UNIVERSIDADE DO EXTREMO SUL – UNESC

CIÊNCIA DA COMPUTAÇÃO

FELIPE COGORNI MAURICIO

ANALISE DE TÉCNICAS DE EXTRAÇÃO DE DADOS NÃO ESTRUTURADOS EM PÁGINAS HTML PARA UTILIZAÇÃO NO ARMAZENAMENTO E MANIPULAÇÃO EM BANCO DE DADOS OBJETO RELACIONAL

CRICIÚMA, JULHO DE 2011

FELIPE COGORNI MAURICIO

ANALISE DE TÉCNICAS DE EXTRAÇÃO DE DADOS NÃO ESTRUTURADOS EM PÁGINAS HTML PARA UTILIZAÇÃO NO ARMAZENAMENTO E MANIPULAÇÃO EM BANCO DE DADOS OBJETO RELACIONAL

Trabalho de Conclusão de Curso apresentado para obtenção do Grau de Bacharel em Ciência da Computação da Universidade do Extremo Sul Catarinense.

Orientador: Prof. MSc. Paracelso de Oliveira Caldas

CRICIÚMA, JULHO DE 2011

FELIPE COGORNI MAURICIO

Analise de Técnicas de Extração de Dados Não Estruturados em Páginas HTML para Utilização no Armazenamento e Manipulação em Banco de Dados Objeto Relacional

Submetido ao corpo docente do Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense como um dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Prof. MSc. Ana Claudia Garcia Barbosa
Coordenadora do Curso de Ciência da Computação

Banca Examinadora:

Prof. MSc. Paracelso de Oliveira Caldas (UNESC)
Orientador

Prof. MSc. Gustavo Bisognin (UNESC)

Prof. Esp. Luciano Antunes (UNESC)

Aos meus pais, Adalberto e Marisselma, minha noiva Geter e meus amigos pelo apoio concedido.

AGRADECIMENTOS

Agradeço a Deus por estar presente em minha vida, por me confortar nos momentos difíceis e por guiar meus caminhos me protegendo em minhas escolhas.

Aos meus pais por todo o amor, carinho e dedicação na minha formação pessoal. Por me apoiarem em minhas decisões e por ajudar em tudo que eu precisei.

A minha noiva Geter pelo amor, carinho e por fazer parte de minha vida. Por sua compreensão e por muito me apoiar nos momentos difíceis.

A todos os meus colegas de curso, por me ajudarem em minha formação e por compartilharem momentos de diversão e descontração ao longo da graduação, especialmente meu amigo de infância Luis que além de compartilhar os melhores momentos da minha vida juntamente com Rafael e Renato, foi parte fundamental na minha formação no curso.

Aos amigos e colegas do Corpo de Bombeiro que permitiram meu estudo “segurando as pontas” na minha ausência.

E finalmente, ao meu incentivador desta pesquisa, meu professor e orientador Paracelso, que me auxiliou em minhas tomadas de decisões, me apoiando e contribuindo com seu conhecimento e sua amizade.

Enfim, a todos que de alguma forma contribuíram para a conclusão desta pesquisa.

“Não basta estar certo, tem que valer a pena”.

RESUMO

A internet hoje é o local que mais se acessa para a realização de pesquisas, porém devido a falta de padronização na construção de sites, especificamente nas páginas HTML, existe uma perda bastante significativa de dados que poderiam ser melhores aproveitados, são os casos dos dados não estruturados em páginas HTML. Para poder utilizar esses dados de uma forma relevante existem técnicas que auxiliam na sua extração. Dentre as técnicas disponíveis, estão as de Processamento de Linguagem Natural, probabilidades e árvores de decisão. Com isso esta pesquisa fundamentou-se na implementação de um protótipo que utiliza de uma ferramenta chamada TreeTagger que implementa essas técnicas, para ajudar na tarefa de extração dos dados de maneira contextualizada e para armazená-los em um banco de dados, facilitando assim a sua manipulação e obtenção de resultados mais relevantes. Para a contextualização é usada a frase de busca feita por um usuário em uma ferramenta de busca na web, no intuito de comparação da frase com o conteúdo na página HTML. Durante a pesquisa, foram realizados alguns testes no protótipo implementado, a fim de verificar os resultados obtidos pelo protótipo e comprovar o êxito nos objetivos do trabalho.

Palavras-chave: Dados Não Estruturados; Páginas HTML; Técnicas de Extração; Processamento de Linguagem Natural; Técnicas de Contextualização; Ferramenta *TreeTagger*.

ABSTRACT

The Internet today is the place where most accesses are done to conduct a research, but due to lack of standardization in the building of sites, specifically in HTML pages, there is a very significant loss of data which could be better utilized, for instance the data unstructured of HTML pages. To use these data in a relevant way, there are techniques that help with the extraction of such data. Among the techniques available there are Natural Language Processing, probability and decision trees. Therefore this research was based on implementing a prototype that uses a tool called TreeTagger that implements the techniques mencioned to help in the task of extracting the data in context and to store them in a database, thus facilitating its handling and in the obtaince for more relevant results. The contextualizing process occurs by a phrase used by the user in a web tool, in order to compare the sentence with the content in the HTML page. During the research some tests were conducted on the prototype implemented in order to verify the results obtained by the prototype and demonstrate their success in the work objectives.

Keywords: Unstructured Data; HTML Pages; Extraction Techniques; Natural Language Processing; Contextualization Techniques; Tool TreeTagger.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

BDOR	Banco de Dados Objeto Relacional
BDOO	Banco de Dados Orientado a Objeto
BDR	Banco de Dados Relacional
BLOB	*Binary Large Objects*
CLOB	*Character Large Objects*
EI	Extração de Informação
HTML	*HyperText Markup Language*
MD	Mineração de Dados
MT	Mineração de Texto
NoDoSe	*Northwestern Document Structure Extractor*
PLN	Processamento de Língua Natural
RI	Recuperação de Informação
SGBD	Sistema de Gerenciamento de Banco de Dados

SUMÁRIO

1 INTRODUÇÃO

Considerando o crescente aumento na quantidade de dados estruturados e não estruturados em diversas bases de dados podem ser observadas duas situações. Em se tratando de sistemas que contem dados estruturados, existe hoje uma infinidade de ferramentas para lidar com esses dados, muito exploradas pela área de TI. Porém, no contexto de sistemas que apresentam dados não estruturados, ainda se tem uma carência, sendo de difícil tratamento e prejudicando a manipulação para obtenção de informações.

Sistemas estruturados são sistemas que possuem uma estrutura predefinida e conhecida. Sistemas não estruturados não possuem uma estrutura prévia para se tratar os dados. São exemplos desses dados: email, fax, documentos de texto, algumas páginas HTML, imagens, entre outros.

Para que se tenham textos não estruturados com relevância, é necessário descobrir seus conceitos dominantes, ou seja, sua idéia central, para classificá-los em grupos, observando suas similaridades de idéias e conteúdos, com base na ocorrência de seus vocábulos (ÁLVARES, 2007).

Analisando a complexidade e a grande quantidade de dados não estruturados no meio eletrônico, principalmente em páginas web, observa-se a necessidade de extrair e modelar informações contidas nesse meio para que possam ser manipuladas por técnicas e ferramentas já utilizadas em banco de dados estruturados (CALDAS, 2003).

Deve-se levar em conta estruturas internas e externas que as interconectam. Atualmente, é muito utilizada a pesquisa por frases ou palavras chaves nas páginas web, esforços têm sido feitos para quebrar esse paradigma, utilizando largamente as estruturas de ligações da web nessas consultas, a fim de melhorar o desempenho de busca na rede, que muito depende dos dados não estruturados.

Existem três classes de problemas relacionados à obtenção dos tipos de informações supracitadas, que são: modelagem e pesquisa, extração e integração de dados e construção e reestruturação.

As linguagens tradicionais não são dirigidas para dados não estruturados. Dessa forma é necessário modificá-las ou remodelá-las. Existem ferramentas, como por exemplo, StruQL, OQL, que utilizam gráficos rotulados como um modelo de dados flexíveis para aplicar uma modelagem (SILVEIRA, 2009).

Para obter uma extração dos dados existem técnicas, cada uma com suas características, que podem, de maneira efetiva, extrair os dados relevantes. Porém, para cada caso, se tem a necessidade de pesquisar melhor cada técnica dependendo de sua finalidade. Entre elas encontramos: técnica baseada em Processamento de Linguagem Natural, baseada em ontologias, entre outras.

Existe também uma técnica chamada de processo de conhecimento de texto (Text Mining), um processo não trivial que extrai padrões úteis e interessantes de conhecimento de documentos não estruturados (ÁLVARES, 2007).

A construção e reestruturação é um processo pela qual podem ser usadas técnicas de SGBD em sistemas de dados não estruturados, após sua armazenagem e estruturação, para sua remodelação, manutenção e manipulação (SILVEIRA, 2009).

Mediante esse contexto, considerando a necessidade de armazenamento de dados não estruturados, que contém muita informação relevante, e as diferentes abordagens para sua realização, esta pesquisa compreende o estudo, definição e aplicação dos métodos, metodologias e métricas para extração de dados não estruturados em páginas HTML, e armazená-las em um banco de dados objeto relacional.

1.1 OBJETIVO GERAL

Analisar e identificar entre as técnicas de extração de conhecimento para dados não estruturados, qual a solução mais adequada para manipulação desses dados e armazenamento em um banco de dados objeto-relacional.

1.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos desta pesquisa consistem em:

a) estudo de técnicas de extração de dados não estruturados;
b) compreensão dos processos de modelagem, extração e reconstrução de dados não estruturados;
c) entendimento do processo de armazenagem de Banco de Dados Objeto-Relacional;
d) analise da estruturação de páginas HTML;
e) seleção e aplicação de técnicas, métodos, metodologias e métricas para estruturação de dados não estruturados em HTML;
f) oferecer um protocolo de extração e armazenagem de dados não estruturados em Banco de Dados Objeto-Relacional.

1.3 JUSTIFICATIVA

O Compartilhamento de informações, por meio da web e alguns outros meios de comunicação, aproximam países e empresas de todos os ramos de atividades. Este compartilhamento volta a atenção dessas empresas para que gerenciem tais informações de

forma estratégica. Através disso, obtêm-se redução de riscos, crises e também o aumento da competitividade (ARAUJO, 2007).

A maior parte da informação disponível na web encontra-se nas páginas HTML. Essas páginas são formadas por dois tipos de informações: as de formatação e as de conteúdo. As de formatação servem para definirem os elementos na página, e as de conteúdo servem para identificarem e qualificarem objetos, esses conteúdos são as informações úteis para um usuário e para que eles possam manipulá-las é necessário extraí-las (CALDAS, 2003).

A valorização da informação e de seus conhecimentos pode ajudar a entender a crescente importância do processo de extração de conhecimento em documentos não estruturados (ARAUJO, 2007).

A web contém informações armazenadas sob diversas maneiras. Utilizando algumas técnicas, como wrappers, que é uma abordagem de extração sintática, somando com outra abordagem como a de ontologia, para extração semântica, é possível estruturar informações a princípio não estruturadas (VIVAN, 2003).

Para verificar ainda grandes volumes de dados, e garimpar conhecimento útil, não somente para os usuários de documentos eletrônicos, mas também para qualquer domínio que utiliza textos não estruturados, possibilitando estratégias organizacionais, temos dentro da Mineração de Dados, ferramentas para Text Mining (Mineração de texto), que através, por exemplo, da freqüência de algumas palavras se verifica a sua importância relativa e de suas sentenças (ARAUJO, 2007).

Apesar de existirem ferramentas a disposição para se obter a extração de dados, falta ainda ferramentas que utilizem do mesmo principio para a mesma extração, armazenamento e consequentemente manipulação de dados não estruturados. Nesta pesquisa é proposto uma sensível contribuição na verificação e analise das técnicas existentes para dados não estruturados, visando às páginas HTML, selecionando a melhor ou as melhores técnicas

para disponibilização do resultado extraído em Banco de Dados Objeto Relacional, permitindo assim a utilização de ferramentas tradicionais.

1.4 ESTRUTURA DO TRABALHO

Este trabalho esta dividido em sete capítulos, considerando que o Capítulo 1 é composto pela contextualização ao tema proposto, objetivos e justificativa para realização do trabalho.

São abordados conceitos de Banco de Dados e os seus três modelos, falando ainda mais especificamente do Banco de Dados Objeto Relacional, escolhido para ser usado no objetivo da pesquisa no Capítulo 2.

Conceitos de dados não estruturados, algumas técnicas para extração de dados não estruturados e alguns projetos desenvolvidos por meio delas são vistos no Capítulo 3.

No Capítulo 4 são apresentados alguns métodos de contextualização usados na Recuperação da Informação, que serão analisados para sua melhor aplicação no projeto de pesquisa e conceitos da linguagem HTML.

A ferramenta *TreeTagger* bem como sua utilização são apresentados no Capítulo 5. São mostrados alguns trabalhos correlatos com esse tema no Capítulo 6.

No Capítulo 7 é descrito o trabalho desenvolvido, a extração de dados não estruturados por meio da ferramenta *TreeTagger* , o armazenamento desses dados em um banco de dados e os resultados obtidos.

Por fim, tem-se a conclusão desta pesquisa, onde também são comentadas algumas sugestões para trabalhos futuros.

2 BANCO DE DADOS

Dados são valores de campos armazenados, matéria-prima para obtenção de informação. Base de dados é uma coleção de dados relacionados logicamente, e que representa um aspecto da vida real (DATE, 2000).

Os sistemas de banco de dados são considerados o equivalente eletrônico de um armário de arquivamento. Em suma são sistemas computadorizados de armazenamento de registros ou dados, onde um usuário pode inserir, excluir, editar, buscar, ou seja, consegue manipular esses dados (DATE, 2000).

Silva (2002) fala ainda que banco de dados é um conjunto de informações manipuláveis de mesma natureza inseridas em um mesmo local, e obedece a um padrão de armazenagem.

Os componentes de hardware de um sistema de banco de dados consistem em: volumes de armazenamento secundário – principalmente discos magnéticos, juntamente com os dispositivos de entrada e saída, canais de dispositivos e assim por diante (DATE, 2000).

Entre a estrutura física de um banco de dados e o usuário está o Sistema de Gerenciamento de Banco de Dados (SGBD), que é o software ou ainda a parte lógica que controla o banco de dados (DATE, 2000).

Os SGBD são estruturas de arquivos e programas que possibilitam a interação do usuário com o banco de dados. Um dos maiores benefícios é a abstração de dados para o usuário, ou seja, se oculta a forma de armazenamento e manutenção dos dados para o mesmo (SILBERSCHATZ; KORTH; SUDARSHAN, 2006).

Os SGBD utilizam diferentes formas de representação, ou modelos de dados, para descrever a estrutura das informações contidas em seus bancos de dados. Atualmente, os

seguintes modelos de dados são mais utilizados pelos SGBD: modelo relacional, orientado a objetos e objeto relacional (DATE, 2000).

2.1 MODELO RELACIONAL, ORIENTADO A OBJETOS E OBJETO RELACIONAL DE SGBD

O Banco de Dados Relacional (BDR), modelo relacional, foi proposto pelo Dr. E. F. Codd em 1970. O modelo relacional é proposto na forma bi-dimensional (linhas e colunas), onde linhas representam os elementos e as colunas os atributos (SEGUNDO; ROCHA; NEVES, 2008).

Na necessidade de representar objetos complexos foi desenvolvido um sistema de dados orientado a objeto (modelo orientado a objetos), ou Banco de Dados Orientado a Objeto (BDOO). Dessa forma o sistema tem também as características principais da orientação a objetos que são o encapsulamento, herança e polimorfismo. Porém a grande desvantagem do BDOO é o seu pobre desempenho (SEGUNDO; ROCHA; NEVES, 2008).

O Banco de Dados Objeto Relacional (BDOR) envolve o benefício dos dois modelos anteriores, tanto BDR na facilidade de uso dos métodos de consulta, quanto BDOO com escalabilidade e suporte a tipos complexos de dados (SEGUNDO; ROCHA; NEVES, 2008).

A Figura 1 mostra a classificação das aplicações SGBD e foram divididas em quatro tipos.

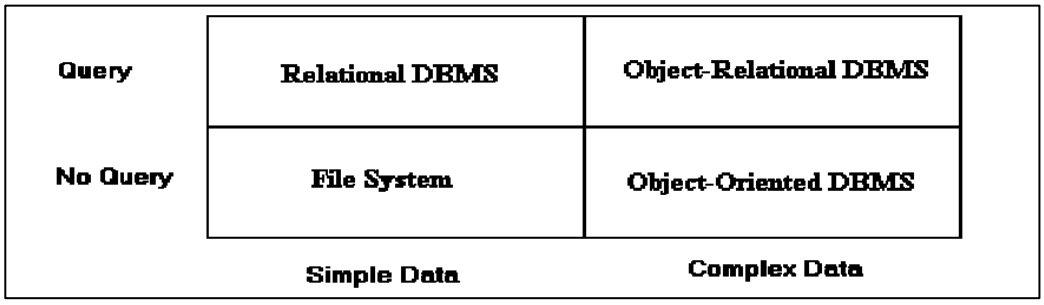


Figura 1. Classificação das Aplicações SGBD
Fonte: SEGUNDO, A.; ROCHA, G.; NEVES, N. (2008)

Conforme Segundo, Rocha e Neves (2008):

a) O quadrante inferior esquerdo (*File System*) mostra aplicações que manipulam dados simples e que não precisam de consultas a dados persistidos. Exemplos de aplicações deste tipo são: Word e Emacs. A única consulta e atualização existente neles é o *get file* e o *put_file* respectivamente, não necessitando então de qualquer outra consulta ou ainda do SQL.

b) No quadrante superior esquerdo (*Relational - DBMS*) mostra aplicações que armazenam registros estruturados, com atributos simples como inteiros, float e string de caracteres. Pode-se citar um banco simples para empregados de uma companhia. Pode ser requisitado uma consulta simples como, por exemplo, a média de salários dos empregados. Para esse exemplo um SQL-2, que permite a inserção, exclusão e consulta dos dados através de estruturas denominadas tabelas ou relações, em um Banco de dados Relacional já faz o necessário.

c) No quadrante inferior direito (*Object-Oriented - DBMS*) temos como exemplo aplicações CAD/CAM. Um projeto de chip é armazenado no disco como um objeto complexo.É visto que não há necessidade de consulta e um BDOO é o tipo de banco mais indicado neste caso.

d) No quadrante superior direito (*Object-Relational - DBMS*) sugere uma aplicação que tenha dados complexos e que necessita frequentemente uma consulta e concorrentemente a outras operações. Sendo assim, o BDOR corresponde a este caso. Bancos

assim são bastante úteis para dados não estruturados, por exemplo, imagens e textos muito frequentes em páginas web.

2.2 BANCO DE DADOS OBJETO RELACIONAL

Conforme Feitosa (2008) o modelo Objeto Relacional é uma extensão do modelo Relacional tradicional, adicionando a ele as características de orientação a objetos. Desta forma toda a tecnologia desenvolvida no modelo Relacional permanece, tais como suporte à linguagem de consulta SQL, gerência de transações, processamento e otimização de consultas.

A semântica da aplicação é modelada e representada por meio de objetos, enquanto sua implementação física é feita na forma relacional. Nesse caso é utilizado o SQL-3, que aceita os padrões SQL-2 sem nenhum problema, porém inclui características que auxiliam a manipulação de objetos, consultas recursivas, instruções de programação e etc.

Para representar um objeto que precise de mais de um registro para ser representado, o modelo relacional é estendido incorporando relações aninhadas. Isso significa que o domínio de um atributo pode ser identificado por uma relação, e não somente um valor atônito.

