
		rdf:type	
		rdfs:label	
		foaf:maker	X
time:Interval		timeline:duration	
		timeline:onTimeLine	X
		rdf:type	
		rdfs:label	

Apêndice D

Quadro D.1: Classes e Propriedades utilizadas na descrição dos dados da Jamendo (DBTune.org). (dc = http://purl.org/dc/elements/1.1/, foaf = http://xmlns.com/foaf/0.1/, mo = http://purl.org/ontology/mo/, rdf = http://www.w3.org/1999/02/22-rdf-syntax-ns#, tag = http://www.holygoat.co.uk/owl/redwood/0.1/tags/, time = http://www.w3.org/2006/time#, timeline = http://purl.org/NET/c4dm/timeline.owl# e owl = http://www.w3.org/2002/07/owl#)

Classe	Identifica EN	Propriedade	Relacionamento de Domínio
mo:ED2K		rdf:type	
		dc:format	
mo:Lyrics		rdf:type	
		mo:text	
mo:MusicArtist	X	rdf:type	
		foaf:made	X
		foaf:based_near	
		foaf:homepage	
		foaf:img	
		foaf:name	
		mo:biography	
		owl:sameAs	
mo:Playlist		rdf:type	
		dc:format	
mo:Record	X	rdf:type	

mo:Record		dc:date	
mo:Record		mo:available_as	
		mo:image	
		mo:track	X
		tag:taggedWithTag	X
		dc:title	
		foaf:maker	X
		dc:description	
		owl:sameAs	
mo:Signal		rdf:type	
		mo:published_as	X
		mo:time	X
mo:Torrent		rdf:type	
		dc:format	
mo:Track	X	rdf:type	
		mo:available_as	
		dc:title	
		mo:license	
		mo:track_number	
		owl:sameAs	
tag:Tag		rdf:type	
		tag:tagName	
time:Interval		rdf:type	
		timeline:onTimeLine	
foaf:Document		rdf:type	