No caso de uma livraria, cada livro possui um título, tem um ou mais autores, tem um preço de venda, tem uma data de publicação, que pode ser representado na forma aninhada. A Figura 2, a seguir, mostra o exemplo da livraria:

Título	Autores		Preço Venda	Data Publicação		
	Nome	Sobrenome		Dia	Mês	Ano
Título 1	João	Silva	30,00	10	JUN	2004
	Maria	Santos				
Título 2	José	Souza	40,00	01	MAI	2005

Figura 2. Representação da relação livros
Fonte: FEITOSA, D.; et al (2008)

O domínio do atributo autores é uma relação, composta por sua vez de dois atributos: nome e sobrenome. Além disso, o atributo autores pode ser multivalorado, ou seja, pode ser composta por diversas tuplas. Isto caracteriza o que denomina relações aninhadas.

Outra característica importante é a possibilidade de definir tipos complexos de dados. As extensões definidas no modelo relacional não ficam restritas nas relações aninhadas. Podem-se definir domínios com valores multivalorados estruturados de diferentes formas, como por exemplo, arrays, sets ou multisets. Os valores desses domínios são sempre objetos que por sua vez podem ter mais objetos.

Esses tipos complexos de dados, também chamados de Tipos Abstratos de Dados (TADs), definem a estrutura dos objetos criados, seus comportamentos e possíveis heranças. Estes tipos de dados são armazenados no banco de dados, diferente dos tipos definidos em linguagem de programação, e que também só podem ser acessados por programas que contêm os tipos declarados em arquivos textuais.

Pode-se criar, por exemplo, tipos utilizados como domínios para nomes (TNome) e para datas (TData) e outro tipo para livros (TLivros), conforme observado na Figura 3.

```
CREATE ROW TYPE Tnome
{ nome VARCHAR(20),
sobrenome VARCHAR (20) },

CREATE ROW TYPE Tdata
{ dia INTEGER,
mes CHAR(3),
ano INTEGER }

CREATE ROW TYPE Tlivro
{ titulo VARCHAR(50),
lista_autores SET OF(TNome),
preco_venda INTEGER,
data_publicacao Tdata }
```

Figura 3. Declaração de tipos de dados
Fonte: SEGUNDO, A.; ROCHA, G.; NEVES, N. (2008)

Foram então criados tipos (TNome e TData) que foram usados em outro tipo (TLivro), essas declarações de tipos não criam tabelas, elas substituem a lista de atributos nos comandos que criam as tabelas (CREATE TABLE). Por exemplo, a declaração CREATE TABLE livros OF TYPE TLivro cria a tabela livros com o tipo definido na Figura 3.

Por meio de um tipo de dado criado previamente, pode-se definir diferentes tabelas, sendo diferenciadas apenas pelo nome. Assim, CREATE TABLE apostilas OF TYPE TLivro, usa o mesmo tipo de dado TLivro, porém para criar a tabela apostilas.

Mais uma característica é o armazenamento de objetos longos, ou seja, objetos complexos, como fotografias, imagens médicas de alta resolução, vídeos, textos longos e etc. Desta forma, os objetos da aplicação são representados diretamente no banco de dados, integrados aos outros dados, e não em arquivos separados. *Character Large Objects* (CLOB) para textos longos e Binary LOB (BLOB) para imagens.

Por tratar de objetos nesse BD é necessário usar o conceito de herança. Herança simples, pois heranças múltiplas não são contempladas nesse modelo. Herança de tipos é

representada por meio de supertipo/subtipo. Na Figura 4 observa-se um exemplo de uma pessoa dentro de uma livraria que pode ser um cliente ou um funcionário.

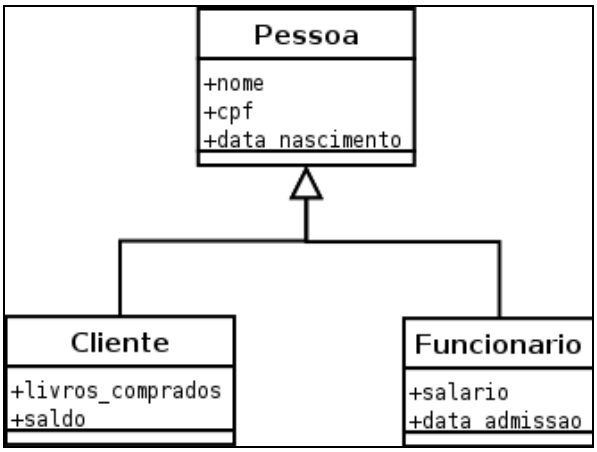


Figura 4. Superclasse Pessoa com subclasses
Fonte: SEGUNDO, A.; ROCHA, G.; NEVES, N. (2008)

Esse Banco de Dados foi escolhido devido ao seu alto grau de representatividade, por ser um Banco de Dados muito explorado e que para esta pesquisa será necessário um lugar para trabalhar com o armazenamento das extrações finais.

3 TÉCNICAS PARA EXTRAÇÃO DE DADOS NÃO ESTRUTURADOS

Dados não estruturados são textos livres não estruturados, ou seja, não apresentando regularidade para sua disposição. Não são facilmente extraídos por ser difícil detectá-los, ao menos que se tenha um conhecimento lingüístico sobre eles. Um exemplo de armazenamento de dados não estruturados é a Web (ÁLVARES, 2007).

Cada vez mais se encontra dados não estruturados em base de dados de organizações, são necessárias técnicas para poder extrair informações relevantes para que possam ser usadas na tomada de decisões (ALBERTO FILHO, 2009).

Com a crescente disseminação de informações na Web e paralelamente a crescente utilização desses mecanismos, se torna necessário um melhor gerenciamento de tais informações através de conceitos de BD. Esse fato é devido aos dados da Web não serem estruturados como, por exemplo, textos, figuras, arquivos de áudio e vídeo (SILVEIRA, 2009).

Com o surgimento da internet os dados não estruturados, formados por textos, imagens, gráficos e etc., chamaram a atenção. Esses tipos de dados possuem as mesmas informações que dados estruturados, porém sua forma é textual dificultando a analise de suas características principais (WIVES, 1999).

A análise de dados não estruturados se torna mais difícil justamente pela falta de estrutura, logo são necessárias técnicas e ferramentas para tratamento desses dados específicos (MORAIS, 2007).

As aplicações de extração de informações podem, por exemplo, serem utilizadas para armazenar textos não estruturados em banco de dados tradicionais para que um usuário possa fazer uma consulta padrão (ZAMBENEDETTI, 2002).

Técnicas de extração da informação podem ser usadas para extrair desde textos estruturados como também sem estrutura nenhuma. Cabendo salientar que a definição de uma técnica é muito importante para cada determinado tipo de documento (ÁLVARES, 2007). Seguem nos próximos subtítulos algumas técnicas usadas na extração.

3.1 MINERAÇÃO DE DADOS

A Mineração de Dados é o processo para extrair informações válidas, previamente desconhecidas e de muito valor a partir de grandes bases de dados. Permite muito mais que uma simples consulta a um banco de dados, pois possibilita o usuário explorar e inferir informação útil a partir dos dados, descobrindo relacionamentos escondidos (ARAUJO, 2007).

Mineração de Dados (MD) por meio de um conjunto de técnicas providas da Estatística, Recuperação de Informação, Inteligência Artificial e Reconhecimento de Padrões, tem como objetivo descobrir conhecimento novo, útil, relevante e não-trivial que esteja em uma grande massa de dados (ALBERTO FILHO, 2009).

Existem vários modelos de MD os quais são classificados de acordo com a finalidade ou com a classe de aplicação para qual são usados. Cada classe possui um conjunto de algoritmos para extração de relações de uma base de dados. Os principais métodos de MD são Classificação, Associação, Agrupamento e Previsão de séries Seqüenciais – Temporais (ARAUJO, 2007).

A Classificação consiste em uma análise da propriedade de um conjunto de objetos de uma base de dados e posteriormente a classificação em diferentes classes predefinidas, de acordo com o modelo de classificação, seguindo os exemplos de tarefas de classificação (ARAUJO, 2007):

a) atribuir palavras-chave a artigos jornalísticos;

b) classificar pedidos de créditos como baixo, médio e alto risco;

c) determinar o número correspondente ao fax;

d) esclarecer pedidos de seguro fraudulentos e;

e) atribuir códigos industriais e designações de trabalho com descrições livres.

Associação usa um conjunto de itens de uma base de dados e verifica o quanto ele implica em outro conjunto de itens na mesma base, verificando assim as tendências de alguns dados para entender e explorar padrões de comportamento dos dados. Suporte e confiança são dois importantes parâmetros para realização da mineração de dados utilizando regras de associação. Suporte corresponde à freqüência com que todos os itens presentes em uma regra aparecem juntos em uma mesma transação da base de dados, mesmo que apareçam juntos com outros itens não presentes à regra. A confiança expressa o percentual de transações em que, ocorrendo o antecedente, ocorre o conseqüente da regra (ALBERTO FILHO, 2009).

Agrupamento é um grupo de objetos similares, geralmente uma classe, que possui um título comum que representa todos os registros nela contidos. Faz-se assim de um grupo maior mais heterogêneo, grupos menores ou clusters de forma mais homogênea. No agrupamento não há classes pré-definidas, os registros são agrupados de acordo com a semelhança (ALBERTO FILHO, 2009).

Os padrões seqüenciais temporais procuram eventos ou compras que acontecem seqüencialmente em um período de tempo, determinando tendências. Uma aplicação típica é a venda por mala direta, que agrega os dados sobre os produtos adquiridos em cada compra. Analisando nesse caso padrões de produtos comprados durante um determinado tempo. No agrupamento não há classes pré-definidas, os registros são agrupados de acordo com a semelhança (ARAUJO, 2007).

3.2 MINERAÇÃO DE TEXTO

O processo de Mineração de Texto (MT) tem como objetivo coletar informações significantes e conhecimentos relevantes em documentos textuais. Procedimento que implica em um grau de dificuldade grande, pois geralmente em documentos textuais é usado linguagem natural, sem a preocupação com a padronização ou estruturação dos dados (ALBERTO FILHO, 2009).

MT é o conjunto de técnicas e ferramentas inteligentes que auxiliam na análise de grandes volumes de dados textuais, com intuito de filtrar conhecimento útil, em qualquer domínio que utiliza textos não estruturados (ARAUJO, 2007).

Existe uma grande diferença entre a MT e as ferramentas de busca, como por exemplo, o G*oogle*. Nessas ferramentas é utilizado o uso de busca exaustiva, suas pesquisas geralmente retornam uma lista de documentos (página web, textos, arquivos pdf) relevantes com palavras-chave, ordenados pela proporção em que essas palavras são encontradas nesses documentos. Após a busca é necessário então analisar e ler o resultado para extrair o conhecimento.

MT, ao contrário dessas ferramentas, utilizam métodos de busca baseados em análise gramaticais e léxicas ou ainda técnicas de *clustering* (agrupamento). Isso permite descobrir entre os documentos, frases ou palavras similares. MT consegue ainda, através de técnicas de visualização de dados, mostrar conceitos-chaves e relações entre palavras e idéias, fazendo com que seja possível encontrar outros documentos (SILVA, 2002).

A área de MT envolve quatro etapas principais: a Coleta de Dados, a de Pré-processamento ou preparação dos dados, a etapa de Análise dos Dados e Extração do Conhecimento e a etapa de Pós-processamento (ARAUJO, 2007).

Na Coleta de Dados os documentos relacionados com a aplicação final são coletados. Na maioria das vezes, isso é feito automaticamente (ARAUJO, 2007).

O Pré-processamento consiste em remover os dados que são desnecessários para o entendimento do texto e facilitar a analise da etapa seguinte. A fase de pré-processamento pode ser dividida em três grandes etapas: correção ortográfica, remoção de Stopwords e o Stemming. Essas fases devem ser seguidas na seqüência, porém nem todas precisam ser usadas (ALBERTO FILHO, 2009).

Na correção ortográfica são analisados e eliminados os possíveis erros ortográficos do texto. Um verificador ortográfico compara as palavras através de um dicionário, caso alguma coincidência seja identificada ela é considerada sintaticamente correta (ALBERTO FILHO, 2009).

As Stopwords são palavras que se repetem muito no decorrer do texto ou palavras sem relevância aparente para o entendimento do texto, sendo então eliminadas (ALBERTO FILHO, 2009).

Existem diversas variações para uma palavra em texto, como por exemplo, plural, formas de gerúndio e sufixos temporais. Na fase de Stemming se faz a remoção dessas variações, obtendo o resultado chamado Stem (Raiz) (ARAUJO, 2007).

Na etapa de Extração de Conhecimento se utiliza algoritmos para extrair a partir de documentos pré-processados, conhecimento na forma de regras de associação, relação, segmentação, classificação de textos, entre outros.

O pós-processamento é a etapa que se analisa, filtra e seleciona os resultados obtidos para um melhor entendimento dos textos coletados (ARAUJO, 2007). Na Figura 5 se observa os processos de MT:

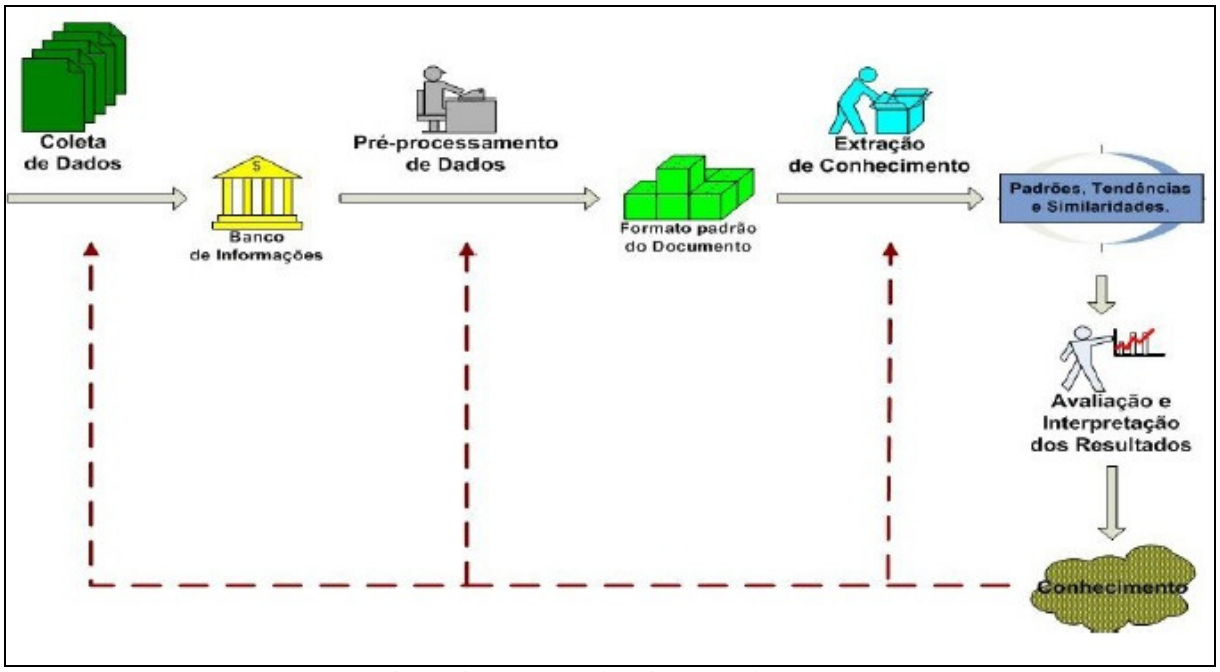


Figura 5. Processos de Mineração de Texto
Fonte: ARAUJO J. (2007)

MT frequentemente utiliza de experiência e resultados de pesquisas em recuperação de informação e processamento de língua natural (ÁLVARES, 2007).

A área de Recuperação de Informação é definida como o processo de busca de documentos relevantes em resposta a uma consulta que satisfaça a necessidade do usuário.

Pode ser usado, por exemplo, métodos estatísticos para processamento automático de textos e comparação a consulta dada (ÁLVARES, 2007).

O Processamento de Língua Natural (PLN) consistem no desenvolvimento de métodos computacionais para poder utilizar informações expressas em língua natural e trabalhar em cima dessas informações de forma de melhor entender tais línguas usando computadores. Por exemplo, uma grande quantidade de textos em língua natural pode ser mais compreendida e útil se for sumarizada (ÁLVARES, 2007).

Pode-se afirmar os seguintes pontos com relação a MT:

a) concentra algoritmos inteligentes e, em alguns casos, análise léxica;

b) processa documentos automaticamente. Categoriza, classifica ou constrói árvores de tópicos e índices de documentos;

c) indexação de conceitos entre os textos;

d) permite visualização do escopo global dos dados e sua relevância;

e) permite aos usuários fazerem associações, correlacionamentos e âncoras entre os documentos para posterior análise.

3.3 TÉCNICAS DE INDUÇÃO DE WRAPPERS

Wrappers são ferramentas de consulta e atualização de dados em uma fonte. Basicamente ele faz uma seleção de dados relevantes para serem extraídos e planeja sua estrutura para serem apresentados. As operações de um wrapper são feitas através de uma linguagem de extração específica (CALDAS, 2003).

Sistemas wrappers localizam informações relevantes em textos estruturados, utilizando esses dados, por exemplo, em um banco de dados. Porém, no contexto da web, o propósito de um wrapper é converter informações implícitas armazenadas em páginas HTML em informações explícitas estruturadas, para posterior processamento (ÁLVARES, 2007).

Um wrapper é um componente de software que transforma um modelo de dados para outro modelo, assim dados de modelos diferentes podem usar uma consulta comum para serem consultados em um modelo comum (GUANDELINE, 2002).

Wrappers possuem um conjunto de regras de extração para cada determinado domínio. Estas regras geralmente são baseadas em regras sintáticas que identificam os delimitadores da informação desejada em um documento (GUANDELINE, 2002).

Os wrappers podem ser construídos manualmente ou com ajuda de softwares para a criação de regras. O principal problema é que essas regras na maioria das vezes são baseadas na estrutura do documento, e os documentos web são muito dinâmicos mudando constantemente, sendo assim essas regras podem ficar incorretas (GUANDELINE, 2002).

Quanto melhor a estruturação dos dados em um documento mais fácil a aplicação de extração, por exemplo, no caso de um XML os dados estão mais claros, pois o foco esta nos dados, porém em HTML o foco esta na apresentação ficando mais difícil (GUANDELINE, 2002).

A extração sintática pode ser feita em qualquer linguagem com marcação tais como SGML, HTML, XML, entre outras. Neste tipo de caso os wrappers utilizam as tags como delimitadores, não existindo assim esquema sobre os dados que desejamos extrair (GUANDELINE, 2002).

Embora a sintaxe HTML não faça exigência de "*end-tags*", as aplicações que utilizam a extração sintática podem exigir "*start-tags*" e "*end-tags*" em todos os elementos do documento. Após a coleta, as informações passam para um modelo mais representativo para ser consultado através de alguma linguagem (GUANDELINE, 2002).

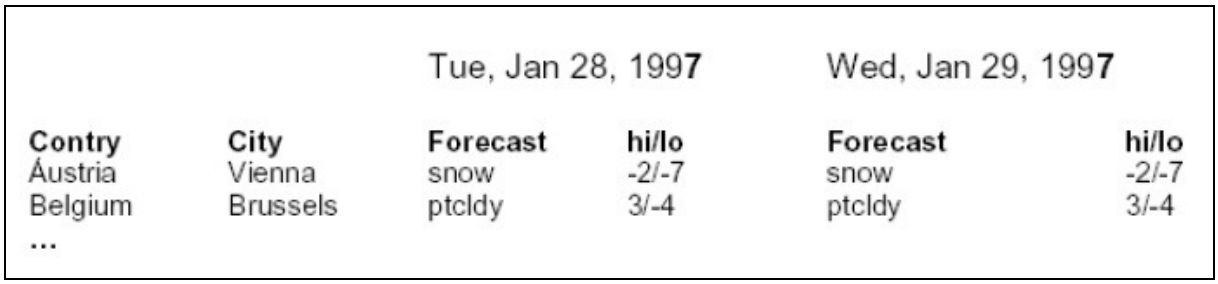
No próximo capítulo será mostrado um projeto chamado TSIMMIS, ele é um dos pioneiros na utilização de wrappers para extração de dados e alocar tais dados em uma estrutura bem definida (GUANDELINE, 2002).

3.3.1 Projeto Tsimmis

Essa ferramenta define como serão encontrados os dados no documento HTML e como serão "empacotados" e armazenados no modelo OEM – Object Exchange Model, que são objetos para Banco de Dados (GUANDELINE, 2002).

Inicialmente se especifica um arquivo de extração com a seguinte tipificação: [variável, fonte, modelo]. Variável é o nome da variável que receberá a informação extraída, fonte é o texto onde se deve encontrar o modelo, o modelo determina como se deve encontrar a informação.

Para demonstrar o método de extração em questão, será mostrado um exemplo detalhado a seguir. Supondo que em uma aplicação web que tenha como resultado um documento HTML com dados meteorológicos, como mostra a Figura 6.



		Tue, Jan 28, 1997		Wed, Jan 29, 1997	
Contry	**City**	**Forecast**	**hi/lo**	**Forecast**	**hi/lo**
Áustria	Vienna	snow	-2/-7	snow	-2/-7
Belgium	Brussels	ptcldy	3/-4	ptcldy	3/-4
...					

Figura 6. Fonte de dados HTML
Fonte: GUANDELINE, E. (2002)

O código gerado por essa aplicação pode ser visualizado na Figura 7.

```
1 <HTML>
2 <HEAD>
3 <TITLE>INTELLICAST: europe weather</TITLE>
4 <A NAME="europe"></A>
5 <TABLE BORDER=0 CELLPADDING=0 CELLSPACING=0 WIDTH=509>
6 <TR>
7 <TD colspan=11><I>Click on a city for local forecasts</I><BR></TD>
8 </TR>
9 <TR>
10 <TD colspan=11><I> temperatures listed in degrees celsius </I><BR></TD>
11 </TR>
12 <TR>
13 <TD colspan=11><HR NOSHADE SIZE=6 WIDTH=509></TD>
14 </TR>
15 </TABLE>
16 <TABLE CELLSPACING=0 CELLPADDING=0 WIDTH=514>
17 <TR ALIGN=left>
18 <TH COLSPAN=2><BR></TH>
19 <TH COLSPAN=2><I>Tue, Jan 28, 1997</I></TH>
20 <TH COLSPAN=2><I>Wed, Jan 29, 1997</I></TH>
21 </TR>
22 <TR ALIGN=left>
23 <TH><I>country</I></TH>
24 <TH><I>city</I></TH>
25 <TH><I>forecast</I></TH>
26 <TH><I>hi/lo</I></TH>
```

```
27 <TH><I>forecast</I></TH>
28 <TH><I>hi/lo</I></TH>
29 </TR>
30 <TR ALIGN=left>
31 <TD>Austria</TD>
32 <TD><A HREF=http://www.intellicast.com/weather/vie/>Vienna</A></TD>
33 <TD>snow</TD>
34 <TD>-2/-7</TD>
35 <TD>snow</TD>
36 <TD>-2/-7</TD>
37 </TR>
38 <TR ALIGN=left>
39 <TD>Belgium</TD>
40 <TD><A HREF=http://www.intellicast.com/weather/bru/>Brussels</A></TD>
41 <TD>fog</TD>
42 <TD>2/-2</TD>
43 <TD>sleet</TD>
44 <TD>3/-1</TD>
45 </TR>
.
.
</TABLE>
</HTML>
```

Figura 7. Código HTML
Fonte: GUANDELINE, E. L. T. (2002)

Após analisar o código HTML anterior, o usuário deve definir um arquivo especifico, constituído de vários comandos [variável, fonte, modelo] mostrado anteriormente. Este arquivo irá mostrar ao extrator o arquivo do qual deseja extrair os dados, quais os dados importantes que devem ser extraídos e como estes devem ser extraídos do código HTML (GUANDELINE, 2002).

O Código da Figura 8 é um exemplo de extração do código que foi citado anteriormente. Este código pode ser visto como um pseudocódigo para a abstração das funções específicas das linguagens de programação.

```
1 [["root",
2 "get('http://www.intellicast.com/weather/europe/')",
3 "#"
4 ],
5 ["temperatures",
6 "root",
7 "*<TABLE*<TABLE*</TR>#</TABLE>*"
8 ],
9 ["_citytemp",
10 "split(temperatures,'<TR ALIGN=left>')",
11 "#"
12 ],
13 ["city_temp",
14 "_citytemp[1:0]",
15 "#"
16 ],
17
["country,c_url,city,weath_today,hgh_today,low_today,weath_tomorrow,hgh_tomor
row,low_tomorrow",
18 "city_temp",
19
"*<TD>#</TD>*HREF=#>#</A>*<TD>#</TD>*<TD>#/#</TD>*<TD>#</TD>*<TD>
#/#*"
20 ]]
```

Figura 8. Possível código de extração
Fonte: GUANDELINE, E. (2002)

Analisando-se o arquivo de especificação anterior, temos que as linhas 1, 2, 3 e 4 definem um comando na forma [variável, fonte, modelo]. Elas indicam também que a variável se chama *root*, a fonte é http://www.intellicast.com/weather/europe e a palavra reservada # mostra que o conteúdo extraído deve ser armazenado na variável *root*.

De acordo com as linhas 5 a 8, a variável *temperatures* receberá valores da variável root conforme o modelo apresentado na linha 7. Este modelo indica a seguinte ação: desconsidere tudo (*) até encontrar a primeira tabela (<TABLE), desconsidere tudo até encontrar a segunda tabela, desconsidere tudo até encontrar a primeira tag </TR>, extrair todo o conteúdo até encontrar a tag </TABLE>.

Nas linhas 9 a 12 é indicado que a variável *_citytemp* receberá todo o conteúdo da variável *split(temperatures, '<TR ALIGN=left>')*. A função *split* cria um vetor com as partes da variável *temperatures* separadas pela tag *<TR ALIGN=left>*.

Os comando incluso nas linhas 13 a 15 indica que a variável *city_temp* receberá os elementos do vetor *city_temp*.

Note que na linha 14 é indicado no primeiro número inteiro, que se deve começar o vetor pelo segundo elemento, assumindo que o vetor comece no "0", no segundo número inteiro indica que o vetor tem que ser lido por inteiro, este número consiste em um índice partindo do último elemento.

O comando das linhas de 17 a 19 são para armazenar no arquivo de especificação os valores nas variáveis definidas pelo usuário. Após os comandos, as variáveis são empacotadas em objetos OEM, esse formato pode ser visto na Figura 9.

```
root complex {
        temperature complex {
                city_temp complex {
                        country string "Austria" city_url url http://www…
                        city string "Vienna"
                        weather_today string "snow"
                        high_today string "-2"
                        low_today string "-7"
                        weather_tom string "snow"
                        high_tomorrow string "-2" low_tomorrow string "-7"
                }
                city_temp complex {
                        country string "Belgium"
                        city_url url http://www…
                        city string "Brussles"
                        …
                        }
                …
                }
}
```

Figura 9. Dados extraídos no formato OEM
Fonte: GUANDELINE, E. L. T. (2002)

Deve-se levar em consideração que podem existir diferentes modelos OEM para a mesma base de dados, e que mudanças no código HTML implicam em mudanças também no arquivo específico.

Existem ainda dois comandos que podem ser usados nesse exemplo, que é o *extract_table* e o *case.* No primeiro comando se extrai automaticamente o conteúdo de uma tabela, quanto no segundo permite que o usuário defina vários arquivos de especificação, assim o *parse* pode verificar qual modelo ele deve usar sobre o documento HTML (GUANDELINE, 2002).

3.4 BASEADO EM ONTOLOGIAS

Muitos documentos possuem relacionamentos entre seus dados e juntos descrevem o contexto da informação. Nesses casos podemos definir um modelo conceitual e extrair a semântica dos dados. Esse modelo é baseado em uma ontologia que descreve os dados de interesse, incluindo relações, aparência léxica e chaves de contexto (GUANDELINE, 2002).

Uma ontologia é utilizada para representar o conhecimento de um determinado domínio, ou seja, representam os objetos, as relações entre eles, as restrições de integridade, os aninhamentos, os tipos de dados e a cardinalidade. Para isso tem-se a necessidade de um perito no domínio de ontologia. Ela é utilizada para compartilhar informações por diversas aplicações (CALDAS, 2003).

Um banco de dados é populado a partir da extração de dados em documentos através de regras pré-estabelecidas pela ontologia. Após a extração é possível ter uma consulta mais precisa com relacionamentos existentes no domínio da aplicação (GUANDELINE, 2002).

Um modelo de extração sintática baseada nas características dos elementos a serem extraídos. Este exemplo foi definido para processar grandes quantidades de informações inter-relacionadas e podem ser definidos por uma pequena ontologia.

a) modelagem de uma ontologia sobre a área de interesse, ou seja, definir quais as relações existentes no domínio da aplicação;
b) é gerado um banco de dados para posterior população. Feito uma análise desta ontologia para gerar regras que adaptem as palavras-chave em constantes a ser usadas no esquema do banco de dados;
c) os dados da web são quebrados em pequenos fragmentos eliminando as *tags* HTML, formando assim pequenos registros desestruturados para posterior processamento;
d) um reconhecedor extrai os dados e as relações esperadas dos pequenos registros, sendo analisados conforme expressões regulares definidas anteriormente;
e) utilizando heurísticas que levam em consideração a cardinalidade descrita na ontologia é populado o banco de dados determinando a estrutura dos registros nele.

Na Figura 10 é mostrada a estrutura do processo descrito, e se percebe que a única entrada do sistema é o documento web. A ontologia é escolhida de acordo com o domínio do documento e acaba na população do banco de dados (GUANDELINE, 2002).

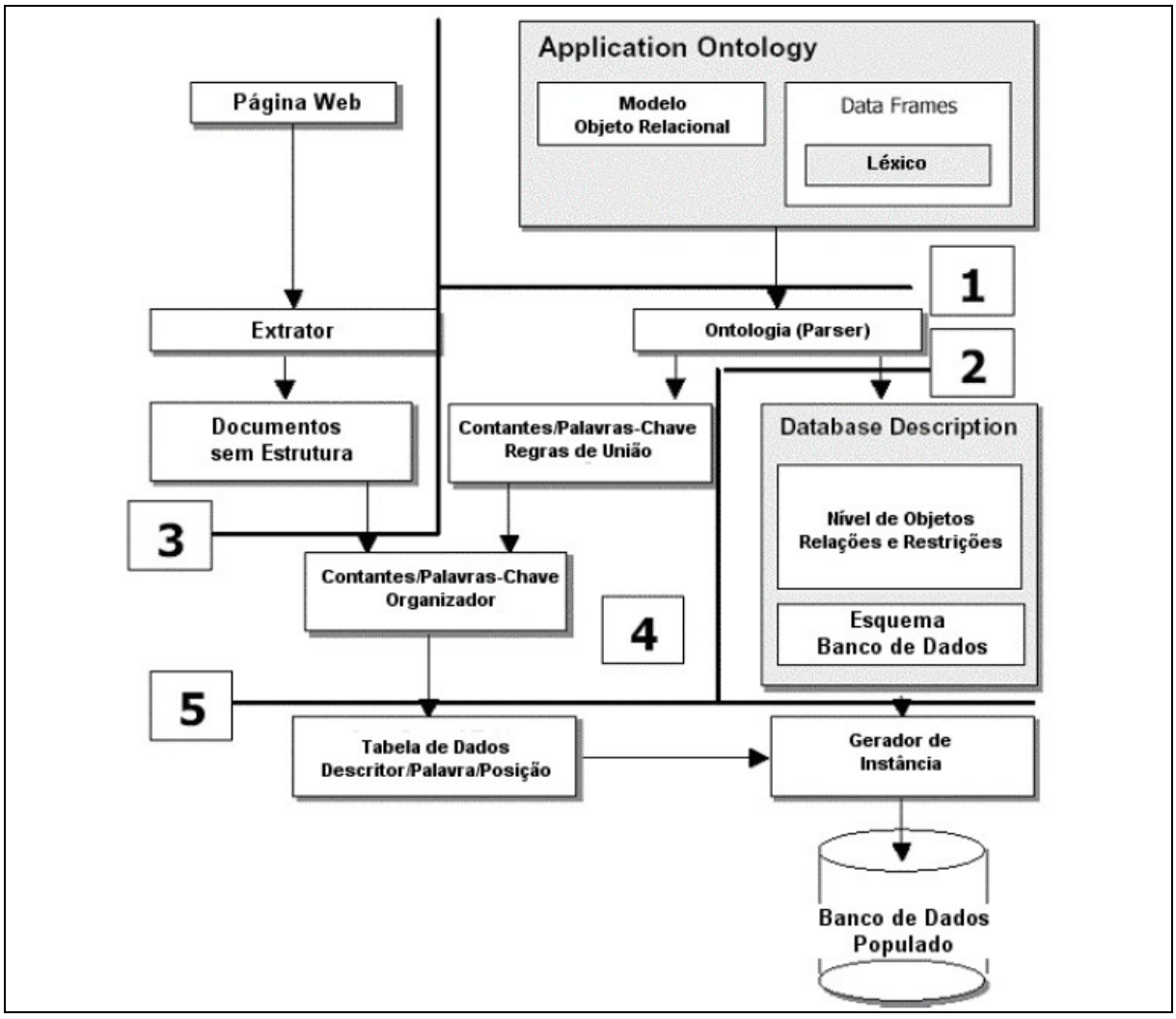


Figura 10. Modelo de extração baseado em ontologia
Fonte: GUANDELINE, E. (2002)

Será mostrado um exemplo para seguir os passos da Figura 10. Nesse exemplo em questão será tratada a fonte de dados de uma venda de automóveis. A Figura 11 apresenta um possível código dessa fonte.

```
'97 CHEV Cavalier, Red, 5 spd, only 7,000 miles on her.
Previous owner heart broken! Asking only $11,995. #1415.
JERRY SEINER MIDVALE, 566-3800
####
'94 CHEV Corsica, 88,281 miles. Ask for #16. $4,900.
Government Surplus533-5885
####
```

Figura 11. Modelo de código para venda de automóveis
Fonte: GUANDELINE, E. (2002)

São criadas regras léxicas, *DataFrame*, compatíveis com as entradas dos dados e as constantes de domínio. As informações léxicas são usadas no quadro *Aplication Ontology*

na Figura 10. Na Figura 12 é apresentado um modelo de ontologia no formato OSM (*Object Oriented Systems Model*) para venda de carros.

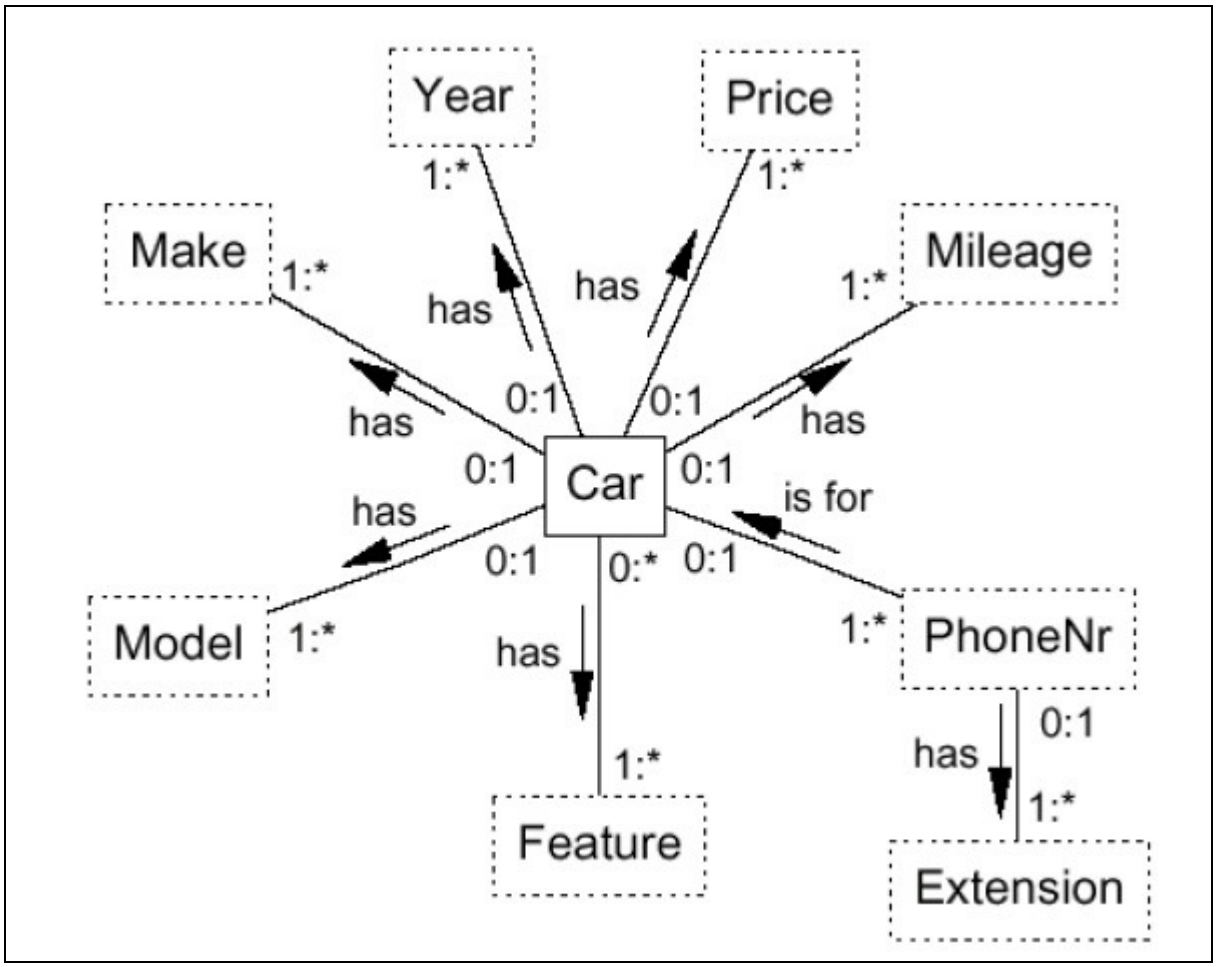


Figura 12. Modelo de ontologia para venda de carros
Fonte: GUANDELINE, E. (2002)

Na Figura 13 tem a representação textual feita da Figura 12.

```
Car [0:1] has Year [1:*];
Year {regexp[2]: "\d{2} : ([^\$\d]|^)\d{2}[^,\dkK]",
"\d{2} : ([^\$\d]|^)\d{2},[^\d]",
"\d{2} : \b'\d{2}\b" };
Car [0:1] has Make [1:*];
Make {regexp[10]: "\bchev\b", "\bchevy\b", ... };
Car [0:1] has Model [1:*];
Model {regexp[16]: "88 : \bolds\S*\s*88\b",
"80 : \baudi\S*\s*80\b", "\bacclaim\b", ... };
Car [0:1] has Mileage [1:*];
Mileage {regexp[8]: "\b[1-9]\d{1,2}k",
"[1-9]\d?,\d{3} : [^\$\d][1-9]\d?,\d{3}[^\d]" }
{context: "\bmiles\b", "\bmi\.", "\bmi\b"};
Car [0:*] has Feature [1:*];
Feature {regexp[20]:
-- Colors
"\baqua\s+metallic\b", "\bbeige\b", ...
-- Transmission
"(5|6)\s*spd\b", "auto : \bauto(\.|,)",
-- Accessories
"\broof\s+rack\b", "\bspoiler\b", ...
-- Engine characteristics
"\bv-?(6|8)", "\b6\s*cyl\b", ...
-- Body/Style
"\b4\s*d(oo)?r\b", "\b2\s*d(oo)?r\b", ...
-- Low mileage
"\blow\s+miles\b", "\blow\s+mi\.", ... };
Car [0:1] has Price [1:*];

...
```

Figura 13. Representação textual do modelo de ontologia para venda de carros
Fonte: GUANDELINE, E. (2002)

Cria-se então um banco de dados utilizando comandos DDL do SQL. A DDL da Figura 13 se observa na Figura 14.

```
create table Car (
Car integer,
Year varchar(2),
Make varchar(10),
Model varchar(16),
Mileage varchar(8),
Price varchar(8),
PhoneNr varchar(8));
create table PhoneNr (
PhoneNr varchar(8),
Extension varchar(4));

create table CarFeature (
Car integer,
Feature varchar(10));.
```

Figura 14. DDL modelo de ontologia para venda de carros
Fonte: GUANDELINE, E. (2002)

Um *DataFrame* serve para especificar palavras chaves e as expressões regulares que reconhecem objetos em um registro. Pode definir uma expressão comum, como por exemplo, representar a constante *ano* com dois ou quatro dígitos (98 ou 1998). Um exemplo de *DataFrame* pode ser visto na Figura 15.

```
Year : \d{2} : ([^\$\d]|^)\d{2}[^,\dkK]
Year : \d{2} : ([^\$\d]|^)\d{2},[^\d]
Year : \d{2} : \b'\d{2}\b
Make : \bchev\b
Make : \bchevy\b ...
Model : 88 : \bolds\S*\s*88\b
Model : 80 : \baudi\S*\s*80\b
Model : \bacclaim\b ...
Mileage : \b[1-9]\d{1,2}k
Mileage : [1-9]\d?,\d{3} : [^\$\d][1-9]\d?,\d{3}[^\d]
KEYWORD(Mileage) : \bmiles\b
KEYWORD(Mileage) : \bmi\.
KEYWORD(Mileage) : \bmi\b
Feature : \baqua\s+metallic\b
Feature : \bbeige\b ...
Feature : (5|6)\s*spd\b
Feature : auto : \bauto(\.|,)
Feature : \broof\s+rack\b
Feature : \bspoiler\b ...
Feature : \bv-?(6|8)
Feature : \b6\s*cyl\b ...
Feature : \b4\s*d(oo)?r\b
Feature : \b2\s*d(oo)?r\b ...
Feature : \blow\s+miles\b
Feature : \blow\s+mi\. ...
Price : [1-9]\d?,\d{3} : \$[1-9]\d?,\d{3}
Price : [1-9]\d{2,3} : \$[1-9]\d{2,3}
PhoneNr : [1-9]\d{2}-\d{4} : (\b|[^\d])[1-9]\d{2}-\d{4}([^\d]|$)
KEYWORD(Extension) : \bext\b
Extension : \d{1,4} : \d{1,4} : (x|ext\.\s+)\d{1,4}\b
```

Figura 15. Exemplo de *DataFrame*
Fonte: GUANDELINE, E. (2002)

O segundo passo se refere a extração dos dados do arquivo HTML, que se divide em dois passos:

a) localizar as informações referentes a ontologia, sem levar em consideração que as informações podem estar em documentos separados;

b) dividir os documentos em pequenos fragmentos e eliminar as *tags* do HTML.

Cada registro é analisado pelo quadro *Constant/Keyword/Recognizer* que gera uma tabela descritor, string, posição. Veja o exemplo da Figura 16.

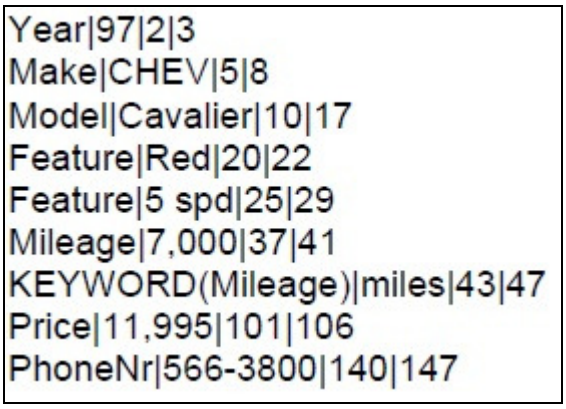


Figura 16. Descritor, string, posição
Fonte: GUANDELINE, E. (2002)

A palavra reservada "KEYWORD(X) \ variável \ posição inicial \ posição final" mostra que é uma expressão regular e que o *DataFrame* tem regras pré-estabelecidas para formatá-la.

Depois do último processo, o quadro *Database-Instance Generator* insere os elementos em um banco de dados, podendo ser feito consultas através de comandos SQL.

3.5 BASEADA EM PROCESSAMENTO DE LINGUAGEM NATURAL (*NLP-based*)

O Processamento de Linguagem Natural (PLN) é usado por diversas ferramentas para aprender regras de extração a fim de extrair informações úteis em documentos. Estas regras são baseadas em restrições sintáticas e semânticas, aplicando técnicas para relacionar frases e elementos da oração (CALDAS, 2003).

PLN consiste no desenvolvimento de métodos computacionais para a análise e manipulação ou codificação de informações expressas em língua natural. O objetivo é compreender a língua, por exemplo, sumarizando uma grande quantidade de texto ficará muito melhor de utilizá-lo de forma relevante. Algumas das principais tarefas de PLN são reconhecimento de contexto, análise sintática, semântica, léxica e morfológica, sumarização e tradução de textos (ÁLVARES, 2007).

As técnicas de PLN são amplamente utilizadas em documentos com pouca ou nenhuma estruturação. O objetivo é compreender textos em alguma língua natural, a fim de encontrar dados que sejam relevantes para extraí-los (MANFREDINI, 2003).

Em sistemas de Extração de Informação são identificados seis módulos principais baseados em PLN: *tokenizador*, processador léxico e/ou morfológico, analisador sintático/semântico, um módulo onde são definidos os padrões de extração, analisador do discurso e o último módulo de preenchimento de *templates* (MANFREDINI, 2003).

A fase de tokenização é feito para separar as palavras do texto, de modo a identificar os constituintes da sentença. Na maioria dos casos, esta tarefa é feita através do reconhecimento dos espaços em branco ou pelos sinais de pontuação (MANFREDINI, 2003).

Após a separação das palavras, é feito uma análise léxica e/ou morfológica de cada uma, classificando-as segundo sua classe morfológica (substantivo, verbo, artigo, etc.) e demais características (feminino, plural, etc.). Este módulo é responsável pelo reconhecimento de nomes próprios e outros itens que podem ter uma estrutura interna (hora, data, etc.) (MANFREDINI, 2003).

O módulo de análise sintática e semântica é responsável por receber os itens léxicos em seqüência e tentar construir uma estrutura sintática para cada sentença do texto. São associadas características para todos os grupos nominais e verbais reconhecidos, para serem combinadas nas fases seguintes com os padrões de extração. Por exemplo, nos grupos nominais, pode-se incluir informações sobre a raiz do núcleo, para identificar se ele corresponde a um nome próprio ou não (MANFREDINI, 2003).

A construção de regras ou padrões de extração consiste na criação de um dicionário de regras de extração para um determinado domínio. Isto permite que sejam extraídos apenas os dados de interesse (MANFREDINI, 2003).

A etapa de análise dos discursos consiste, ao contrario do que foi tratado nas anteriores, em relacionar diferentes elementos do texto, relacionando as sentenças. Este módulo possui alguns tratamentos de problemas como:

a) analise de frases nominais que se refere à tarefa de reconhecer e interpretar apostos e outros grupos nominais complexos;

b) resolução de correferência, que trata o problema de identificar quando uma nova frase nominal (normalmente, um pronome) se refere à outra já citada anteriormente;

c) descoberta de relacionamento entre as partes do texto, que objetiva estruturar as palavras do texto em uma rede associativa, fornecendo suporte à tarefa de extração.

O módulo de preenchimento de *templates* (que são frases com lacunas a serem preenchidas pelas informações extraídas) recebe os dados dos estágios anteriores e preenche os *templates* definidos pela aplicação (MANFREDINI, 2003).

3.6 TÉCNICAS ESTATÍSTICAS DE APRENDIZADO DE MAQUINA

A área de aprendizado de máquina estuda construir programas que possam aprender e melhorar seus desempenhos a partir de experiências. Então quando seu desempenho aumenta em relação a determinadas tarefas ele aprende com essa experiência (OLIVEIRA, 2009).

No Aprendizado de Maquina é aplicado indução sobre um conjunto particular de exemplos (experiências) para se obter conclusões genéricas. Existem dois tipos principais de aprendizado indutivo: supervisionado e não-supervisionado (OLIVEIRA, 2009).

No supervisionado, ou com professor, é feito por meio de exemplos rotulados no formato entrada/saída desejada. O algoritmo do Aprendizado de Maquina gera um modelo correto para rotular novas entradas (OLIVEIRA, 2009).

No aprendizado não-supervisionado os exemplos rotulados não existem. O algoritmo deve, através de uma medida de qualidade, rotular os dados de entrada por grupos e conforme sua representação (OLIVEIRA, 2009).

Estas técnicas estão sendo utilizadas especialmente para modelo de uma estrutura a partir de dados e como usar da melhor forma dados rotulados e não rotulados. Um exemplo dessa técnica é o *Hidden Markov Models*, são sistemas determinísticos de aprendizado de regras que extraem informações de textos não estruturados e transformam em um registro estruturado (ÁLVARES, 2007).

As técnicas para extração de informação geralmente analisam termos simples, ou seja, palavras individuais. Nessa técnica são utilizados modelos estatísticos, como por exemplo, a freqüência em que uma palavra é encontrada em um texto, salientando que pode ser usado uma ou mais palavras (SCARINCI, 1997).

É atribuído um peso para cada palavra conforme a sua freqüência, e feito uma comparação entre as palavras. As palavras de maior peso são as palavras-chave, e desse peso depende a eficiência do resultado (SCARINCI, 1997).

Normalmente se identifica um número limite de termos ou uma porcentagem de termos conforme o número total de termos existentes no texto. Existem algumas variações do processo estatístico, porém as características principais são as mesmas (SCARINCI, 1997).

É uma técnica fácil de aplicar e razoavelmente efetiva. Contudo, ela não faz relação entre as palavras, não promovendo assim uma relação intrínseca das palavras com o documento num todo, não identificando o conteúdo completo do documento (SCARINCI, 1997).

3.7 TÉCNICAS BASEADAS EM MODELO

Ferramentas que usam dessa técnica utilizam objetos criados por usuários que servem como modelos. A técnica procura por porções de dados na web que se compare com o modelo pré-definido e o ajusta na estrutura (CALDAS, 2003).

Essa técnica se baseia na aquisição de dados na web que se assemelham a estrutura de objetos de interesse, ou seja, um padrão de objeto que deve ser procurado em uma página (CARDOSO, 2004).

A estrutura é provida conforme um conjunto de primitivas de modelagem (como tuplas, por exemplo) que estão de acordo com um modelo de dados. Depois são aplicados algoritmos que procuram por objetos que estão de acordo com a estrutura padrão definida (CARDOSO, 2004).

3.7.1 NoDoSe

Northwestern Document Structure Extractor (NoDoSe), é uma ferramenta com interface gráfica, que permite aos usuários, decompor hierarquicamente o documento, e mostrando os pontos “interessantes” e descrevendo suas semânticas.

O processo de decomposição acontece em níveis, onde o usuário cria objetos complexos em cada nível em outros objetos de estrutura simples. Após “ensinar” o NoDoSe, o usuário pode deixá-lo identificar outros objetos no documento. Isso é feito por meio de um componente de mineração que aborda a gramática por meio dos objetos criados pelo usuário (CARDOSO, 2004).

4 EXTRAÇÃO E MANIPULAÇÃO DE DADOS NÃO ESTRUTURADOS

A Recuperação de Informação (RI) é uma área da computação que tem como objetivo recuperar páginas webs, que são livres de qualquer formatação, e indexá-las de acordo com ocorrência de palavras-chave de modo estatístico. O resultado passa ainda por uma triagem do usuário que seleciona as páginas de maior interesse (SILVA, 2003).

Segundo Araujo (2007), a RI é uma área que se dedica a pesquisa de tecnologias para manipulação e recuperação, ou seja, através da representação, armazenamento e da organização permitem ao usuário um fácil acesso as grandes coleções de informações em diferentes formatos de apresentação.

Define-se RI como um processo de busca em documentos relevantes em resposta a uma consulta que favoreça as necessidades de informação de um usuário, por exemplo, usando métodos estatísticos para o cálculo de similaridade entre dois documentos (ÁLVARES, 2007).

Para localizar informações, a RI utiliza de indexação, efetuado uma busca mais rápida e eficiente. Essa indexação é considerada um tipo de filtro, capaz de identificar e selecionar características de um documento, utilizando os termos mais relevantes e excluindo os menos significativos (SILVA, 2002).

A indexação pode ser feita de três formas: Tradicional, *full-text* e por parte do texto (*tags*). Na tradicional os termos que farão parte do índice são selecionados manualmente, no *full-text* os termos são usados como parte do índice e no de *tags* a seleção de termos é feito de forma automática (SILVA, 2002).

A RI, no entanto, apresenta alguns problemas abaixo:

a) palavras sinônimas: palavras que expressam o mesmo significado. Por exemplo, *data de notificação* e *data de aceitação* em eventos científicos possuem o mesmo significado;

b) polissemia: palavra que possui mais de um significado de acordo com seu contexto. Por exemplo, *látex* pode se referir a matéria prima da borracha ou ao software de geração de textos;

c) frase: algumas palavras em conjunto formam uma frase, porém se analisadas separadamente apresentam sentidos diferentes. Por exemplo, a expressão *submissão de artigos* se analisadas isoladamente podem mudar o contexto;

d) contexto local: a relevância de uma palavra varia de acordo com o contexto abordado. Por exemplo, numa busca sobre *roubos de bancos*, a palavra *roubos* é muito relevante para saber o que se procura sobre *bancos*.

e) contexto global: alguns documentos não contem palavras ou frases boas para indexação, o documento pode ser relevante dependendo do seu contexto como um todo. Por exemplo, na frase "*um homem armado tomou o dinheiro e fugiu"*, as palavras isoladas não demonstram o contexto, porém com todas claramente se descreve um roubo.

Com os problemas citados acima, o usuário acaba perdendo na precisão de busca. Resultados ruidosos, ou seja, muitas paginas que não condizem com o interesse do usuário, páginas perdidas na busca por não serem indexadas corretamente ou por nem serem indexadas (SILVA, 2003).

```
'97 CHEV Cavalier, Red, 5 spd, only 7,000 miles on her.
Previous owner heart broken!  Asking only $ 11,995.  # 1415.
JERRY SEINER MIDVALE, 566-38006-3800
```

Figura 17. Trecho de um anúncio de venda de carros
Fonte: SILVA, T. D. M. S. (2003)

No trecho da Figura 17 mostra um exemplo que seria indexado de forma errada, pois por meio de suas palavras isoladas não seria possível referenciar seu contexto (SILVA, 2003).

Apesar da RI obter alguns avanços significativos através de algumas técnicas desenvolvidas, ainda não se obtém contexto às páginas indexadas. O uso de palavras-chave, suas combinações e análises estatísticas ainda não são suficientes para um sistema RI recuperar e indexar páginas de acordo com seus contextos (SILVA, 2003).

Ainda que a web seja rica e grande em relação a documentos, ela também é bastante confusa. Assim para amenizar essa confusão a RI desempenha um papel importante para a recuperação de documentos relevantes, porém frustram as expectativas dos usuários, grande parte das vezes, em pesquisas na web, retornando informações fora do contexto (CARDOSO, 2004).

Nesse trabalho é visada a contextualização da extração em páginas HMTL com o assunto de interesse, conforme já fundamentado é um dos problemas relacionados a RI e através de pesquisas em métodos de contextualização de termos e somando a um método de busca já idealizado é que se pretende chegar ao desenvolvimento final.

4.1 LINGUAGEM HMTL

A *HyperText Markup Language* (HTML) é a linguagem usada para preparar documentos Web. A HTML marca textos como títulos, parágrafos, listas, inclui imagens em documentos, inclui formulários de preenchimento, entre outros, por meio de comandos chamados *tags* ou *elementos*. Localiza também vínculos de hipertexto por meio da URL que fica incluído nas instruções HTML. Esses hipertextos podem apontar para outras imagens, vídeos, arquivos de áudio, e etc. em qualquer lugar da internet (GRAHAM, 1998).

HTML é a linguagem de formatação das páginas publicadas na Web ou em uma intranet. Não é uma linguagem de programação, mas somente de marcação de documentos a partir do qual toda formatação é feita. Simplesmente pode ser comparado a um processador de textos porem sem nenhum botão para formatar o texto. Para colocar em negrito, por exemplo, ao invés de usar um botão teria que colocar uma marca especial para indicar que um trecho esta em negrito. A cada uma dessas marcas se dá o nome de *tag* ou *elemento* (CARVALHO, 2001).

Na Figura 18 é mostrado um trecho de código HTML para exemplificação.

```
<! DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<! -- saved from url=(0040)http://parintins.net/shopping/index.php3 -->
<HTML><HEAD><TITLE>Parintins Shopping - Cultura da Amazônia!</TITLE>
<META content="text/html; charset=iso-8859-1" http-equiv=Content-Type>
<STYLE type=text/css>A:hover {COLOR: #000000}
                     A:hover FONT {COLOR: #ff6600}
                     A:link FONT {COLOR: #000000}
                     A {TEXT-DECORATION: none}
                     A:hover {TEXT-DECORATION: underline}
                       .input {BACKGROUND: #ffffcc;
                               COLOR: #ff0000;
                               FONT-WEIGHT: normal}
</STYLE>
<META content="MSHTML 5.00.2614.3500" name=GENERATOR></HEAD>
<BODY  aLink=#00ff00 background=index_arquivos/bg_br.gif bgColor=#ffffff
       leftMargin=0 link=#006600 topMargin=0 vLink=#006600
       marginheight="0">
<TABLE bgColor=#00ff00 border=0 cellPadding=0 cellSpacing=0 width=780>
 <TBODY>
 <TR>
  <TD><FONT color=#000000 face="Verdana, Arial, Geneva"
      size=1><B>  <A href="http://parintins.net
      /shopping/index.php3?
```

Figura 18. Exemplo de Código HTML
Fonte: CALDAS, P. (2003)

4.2 MÉTODOS DE CONTEXTUALIZAÇÃO

Marangon (2006) descreve alguns métodos de contextualização, num primeiro momento classifica de acordo com a seleção de páginas:

a) filtro de seleção: baseia-se em um critério de seleção para um conjunto de páginas. Por exemplo, selecionar apenas páginas com domínio *.br*;
b) filtro de descarte: baseia-se em um critério de descarte de páginas. Por exemplo, descarta todas as páginas do domínio *.br*;
c) ponderação: todas as páginas são selecionadas para uma avaliação posterior, porém ordenadas por peso, por exemplo, de relevância;
d) agrupamento: as páginas podem ser selecionadas individualmente ou em grupo, conforme sua categoria, diretório ou domínio.

Após essa classificação, Marangon (2006), aborda ainda alguns desses métodos para contextualização de páginas web:

a) consulta por termos;
b) contextualização pela URL;
c) contextualização por referência direta.
d) contextualização por agrupamento por referência direta;
e) contextualização por agrupamento das páginas, analisando a estrutura de hyperlinks;
f) contextualização pelo agrupamento das páginas, utilizando similaridade do texto;
g) contextualização pela categorização da página;

4.2.1 Consulta por Termos

É a forma básica de consulta fornecida pelos mecanismos de busca, buscam por um ou mais termos, utilizando de várias formas de consulta: consulta booleana, consulta por proximidade, etc (MARANGON, 2006).

4.2.2 Contextualização pela URL

A utilização da URL para o contexto de uma página é a analise das informações contidas no esquema de localização e acesso contidas dentro da URL. O esquema geral de uma URL é apresentado na Figura 19.

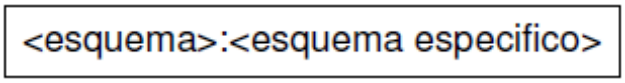


Figura 19. Esquema geral de uma URL
Fonte: MARANGON, S. (2006)

Sendo:

a) **esquema** – um protocolo como: ftp, http, gopher, etc.;

b) **esquema específico** – a parte específica do esquema e depende do esquema escolhido.

A sintaxe para o esquema específico de um protocolo HTTP é definida como na Figura 20.

Figura 20. Sintaxe do esquema para HTTP
Fonte: MARANGON, S. L. (2006)

Sendo:

a) **host_domain:** o nome DNS do *host*;

b) **port:** o número de uma porta TCP/UDP;

c) **path:** o caminho para um arquivo;

d) **search_part:** parâmetros extras dependentes do protocolo.

Existem três tipos de análise da URL que podem ser feita para sua contextualização: análise do domínio, análise do caminho de diretórios e parâmetros extras e análise de padrão.

A análise de domínio pode ser realizada por: código de país, domínio genérico e termos específicos. Código de país filtra páginas pela origem. Pelo domínio pode ser contextualizado pela categoria de primeiro nível, por exemplo, instituições de comunicação (.INFO.BR), radio difusão (.TV.BR), empresas comerciais (.COM.BR). As mesmas heurísticas e técnicas de procura por um termo pode ser usada para análise de uma URL (MARANGON, 2006).

4.2.3 Contextualização por Referência Direta

Os *hyperlinks* das páginas webs formam um grafo com sentido único, porém se usados em sentido invertido poderia ser muito utilizado, por exemplo, para *clipping* de notícias para uma determinada empresa, analisando páginas que fazem referência à empresa (MARANGON, 2006).

4.2.4 Contextualização pelo Agrupamento de Páginas

Através do agrupamento de páginas é selecionado páginas que tenham relacionamentos entre si. Agrupamentos que podem ser feito, por exemplo, através de métodos da análise de estrutura de *hyperlinks*, baseado na similaridade do texto, relacionado assim o contexto dos grupos da qual fazem parte (MARANGON, 2006).

4.2.5 Agrupamento por Referência Direta

Esse método utiliza a idéia de contextualização por referência direta e o agrupamento pela análise da estrutura de *hyperlinks* de maneira mais simplificada. É usada apenas uma página escolhida para agrupamento de páginas relacionadas a ela. É feito também uma limitação de páginas relacionadas para agrupar, sendo o tamanho do conjunto baixo, deixando apenas páginas estritamente relacionadas (MARANGON, 2006).

4.2.6 Contextualização pela Categorização da Página

A categorização da página é utilizada quando se deseja selecionar páginas de um assunto específico em um conjunto de páginas com assuntos variados. Por exemplo, no caso de um serviço de *clipping*, como apresentado anteriormente, poderia usar a contextualização pela URL e poderia identificar uma grande quantidade de páginas de notícias, porém não conseguiria identificar o assunto delas. Sendo assim a categorização seria uma boa sequencia para a contextualização pela URL (MARANGON, 2006).

4.2.7 Etiquetagem Morfossintática de Textos

A etiquetagem morfossintática de um texto implica em utilizar de etiqueta ou rótulo (*tag*), com etiquetas já predefinidas (*tagset*), para definir palavras, símbolos de pontuação, palavra estrangeira, ou fórmula matemática presente em um texto, conforme o seu contexto (ÁLVARES, 2007).

A Figura 21 apresenta um exemplo de etiquetagem, e uma possibilidade de etiqueta diferente a ser usada (PTO) para designar final de uma sentença.

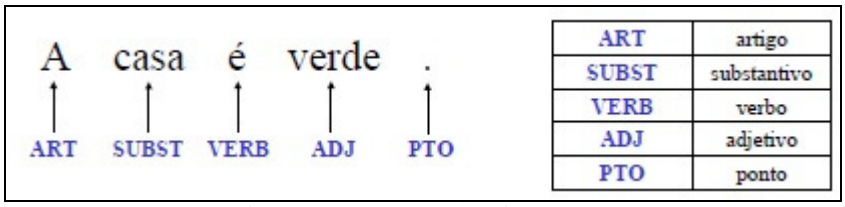


Figura 21. Exemplo de etiquetagem
Fonte: ÁLVARES, A. (2007)

As etiquetas por sua vez podem ser refinadas com atributos referentes a sua classe. No caso da língua portuguesa, por exemplo, o atributo substantivo pode ser refinado com relação ao tipo (comum, próprio), grau (aumentativo, diminutivo), gênero (feminino, masculino), e assim por diante. Pode-se até criar uma etiqueta específica para uma única palavra (ÁLVARES, 2007).

O etiquetador pode ser induzido automaticamente, usando de regras que analisam as palavras e/ou seu contexto (análise das palavras ou etiquetas que vem antes e depois da palavra em foco analisada) (ÁLVARES, 2007).

Segundo Álvares (2007) a tarefa do etiquetador pode ser dividida em três módulos: escrutinador léxico, classificador gramatical e desambiguizador. O escrutinador léxico é responsável por separar (*tokenization*) e identificar as orações no texto. O classificador gramatical classifica os termos do texto com o auxilio, geralmente, de um léxico. O desambiguizador utiliza informações do contexto para determinar a etiqueta de termos que podem pertencer a mais de uma classe gramatical (ÁLVARES, 2007).

Os etiquetadores podem ser classificados em três tipos: simbólica, probabilística e híbrida. Na simbólica os etiquetadores se baseiam em regras, restrições e árvores de decisões não probabilísticas. Na abordagem probabilística, são fundamentadas através de técnicas como redes neurais, máxima entropia, Modelo de Markov e árvores de decisões probabilísticas. Os híbridos utilizam da simbólica e probabilística no processo de etiquetagem (ÁLVARES, 2007).

4.2.8 Modelo Contextual

Loh (2001) indica em seu trabalho um modelo contextual que também visa a contextualização para a representação de um conceito. É utilizada uma lista de termos chamados positivos e negativos para comparação. Para se considerar o conceito, todos os termos positivos devem estar presentes e nenhum termo negativo pode aparecer. Por exemplo, no domínio medico, o conceito "alcoolismo" pode ser representado pela regra:

1. álcool –nega;
2. hálito etílico.

Sendo que o símbolo "-" indica um termo negativo. O "-nega" serve para eliminar frases do tipo "o paciente nega uso de álcool".

4.2.9 Modelo Espaço-Vetorial

O conceito é definido com um conjunto de palavras que devem ser encontradas dentro de documentos. Esse conjunto é representado por um vetor de termos, que tem um grau de afinidade com o conceito. Os que têm um grau de afinidade mais forte são indicadores do conceito. Os que possuem um grau mais fraco precisam passar por uma função de ativação mostrada na Figura 22, que relaciona outros termos com esse para fazer verificação (WIVES, 2004).

$$ga(C, D) = \frac{\sum_{i=1}^{k} gs_i(tc_i, td_i)}{n}$$

Figura 22. Função de ativação
Fonte: WIVES, L. (2004)

Onde:

a) *ga*: é o grau de relação entre o conceito *C* e o documento *D*;

b) *i*: é um índice para os termos comuns ao conceito e ao documento;

c) *k*: é o número total de termos comuns ao conceito e ao documento;

d) *n*: é o número total de termos (sem contagem repetida);

e) *gs*: é uma função que calcula o grau de similaridade entre os pesos do *i-ésimo* termo;

f) *tci*: é o peso do *i-ésimo* termo no conceito *C e*;

g) *tdi*: é o *i-ésimo* peso do termo no documento *D*.

O grau de similaridade *gs* é dado pela formula mostrada na Figura 23.

$$gs(tc,\ td) = \frac{1}{2}\left(\frac{tc}{td} + \frac{1-td}{1-tc}\right) \quad \text{se tc <= td, ou}$$

$$gs(tc,\ td) = gs(td, tc) \quad \text{se tc > td}$$

Figura 23. Formula do grau de similaridade
Fonte: WIVES, L. (2004)

A função apresentada na Figura 22 tem que ser considerada, pois os termos podem ter graus de importância diferentes no conceito e no documento. Pesos diferentes, o grau se torna baixo, pesos iguais se torna alto e se o termo não aparece em nenhum dos dois documentos, sendo comparados, o grau é zero.

Segundo Rodrigues (2008) nesse modelo são apresentadas algumas desvantagens como, por exemplo, que não há consenso sobre os pesos que devem ser aplicados nos componentes, não dá para incluir dependências entre os termos para considerar frases ou termos adjacentes, entre outros.

Em suma o trabalho após utilizar de algumas técnicas para contextualizar uma busca, pretende utilizar de técnicas para extração de informações no resultado da busca para extrair informações relevantes e armazená-las em um BDOR a fim de usar tais informações em buscas simples para se obter um grau elevado de precisão na busca.

5 FERRAMENTA TREETAGGER

Para realizar a tarefa de etiquetagem morfossintática de textos, em inglês chamado de *part-of-speech* (POS), foi utilizada uma ferramenta chamada TreeTagger que está disponível gratuitamente na web.

Segundo Schmid (1996) a ferramenta *TreeTagger* utiliza de técnicas de PLN para fazer a etiquetagem morfossintática do texto. A etiquetagem utiliza de árvores de decisão para tratar as probabilidades de transição, um problema que o modelo de Markov enfrenta quando tem que estimar pequenas quantidade de dados. Baseado nesse método o *TreeTagger* alcançou 96,36% de precisão.

Os métodos probabilísticos usados pelo modelo de Markov são baseados em técnicas chamadas de primeira ordem e segunda ordem. Por causa dos inúmeros parâmetros, esses métodos têm dificuldades em estimar pequenas probabilidades com precisão. O *TreeTagger* apresenta uma técnica baseada em árvores de decisão que resolve esse problema.

Os possíveis contextos não são apenas trigramas, bigramas e unigramas, mas também contexto usando a árvore de decisão binária para estimar as probabilidades de transição, a probabilidade de um trigrama, por exemplo, é determinado pela sequencia do caminho correspondente, como mostra a Figura 24.

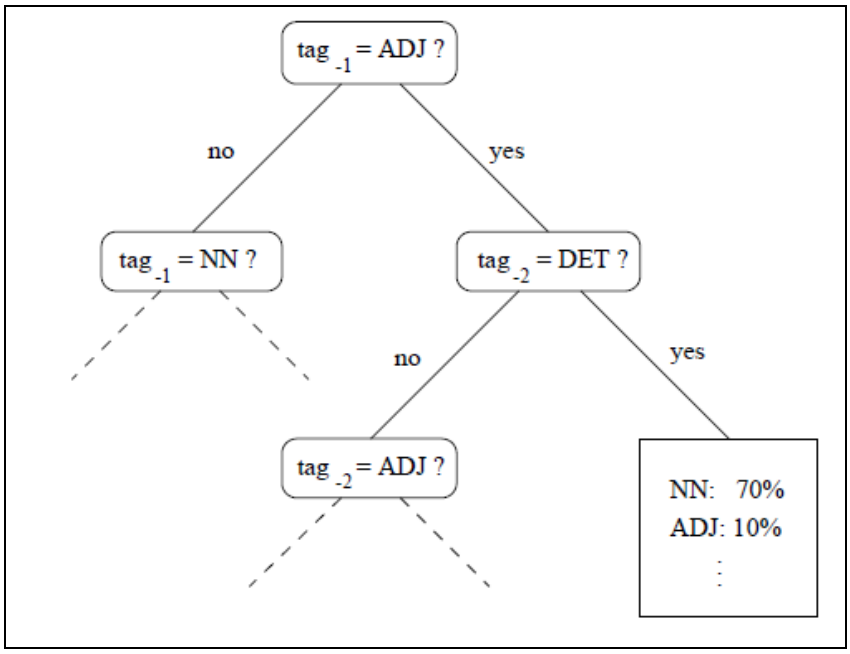


Figura 24. Uma simples árvore de decisão
Fonte: SCHMID, H. (1996)

O *TreeTagger* possui também um léxico com as probabilidades de *tags* primárias para cada palavra. O léxico possui três partes: *fullform lexicon*, *suffix lexicon* e *default entry*. O *fullform lexicon* é a palavra armazenada na sua forma primária. É o primeiro a ser procurado, se a palavra for encontrada a *tag* correspondente é retornada. Se a palavra não é achada entra na segunda parte, o *suffix léxico* que faz a verificação dos sufixos das palavras, mostrado na Figura 25. O *default entry* é utilizado caso o *suffix lexicon* falhar. São verificados os sufixos mais freqüentes e devolvido a *tag* mais semelhante.

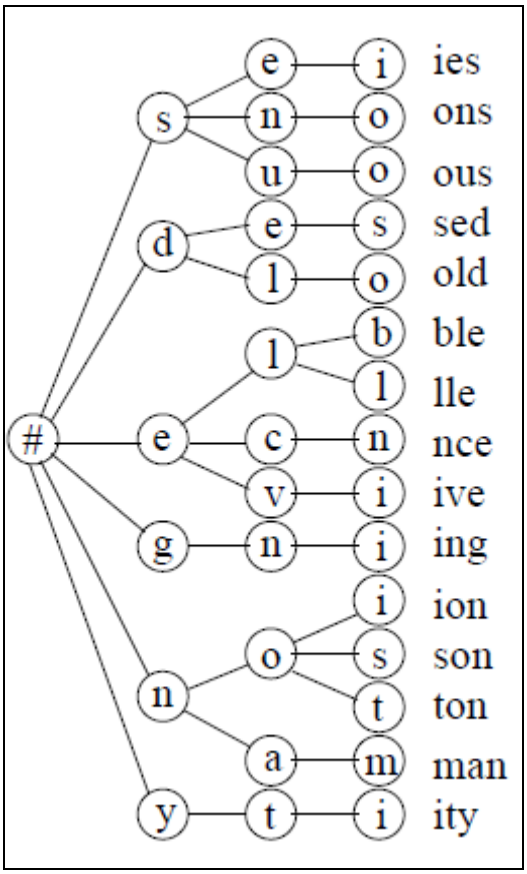


Figura 25. Uma simples árvore de sufixo de tamanho 3
Fonte: SCHMID, H. (1996)

A ferramenta então faz a classificação gramatical de cada palavra no texto, usando o espaço entre elas para defini-las. Segue abaixo as etiquetas definidas pelo autor da implementação da língua portuguesa na ferramenta:

a) ADJ: adjetivo;

b) ADV: advérbio;

c) DET: determinante;

d) CARD: número cardinal/ordinal;

e) NOM: nome comum/próprio;

f) P: pronome;

g) PREP: preposição;

h) V: verbo;

i) I: interjeição;

j) VIRG: separadores dentro da oração;

k) SENT: separadores de orações.

Existem também combinações: PREP+DET, por exemplo, nas palavras “do”, “das”, etc.; V+P, por exemplo, nas palavras “levou-me”, “disse-lhe”, etc.

A ferramenta também já classifica gramaticalmente conforme o contexto da frase, por exemplo, a frase “Eu prego as boas atitudes.” Será classificada da seguinte maneira:

a) Eu: P (pronome);

b) prego: V (verbo);

c) as: DET (determinante);

d) boas: ADJ (adjetivo);

e) atitudes: NOM (nome comum);

f) .: SENT (separador de oração).

Enquanto que a frase “Esse prego é grande” será classificada assim:

a) Esse: DET (determinante);

b) prego: NOM (nome comum);

c) é: V (verbo);

d) grande: ADJ (adjetivo);

e) .: SENT (separador de oração).

Observa-se que a palavra “prego” foi classificada de maneira diferente nas duas frases conforme sua contextualização. Na primeira frase, “prego” vem do verbo “pregar” e na segunda “prego” vem do substantivo.

5.1 UTILIZAÇÃO DA FERRAMENTA

A documentação da ferramenta está toda em inglês e devido a isso a explicação de seu funcionamento está voltada para a gramática da língua inglesa, porém as técnicas apresentadas são as mesmas para qualquer outra língua.

O executável para abrir a ferramenta encontra-se na pasta *bin*, e chama-se *wintreetagger.* A tela inicial mesmo após a colocação dos arquivos necessários, não apresenta a opção para a língua no menu *Language*, é necessário escrever "*pt"* na caixa de texto em branco no mesmo menu, como mostra a Figura 26.

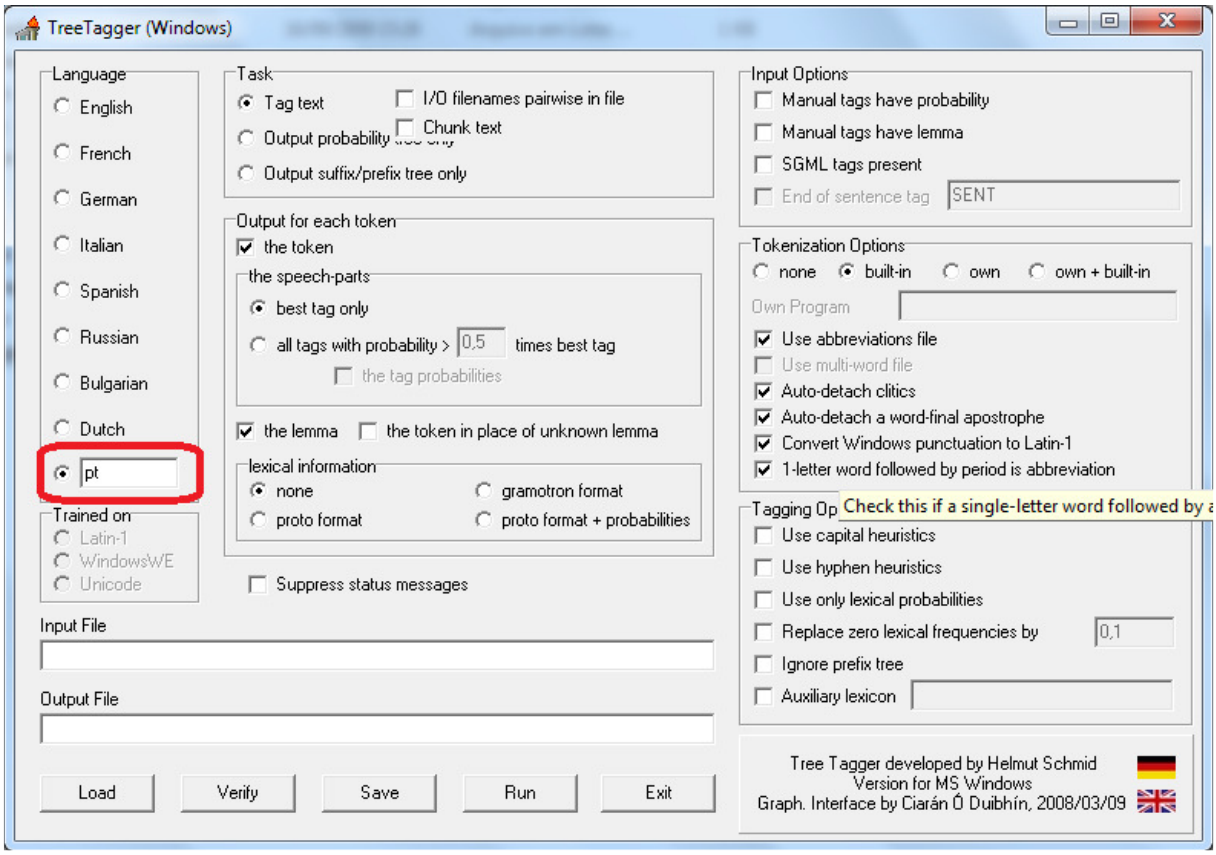


Figura 26. Selecionar língua portuguesa no *TreeTagger*

Depois clicar dentro da caixa de texto em baixo de *Input File* para selecionar o arquivo de entrada a ser etiquetado (arquivo de texto *.txt). Clicar também dentro da caixa de texto em baixo de *Outpu File* para criar um arquivo de saída onde a ferramenta irá colocar os resultados, conforme mostra a Figura 27.

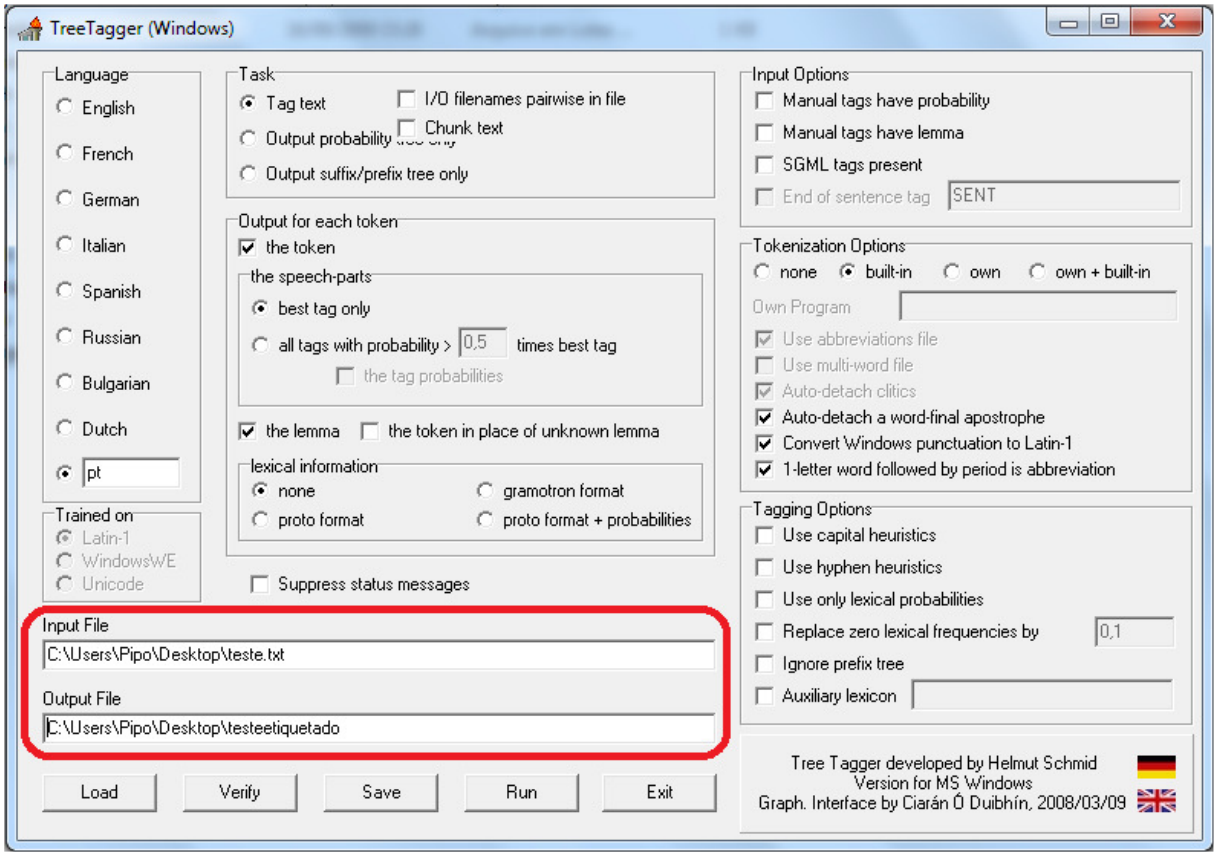


Figura 27. Seleção dos arquivos de entrada e saída no *TreeTagger*

Feita as seleções dos arquivos de entrada e saída, basta clicar no botão *Run* e o arquivo etiquetado estará no caminho indicado no campo especificado em *Output File*.

É possível verificar a árvore de probabilidades da versão em português criada pelo autor Pablo Gamallo, selecionando no menu *Task* a opção *Output probability tree only,* e selecionando apenas um arquivo de saída em *Output File.* No mesmo menu escolhendo a opção *Output suffix/prefix tree only* e selecionando um arquivo de saída é possível verificar a árvore dos sufixos e prefixos existentes da versão também.

Outra opção da ferramenta interessante de ser usada é no menu *Output for each token* no submenu *lexical information* que contem as opções: *none, proto format, gramotron format* e *proto format + probabilities.*

A opção *none* é padrão e não traz informações adicionais do léxico. Na opção *proto format* traz informações das possíveis etiquetas que foram escolhidas no processo de etiquetagem. A *gramatron format* mostra o mesmo resultado de *proto format,* porém em um

formato diferente. Em *proto format + probabilities* é o resultado de *proto format*, porém com a porcentagem da probabilidade junto.

No menu *Output for each token* tem ainda a opção *the lemma* que por padrão já vem selecionada e mostra a forma primária da palavra etiquetada. No arquivo de saída ela é identificada após a etiqueta em cada linha.

As demais opções da ferramenta são irrelevantes ou não funcionam na versão da língua portuguesa, acredita-se que por opção do autor da versão.

6 TRABALHOS CORRELATOS

As áreas de RI e EI são amplamente utilizadas para obter informações em grandes bases de dados onde se tenha alguma estrutura ou até nenhuma. E neste domínio de estudo destacam-se diversas aplicações desenvolvidas pelas técnicas baseadas em wrappers, ontologia, distância de edição em árvores, no caso da EI e técnicas da RI como, por exemplo, modelo booleano, modelo contextual, modelo probabilístico, entre outras.

Existem bastantes trabalhos desenvolvidos nas áreas de RI e EI em monografias, teses, artigos, entre outras. Porém em diversos deles, especificamente na área de RI, se relata a dificuldade com que se tem em contextualizar a busca em documentos e colocam algumas alternativas, sendo o foco desse trabalho aprofundar a contextualização em buscas, em vista disso são descritos alguns trabalhos correlatos nesse tema.

6.1 MODELOS PARA CONTEXTUALIZAÇÃO

Foram pesquisados alguns métodos de contextualização, e apenas o trabalho a seguir estuda a contextualização como foco principal.

6.1.1 Análise de Métodos para Programação de Contextualização

Essa Dissertação de Mestrado do Curso de Sistemas Eletrônicos da Engenharia na Universidade de São Paulo (MARANGON, 2006).

Tem como objetivo analisar os métodos referentes a RI e mineração na Web para utilizar na contextualização de páginas na Web e como tais métodos podem ser combinados para um processo de programação de contextualização de páginas na Web. Os métodos foram

aplicados em um mecanismo de busca chamado NUTCH, pois o mesmo tem código aberto, não possui fins comerciais, possui desenvolvimento ativo, boa documentação, entre outras.

Os métodos de contextualização utilizados foram o de contextualização pela URL, contextualização por referência direta, contextualização por agrupamento por referência direta e contextualização por categorização, utilizando o classificador Rocchio.

Os testes realizados mostraram que apesar de ter diferenças de resultados em outros trabalhos que utilizaram o classificador Rocchio também, devido a alguns fatores, o classificador Rocchio chega muito próximo dos resultados do classificador Support Vector Machines (SVM), considerado um dos melhores classificadores.

6.2 TRABALHOS QUE UTILIZAM DA CONTEXTUALIZAÇÃO

Existem alguns trabalhos pesquisados, apesar de não ter o foco na contextualização, utilizam de algum método para contextualizar como a etiquetagem morfossintática de textos, modelo espaço-vetorial, entre outros.

6.2.1 Extração de Informação de Artigos Científicos: uma Abordagem Baseada em Indução de Regras de Etiquetagem

Essa Dissertação de Mestrado do Curso de Ciências da Computação e Matemática Computacional na Universidade de São Paulo (ÁLVARES, 2007).

Tem como objetivo criar uma Ferramenta Inteligente de Apoio à Pesquisa para apoiar estudantes ou pesquisadores iniciantes na pesquisa de documentos utilizando características relevantes dos artigos e suas relações de similaridade. Sua abordagem na

contextualização utiliza de um processamento lingüístico chamado etiquetagem, que rotula os termos conforme sua classe gramatical para posterior analise.

Na conclusão dos testes mostra que são identificadas e extraídas quase todas as informações comuns em uma referência. Porém mostra que a precisão de extração está diretamente ligada a precisão da etiquetagem como mostra a Figura 28.

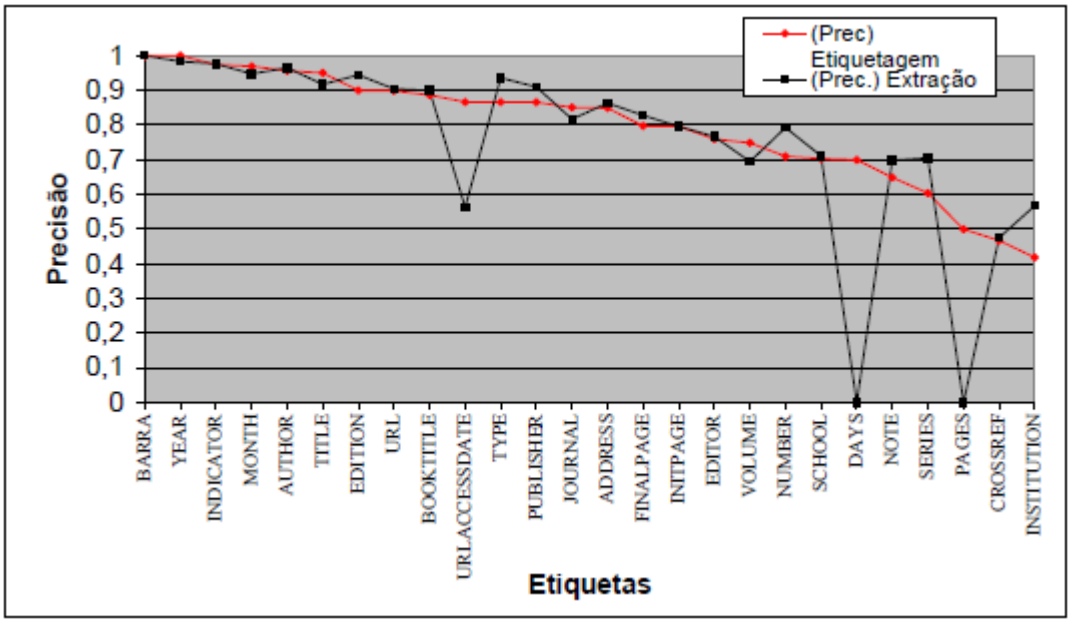


Figura 28. Precisão da etiquetagem e da extração por etiqueta
Fonte: ÁLVARES, A. C. (2007)

6.2.2 Utilizando *Conceitos* como Descritores de Textos para o Processo de Identificação de Conglomerados (*clustering*) de Documentos

Essa Tese de Doutorado do Curso de Pós-Graduação em Ciência da Computação na Universidade de Federal do Rio Grande do Sul, em Porto Alegre, RS (WIVES, 2004).

Tem como objetivo melhorar o processo de identificação e análise de conglomerados de fácil compreensão ao usuário e utilizar de métodos para analise de conglomerados através de conceitos.

Nos resultados envolvendo processos de pré-processamento e manipulação do vocábulo, incluindo a técnica proposta através de conceitos, nota-se uma melhora significável para analise da interpretação dos resultados.

6.2.3 Extração de Informação para Busca Semântica na Web Baseada em Ontologias

Essa Dissertação de Mestrado do Curso de Pós-Graduação em Engenharia Elétrica na Universidade de Federal de Santa Catarina, em Florianópolis, SC (SILVA, 2003).

Tem como objetivo utilizar do sistema MASTER-Web para classificar e armazenar dados extraídos na Web com relevância e armazená-las em uma base de dados. Sua contribuição está na reutilização das regras de extração e ontologias usadas nesse sistema para que possam ser reusadas em diversos outros domínios com pouca programação.

Foram utilizadas 133 páginas classificadas pelo MASTER-Web. Cinco informações foram utilizadas para extração no experimento: localidade, período, data de submissão, data de notificação e tópicos.

Enquanto o processamento da extração era feito, foram feitos alguns ajustes eram feitos para efeito de correção. O primeiro resultado esta especificado na Figura 29, conforme sua Cobertura e Precisão. Cobertura é o número de informações extraídas da página sobre o número de informações apresentadas na página. A Precisão é o número de informações extraídas corretamente e o número de informações extraídas.

	Cobertura	*Precisão*
Local	74,07%	74,07%
Período	86,49%	81,08%
Data limite	78,04%	70,13%
Data de aceitação	93,75%	81,25%
Lista de tópicos	66,67%	59,56%

Figura 29. Resultados individuais da extração
Fonte: SILVA, T. (2003)

Em um segundo teste feito, com novas páginas classificadas pelo MASTER-Web, foram usados os mesmos métodos do primeiro teste, porém sem fazer qualquer ajuste no processamento de extração. A Figura 30 mostra esse novo resultado.

	Cobertura	*Precisão*
Local	68,75%	68,75%
Período	78,49%	57,89%
Data limite	75%	71,43%
Data de aceitação	88,89%	77,78%
Lista de tópicos	70%	59,99%

Figura 30. Resultados individuais da extração com uma nova composição de páginas
Fonte: SILVA, T. (2003)

Os resultados não diferiram muito do primeiro para o segundo teste, comprovando a viabilidade do sistema e cabe salientar que o processo não usa aprendizado de maquina nem dicionário de termos.

7 EXTRAÇÃO DE DADOS NÃO ESTRUTURADOS EM PÁGINAS HTML UTILIZANDO A FERRAMENTA TREETAGGER PARA CONTEXTUALIZAÇÃO DA PESQUISA

Para fazer a extração dos dados não estruturados e para sua contextualização foi escolhida a técnica de etiquetagem morfossintática de texto que utiliza de técnicas de PLN, probabilidade e árvores de decisão. A contextualização é feita usando a ferramenta *TreeTagger* e a extração dos dados não estruturados são feitas comparando os resultados da ferramenta, entre a pesquisa feita pelo usuário e a página HTML.

A ferramenta *TreeTagger* implementa a etiquetagem morfossintática de texto supracitada e tem como objetivo neste projeto classificar os tipos de palavras da língua portuguesa, fornecendo a classe gramatical das palavras da página HTML e da pesquisa realizada pelo usuário, fazendo assim a contextualização da pesquisa.

A implementação do aplicativo deste projeto tem como objetivo utilizar da ferramenta *TreeTagger* para fazer uma contextualização de uma página HTML selecionada na web após uma pesquisa realizada por um usuário e armazenar os dados não estruturados relevantes da pesquisa em um banco de dados.

O protótipo utiliza a frase de pesquisa feita pelo usuário em uma ferramenta de busca na web para classificá-la gramaticalmente por meio da ferramenta *TreeTagger* e também para classificar a página HTML, a fim de encontrar apenas os resultados que estão contextualizados com o assunto da pesquisa.

Para não limitar muito os resultados, utilizam-se apenas as seguintes classes gramaticais para comparação: adjetivo, advérbio, número cardinal/ordinal, nome comum/próprio e verbo.

A fim de avaliar os resultados obtidos pela extração dos dados foram utilizadas algumas páginas selecionas na web, utilizando o *Google*, uma ferramenta de busca gratuita disponível na internet.

Nas páginas HTML são retiradas as *tags* por não apresentarem informações relevantes a pesquisa e utiliza-se o restante do texto na qual é a parte relevante de informações na página para o usuário.

Feito a eliminação das *tags* utilizadas na página HTML, o aplicativo desenvolvido percorre o texto inicialmente apenas para encontrar as palavras da pesquisa que o usuário digitou na página HTML, sem levar em conta a contextualização.

Após uma frase ser encontrada, é passada então para a fase de contextualização, no qual as classificações gramaticais das mesmas palavras serão comparadas para verificar se são do mesmo tipo, se as palavras forem das mesmas classes gramaticais, a frase é armazenada no banco de dados, caso contrário a frase é descartada.

A fim de realizar esta pesquisa foi adotada uma metodologia com o objetivo de compreender os temas abordados, definir métodos, metodologias e métricas para implementação de um protótipo, extrair os dados não estruturados de uma página HTML de uma forma contextualizada e modelar o banco de dados para sua posterior população de dados.

7.1 METODOLOGIA

O desenvolvimento desta pesquisa teve como metodologia adotada as seguintes etapas: levantamento bibliográfico; modelagem do banco de dados; implementação e realização de testes.

A pesquisa bibliográfica deste trabalho fundamentou-se na descrição e compreensão de todos os temas envolvidos, como o estudo de banco de dados objeto relacional, o entendimento de páginas HTML e dados não estruturados, o estudo de técnicas para extração e contextualização de dados não estruturados e o funcionamento da ferramenta TreeTagger.

Os estudos realizados tiveram como foco principal o entendimento de técnicas para extração e contextualização de dados não estruturados, o qual envolveu um período maior em relação às outras etapas, devido a ausência de bibliografia e sua complexidade.

7.1.1 Modelagem do Banco de Dados

Para a modelagem do banco de dados da aplicação desenvolvida neste projeto foi utilizado o software *DB Designer 4*. O banco de dados utilizado para o desenvolvimento foi o *Firebird 2.5*. E a interface gráfica para criação do banco foi com o software *EMS QuickDesk 2.0*.

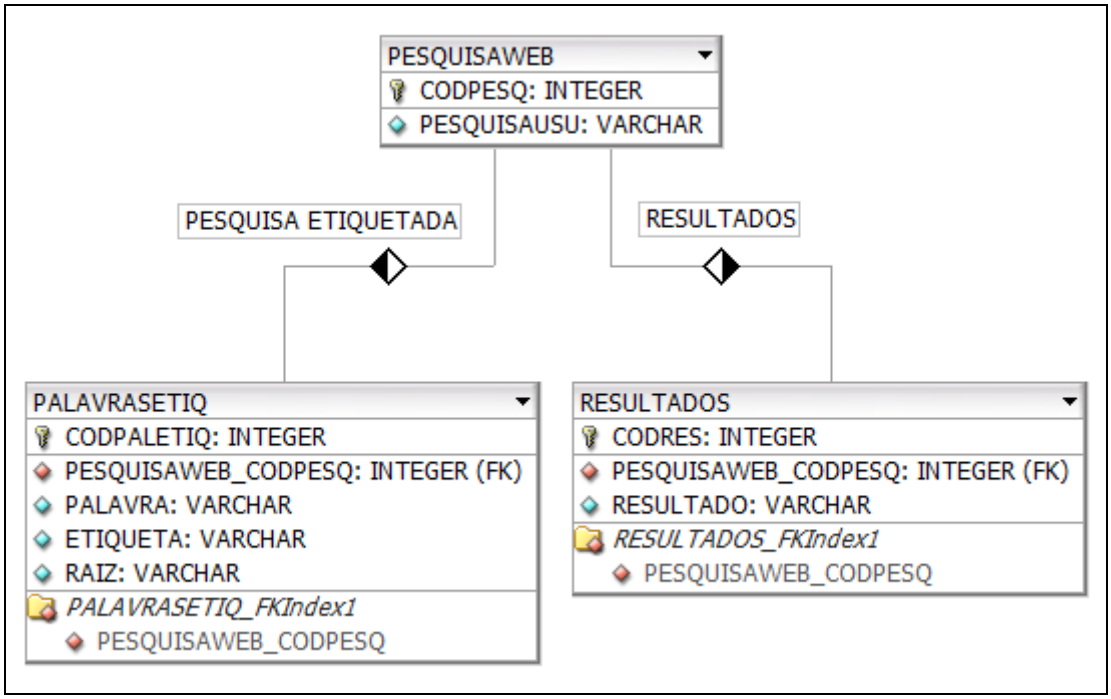


Figura 31. Modelagem do banco de dados

A Figura 31 mostra as tabelas e atributos do banco de dados criado, chamado de ETIQUETADOR_CONTEXTUAL. A tabela PESQUISAWEB contem os atributos:

a) CODPESQ: *integer,* auto incrementado, *primary key (PK)*;

b) PESQUISAUSU: *varchar (100)*, recebe a pesquisa do usuário;

A tabela PALAVRASETIQ contem os atributos:

a) CODPALETIQ: *integer*, auto incrementado, *PK*;

b) PALAVRA: *varchar (100)*, recebe as palavras que compõem a frase da pesquisa do usuário;

c) ETIQUETA: *varchar (10)*, recebe as etiquetas das palavras;

d) RAIZ: *varchar (100)*, recebe as palavras primárias geradas pelo TreeTagger;

e) CODPESQUISAWEB: *integer*, *foreign key (FK)*, recebe a *PK* da tabela PESQUISAWEB.

A tabela RESULTADOS contem os atributos:

a) CODRES: *integer*, auto incrementado, *PK*;

b) RESULTADO: *varchar (10000)*, recebe o resultado gerado pela aplicação desenvolvida neste projeto;

c) CODPESQUISAWEBRES: *integer*, *foreign key (FK)*, recebe a *PK* da tabela PESQUISAWEB.

7.1.2 Implementação e Realização de Testes

O protótipo foi implementado por meio da linguagem c++ e do ambiente de programação C++ Builder 6. Ao protótipo foi dado o nome de Extração Contextualizada. Será apresentada, com um exemplo feito nos testes, a utilização do protótipo.

Ao iniciar o protótipo coloca-se no campo na parte superior da tela, indicado ao lado de *PESQUISA WEB*, a frase de busca que irá ser feita em uma ferramenta de busca na web e clicar no botão *OK*. Feito isso, a frase será armazenada no banco de dados e irá ser copiada para o espaço em branco do lado superior esquerdo. A Figura 32 mostra a tela inicial e o local do texto de pesquisa.

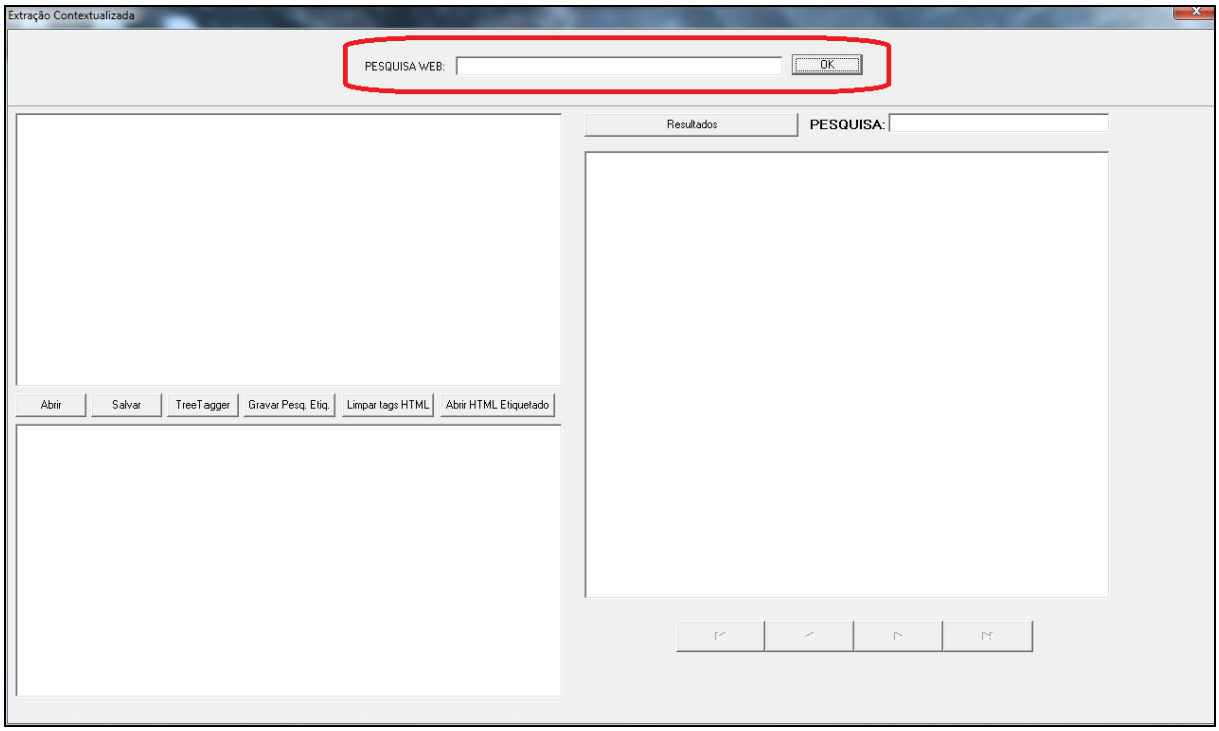


Figura 32. Tela inicial do protótipo e local de escrita do texto de pesquisa

A seguir abre-se um browser de internet, neste projeto foi utilizado o *Mozilla Firefox 4* e acessar uma ferramenta de busca na web, como por exemplo, o *Google* para fazer uma pesquisa com o mesmo texto utilizado no protótipo.

Selecionar uma página do resultado da pesquisa e abrir o código fonte contendo o HTML da página. No caso do Mozilla Firefox 4 é feito da seguinte maneira, acessar o botão *FireFox*, item *Desenvolvedor web->Código-fonte*. Na janela do código fonte usar Ctrl+A do teclado para copiar todo o texto e depois Ctrl+C para copiar, em seguida usar Ctrl+V para colar em um bloco de notas. Salvar o conteúdo do bloco de notas. Sugere-se criar uma pasta com o nome do site e o domínio e o arquivo com o nome do site seguido de HTML para uma melhor organização.

Voltando ao protótipo, em seguida é necessário salvar a frase em um documento texto, clicando no botão *Salvar*, como mostra a Figura 33. Sugere-se salvar o arquivo com nome de *pesquisa* e guardar na mesma pasta onde foi colocado o arquivo texto da página HTML.

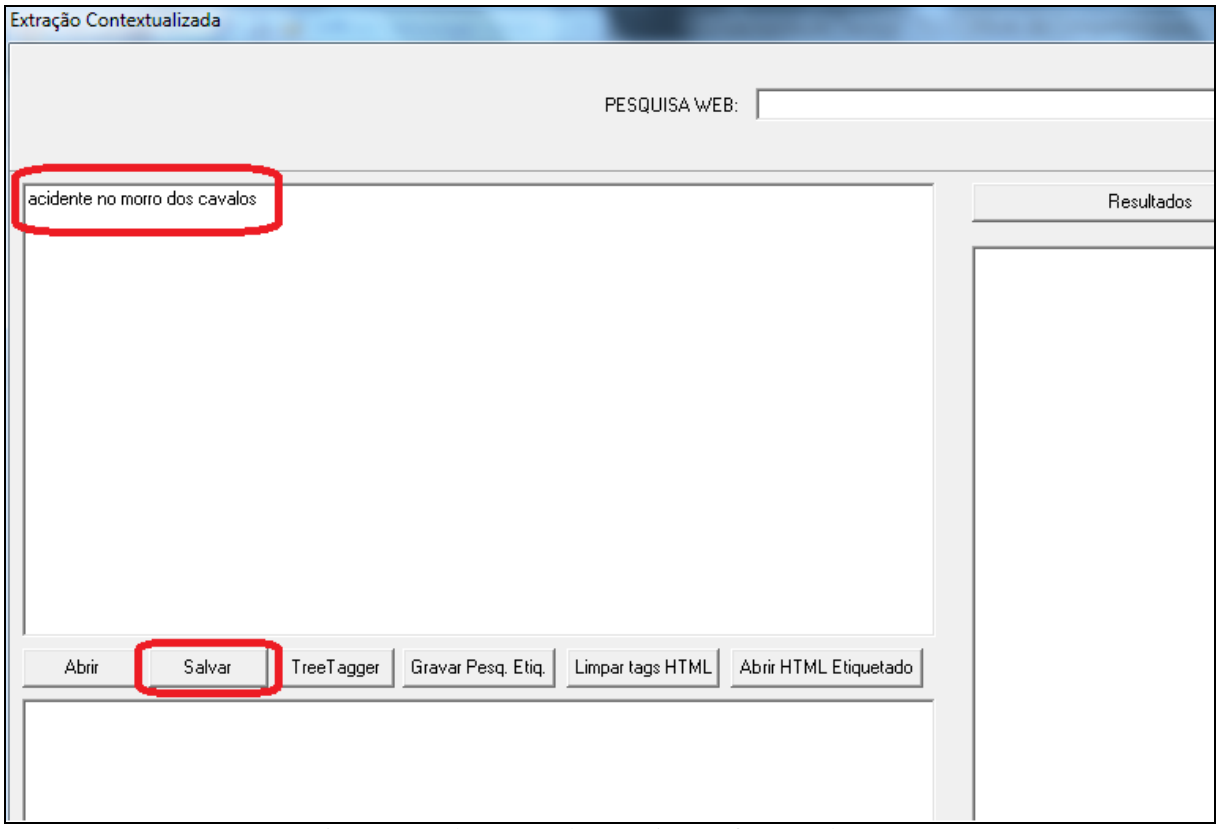


Figura 33. Salvar texto da pesquisa em formato de texto

Logo após utilizar a ferramenta *TreeTagger* para fazer a etiquetação da frase de pesquisa realizada que foi salva com o nome de *pesquisa*. Para abrir o *TreeTagger* basta clicar no botão *TreeTagger* que está ao lado do botão *Salvar* mostrado na Figura 33. A utilização da ferramenta *TreeTagger* está documentada no Capítulo 7 deste trabalho. Sugere-se dar o nome de *pesquisaetiquetada* ao arquivo de saída da ferramenta.

Depois de etiquetar a pesquisa é necessário abri-la para gravar no banco de dados clicando no botão *Abrir* e selecionando o arquivo *pesquisaetiquetada*. Ao aparecer o texto do arquivo no espaço em branco do lado superior esquerdo clicar no botão *Gravar Pesq. Etiq*. A Figura 34 demonstra esse processo.

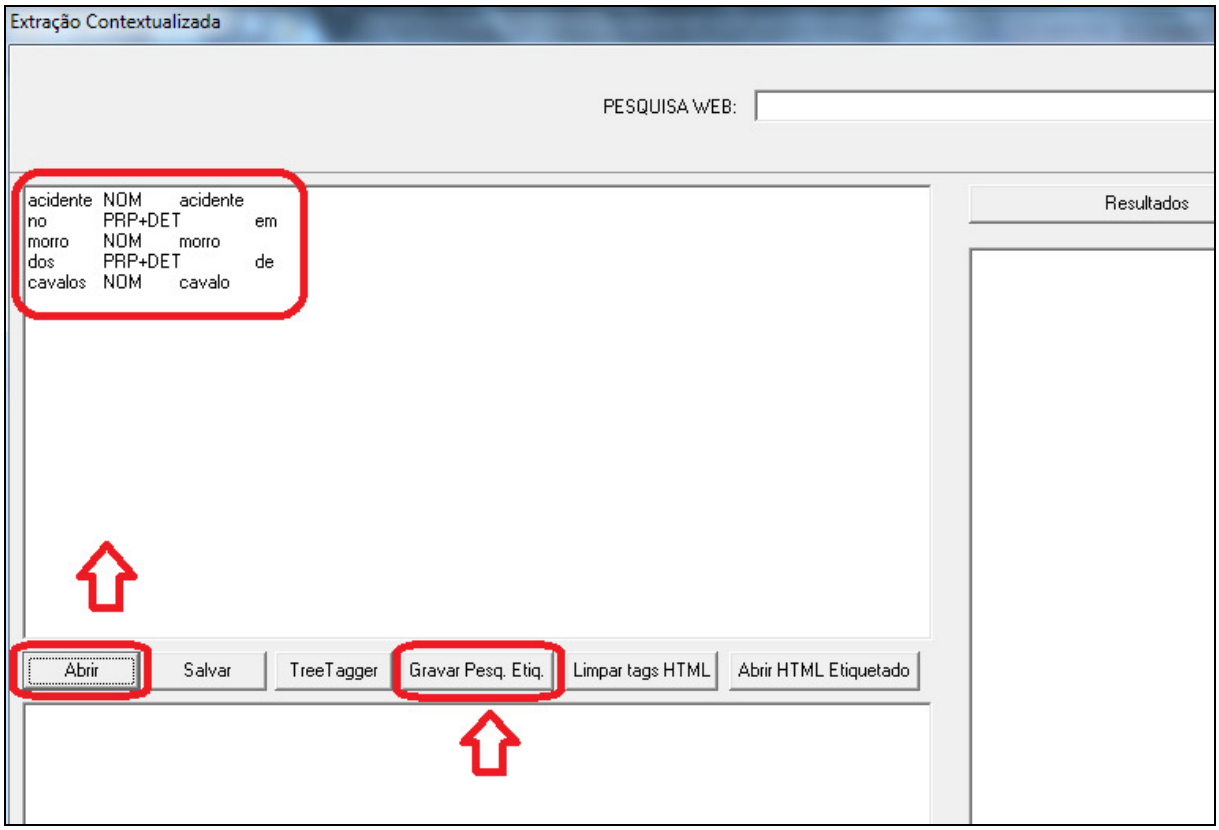


Figura 34. Abrir pesquisa etiquetada e gravar no banco de dados

Abrir a página HTML salva com o botão *Abrir* e depois clicar no botão *Limpar tags HTML* para eliminar as *tags* do texto HTML e salvar novamente o novo documento texto clicando no botão *Salvar*, conforme é apresentando na Figura 35. Sugere-se nomear o arquivo com mesmo nome e acrescentar a palavra *limpo*. Ex.: *clicrbsHTMLlimpo*.

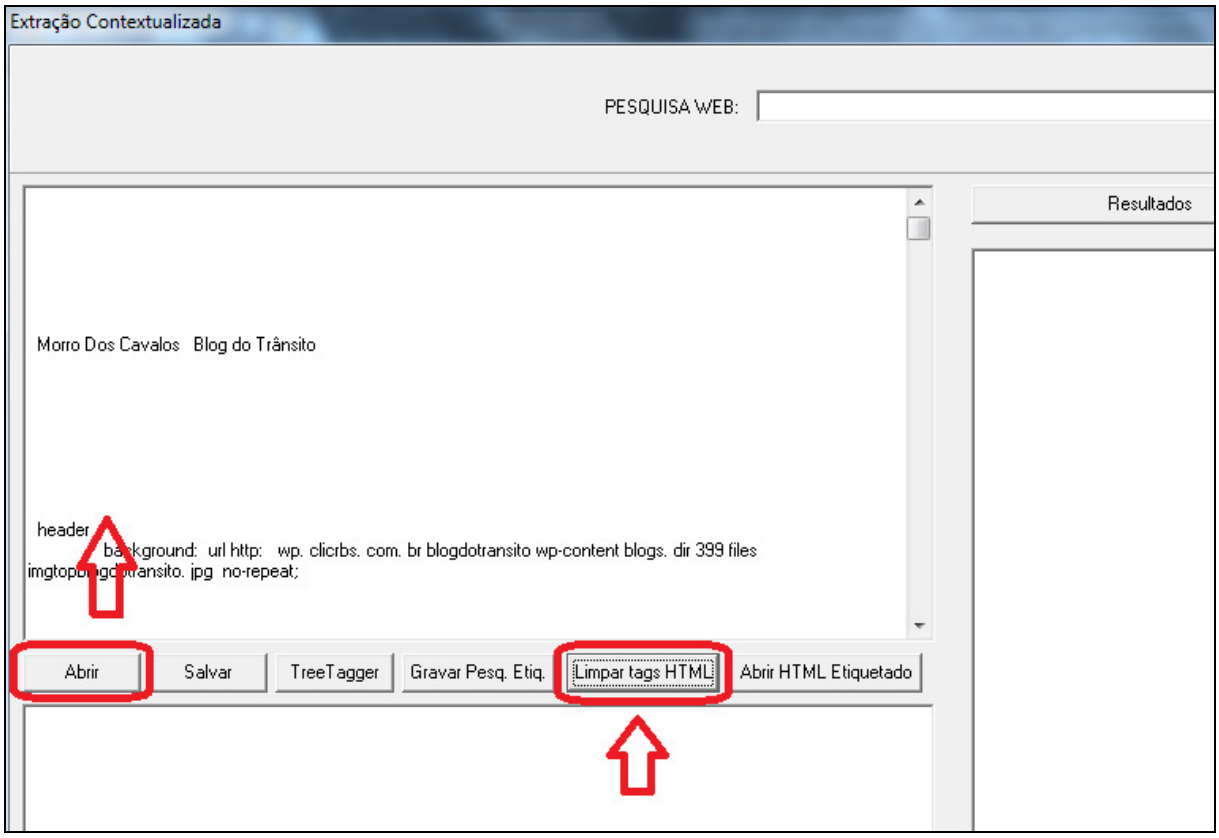


Figura 35. Abrir página HTML e limpar *tags*

Em seguida usar novamente a ferramenta TreeTagger para etiquetar o arquivo HTML limpo (sem *tags*). Sugere-se colocar o nome do arquivo de saída da ferramenta com o mesmo nome inicial trocando o nome *limpo* por *etiquetado*. Ex.: clicrbsHTMLetiquetado. Depois de etiquetado, abrir o arquivo usando o botão *Abrir HTML Etiquetado*, como mostra a Figura 36.

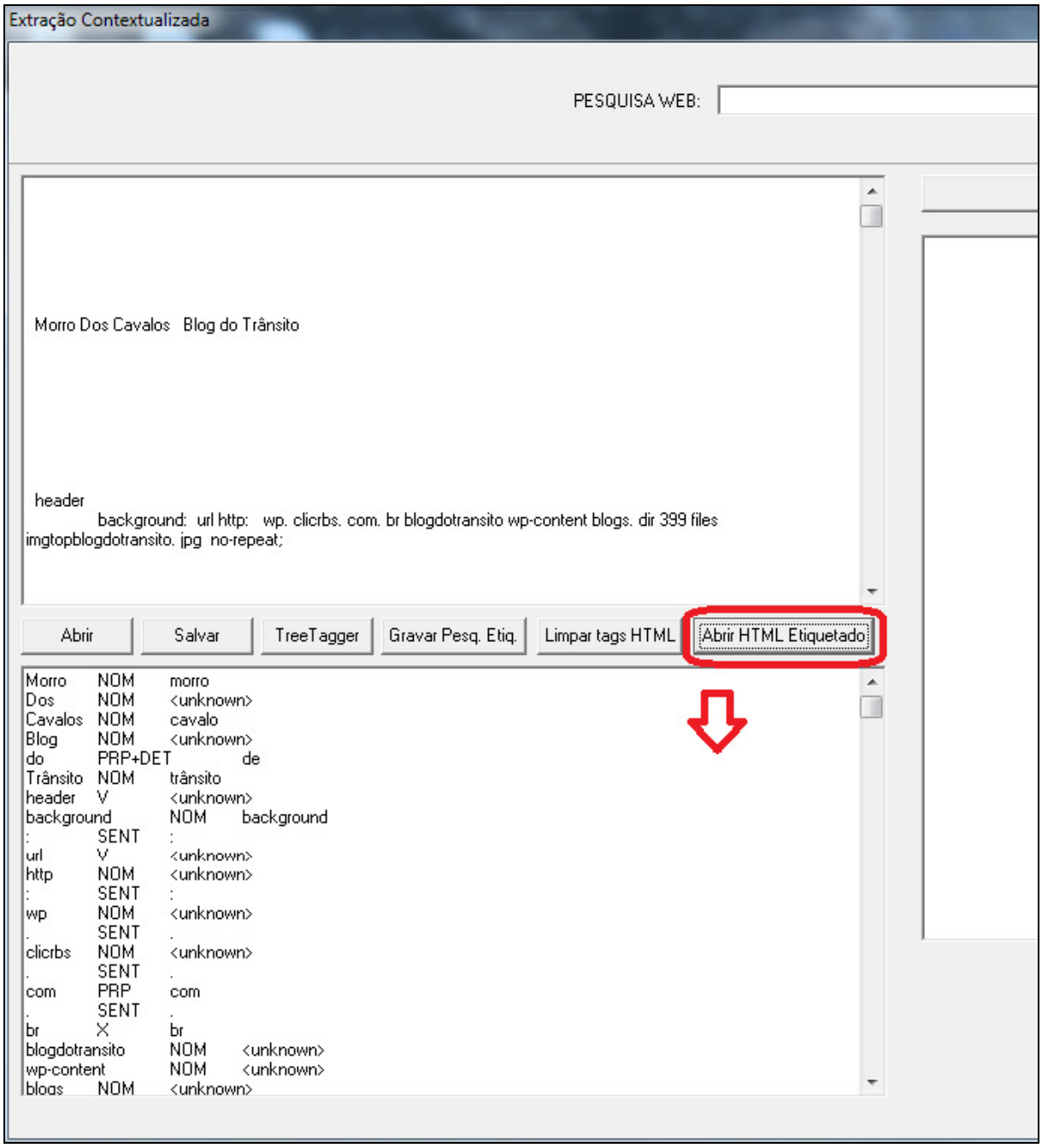


Figura 36. Abrir arquivo HTML etiquetado

Para verificar os resultados, basta clicar no botão *Resultados* e será mostrado no espaço em branco do lado direito os resultados armazenados no banco de dados. Para passar pelos registros foi colocado um componente com setas para percorrê-los (Figura 37).

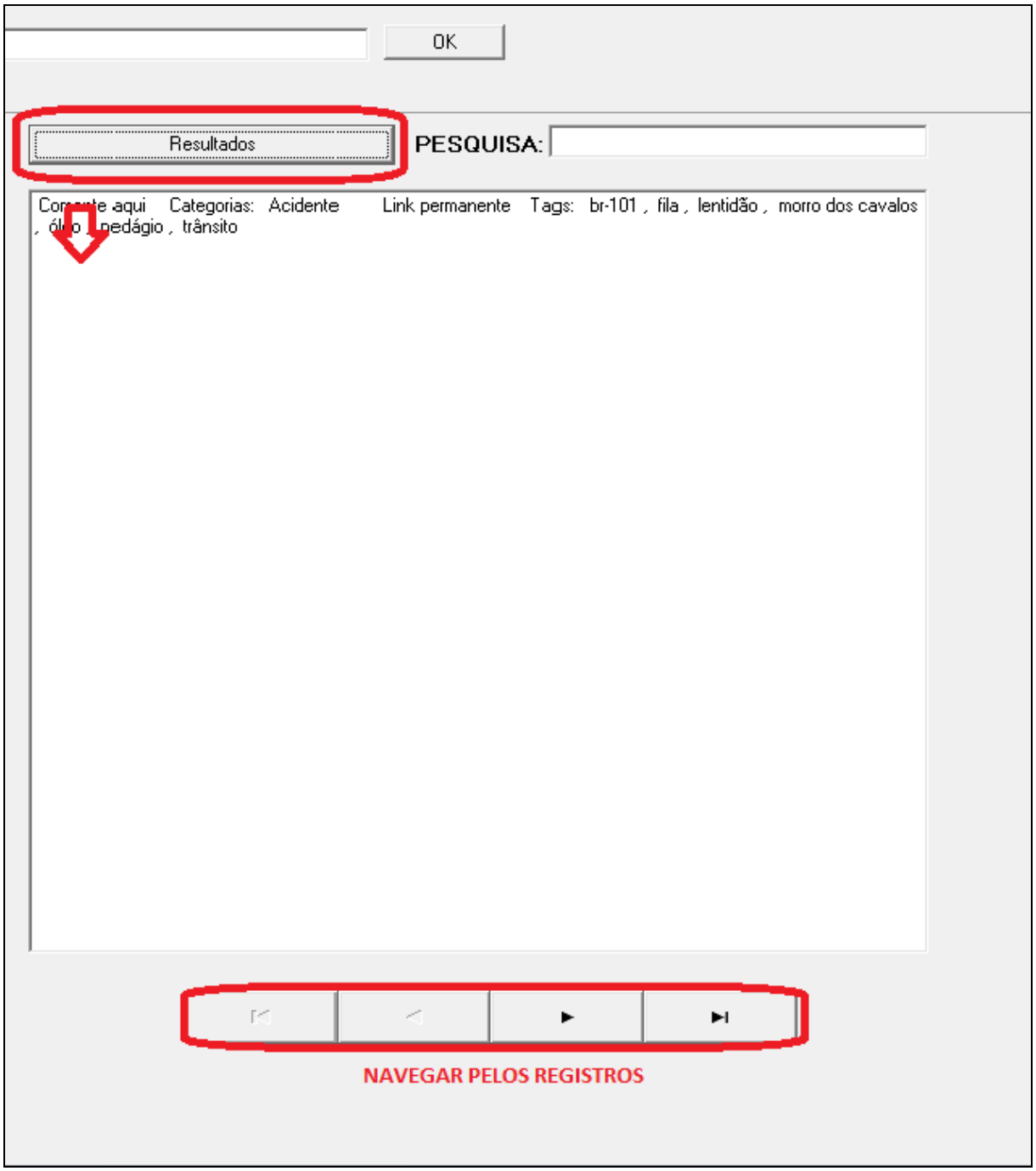


Figura 37. Resultados e navegação de registros

Para melhorar a pesquisa foi colocado um campo para filtrar informações adicionais dentro dos registros. Neste exemplo os resultados dos registros podem ser filtrados com a palavra *combustível* e apenas um registro irá aparecer, conforme mostra a Figura 38.

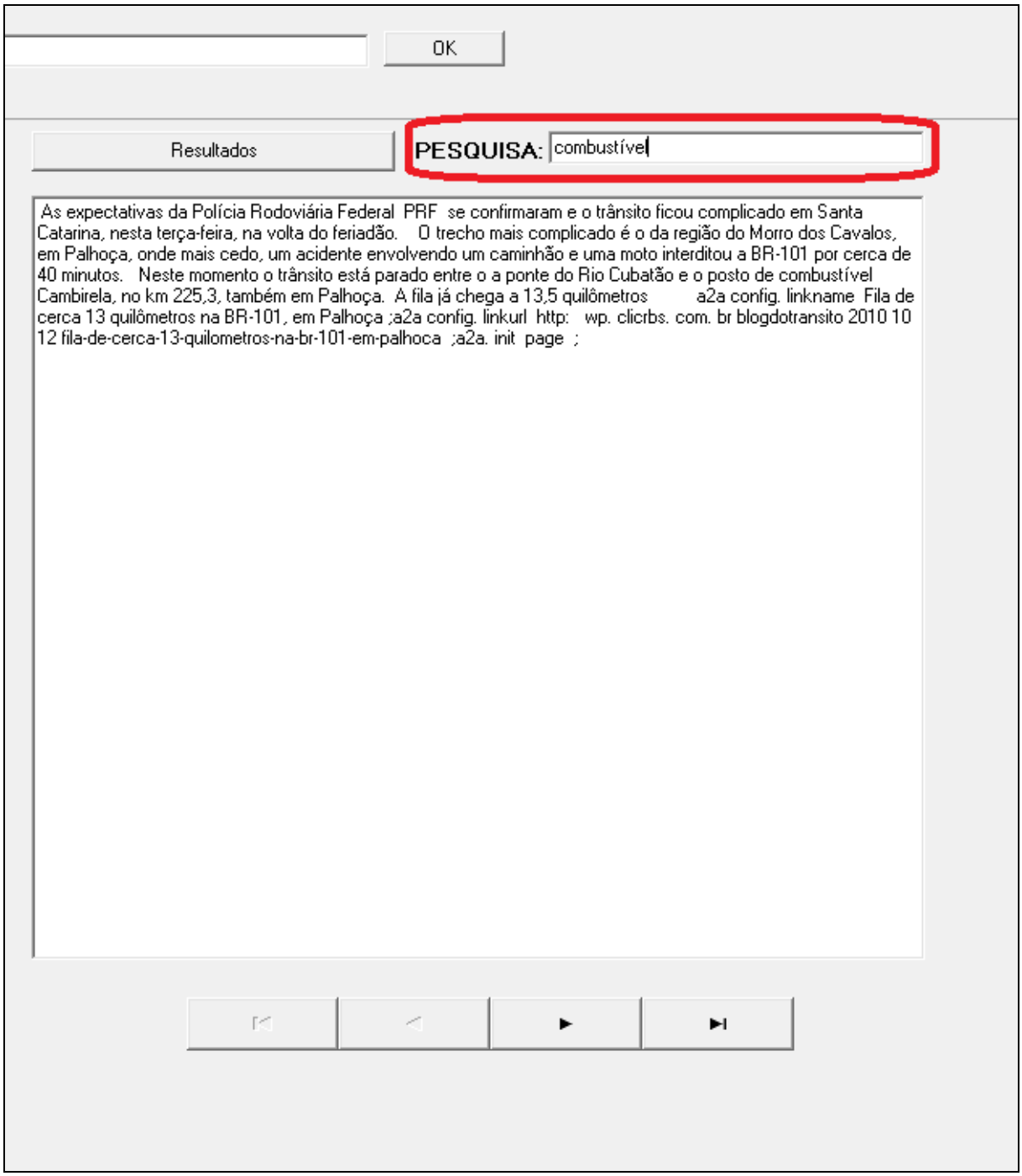


Figura 38. Filtro de pesquisa para os registros

Finalizada a implementação do protótipo, foram efetuados alguns testes para a fim de verificar os resultados gerados.

7.2 RESULTADOS OBTIDOS

Os resultados foram obtidos utilizando as métricas de avaliação precisão, cobertura e F-measure (em particular F-1), como forma de estimar a qualidade de extração das informações das páginas HTML.

Segundo Álvares (2007) Precisão (precision) é definida como a quantidade de informações corretamente extraídas sobre todas as informações extraídas, ou seja, refere-se a quanto de informações confiantes foram extraídas. Cobertura (recall) é definida como a quantidade de informações corretamente extraídas sobre todas as informações extraídas, ou seja, refere-se o quanto de informação relevante foi corretamente extraída.

Na Figura 39 é mostrada as formulas da precisão (P) e da cobertura (C) respectivamente.

$$P = \frac{N_c}{N_p}$$

$$C = \frac{N_c}{N_t}$$

Figura 39. Formulas da precisão e da cobertura
Fonte: ÁLVARES, A. (2007)

Onde Nc é o número de *slots* que foram corretamente preenchidos pelo sistema, Np é o número total de *slots* que foram preenchidos pelo sistema e Nt é o número total de *slots* que deveriam ser preenchidos pelo sistema. Essas medidas são inversamente relacionadas, quanto maior a precisão menor a cobertura e quanto maior a cobertura menor a precisão.

Para outra avaliação utilizando as medidas de cobertura e precisão utiliza-se outra medida chamada F-measure que avalia o sistema de extração (Figura 40).

$$F - measure = \frac{(\beta^2 + 1) * C * P}{\beta^2 * (C + P)}$$

Figura 40. Formula *F-measure*
Fonte: ÁLVARES, A. (2007)

Onde o parâmetro β quantifica a preferência da cobertura sobre a precisão. Frequentemente é usado 1 como valor desse parâmetro para balancear as duas medidas como mostra a Figura 41.

$$F_1 = \frac{2 * C * P}{C + P}$$

Figura 41. Formula com o parâmetro $\beta = 1$
Fonte: ÁLVARES, A. (2007)

Na realização dos testes do protótipo para extrair dados não estruturados de páginas HTML, utilizou-se um notebook com sistema operacional Windows Seven Ultimate, processador Intel Core 2 Duo 2GHz e 3GB de memória RAM.

Para a avaliação dos resultados foram feitas três pesquisas na web, as frases utilizadas foram “acidente no morro dos cavalos”, “manifestação de bombeiros no rio” e “greve dos professores” respectivamente. Foi escolhida uma página de cada assunto retornada pela ferramenta de busca na web *Google*.

Foram utilizadas as seguintes URL para cada pesquisa respectivamente:

a) http://www.clicrbs.com.br/esportes/rs/noticias/default,36255,Dupla-brasileira-cai-na-primeira-rodada-em-Auckland.html
b) http://oglobo.globo.com/pais/mat/2011/06/04/manifestacao-de-bombeiros-no-rio-termina-com-439-presos-924615638.asp.
c) http://g1.globo.com/rio-de-janeiro/noticia/2011/06/greve-de-professores-atinge-menos-de-2-da-categoria-diz-secretaria.html

Apesar da retirada das *tags* nas páginas HTML ainda existem formatações utilizadas nos textos fora das *tags*, por isso essas formatações vieram muitas vezes misturadas

aos resultados obtidos, mas não prejudicaram no entendimento das informações. Nos resultados mostrados são marcadas em negrito as formatações para melhor identificá-las.

Os resultados para a primeira pesquisa foram as seguintes:

a) Acidente congestiona BR-101 na altura do Morro dos Cavalos Notícias RS - clicEsportes;

b) **raquo; Uacute;ltimas not iacute;cias Uacute;ltimas not iacute;cias Imprimir Enviar Corrigir Alterar tamanho da letra: A- A** Acidente congestiona BR-101 na altura do Morro dos Cavalos **OAS publicidade. OAS AD Middle1**;

c) Um acidente entre os kms 238 e 239 da BR-101, em Palhoça, Grande Florianópolis, perto do Morro dos Cavalos, complica o tráfego na região, especialmente na saída da Praia da Pinheira. Segundo as primeiras informações da Polícia Rodoviária Federal, a pista da rodovia já está liberada, e patrulheiros estão orientando o trânsito no local. As vítimas do acidente foram levadas de helicóptero para hospitais da Grande Florianópolis. Ainda não há detalhes sobre as causas do acidente, nem o número de veículos envolvidos. **Link de Origem google ad client pub-2809266142650922; Esportes, 300x250, 16 03 09 google ad slot 1186308446;google ad width 300;google ad height 250;google language pt; document nbsp; Imprimir Enviar Corrigir Voltar Topo document. ready…**

Por meio da métrica utilizada nesta pesquisa os resultados apresentaram os seguintes valores: precisão = 66%, cobertura = 100% e F1 = 79%.

Para a segunda pesquisa foram observados os seguintes resultados:

a) Manifestação de bombeiros no Rio termina com 439 presos - O Globo;

b) Manifestação de bombeiros no Rio termina com 439 presos;

c) RIO DE JANEIRO Reuters - Uma manifestação de bombeiros no Rio de Janeiro, iniciada na véspera, resultou em invasão a quartel, tiros, exoneração do comandante da corporação e detenção de 439 rebelados.

Por meio da métrica utilizada nesta pesquisa os resultados apresentaram os seguintes valores: precisão = 66%, cobertura = 100% e F1 = 79%.

Na terceira pesquisa os resultados foram os seguintes:

a) G1 - Greve de professores atinge menos de 2% da categoria, diz secretaria - notícias em Rio de Janeiro;

b) Greve de professores atinge menos de 2% da categoria, diz secretaria Sindicato iniciou paralisação na terça-feira 7. Professores vão participar de um ato nas escadarias da Alerj;

c) A Secretaria estadual de Educa **231; 227;o** Seeduc diz que a greve de professores das escolas do estado do Rio atingem menos de 2% dos 51 mil docentes. A Seeduc ainda **est 225;** verificando se **h 225;** escolas estaduais sem funcionamento total.

Por meio da métrica utilizada nesta pesquisa os resultados apresentaram os seguintes valores: precisão = 100%, cobertura = 75% e F1 = 85%. A média obtida pelo *F-measure* foi de 81%.

CONCLUSÃO

Existem algumas técnicas para extração de dados não estruturados na web, entre elas está a de Processamento de Linguagem Natural. Com essa técnica é possível fazer uma analise sintática, semântica, morfológica e contextual de textos. Com essa análise se consegue extrair textos de forma contextualizada e de relevância mesmo não contendo estrutura nenhuma.

Com isso esta pesquisa fundamentou-se no entendimento da extração de dados não estruturados de forma contextualizada, utilizando a ferramenta TreeTagger que implementa a etiquetagem morfossintática de textos e utiliza de técnicas de Processamento de Linguagem Natural, probabilidades e árvores de decisão para classificar gramaticalmente o texto de páginas HTML assim como classificar também a pesquisa de um usuário em uma ferramenta de busca na web, a fim de compará-los aumentando a precisão dos resultados, para posteriormente armazenar os dados em um banco de dados para sua melhor manipulação.

Durante a pesquisa, algumas dificuldades foram encontradas, especialmente na implementação da contextualização dos dados. Mesmo eliminando as *tags* das páginas HTML ainda sobram no texto frases e palavras que fazem parte da formatação HTML que não possuem coerência com o texto e acabam atrapalhando em alguns momentos para a contextualização da frase e consequentemente em alguns resultados.

No entanto, estas dificuldades não alteraram de uma forma significativa os resultados finais e os objetivos desta pesquisa em relação a extração dos dados, a contextualização e a implementação do protótipo foram alcançados.

Após a compreensão do método e a implementação do protótipo, foram executados alguns testes os quais demonstraram resultados satisfatórios com informações relevantes da pesquisa realizada.

Considerando a pesquisa realizada, são descritas algumas sugestões de trabalhos futuros com objetivo de dar continuidade ao projeto:

a) Implementar um novo protótipo para utilizar o TreeTagger em uma ferramenta de busca na web de código aberto a fim de classificar de acordo com o contexto as páginas retornadas para o usuário;
b) Desenvolver outra etapa do protótipo para eliminar as formatações remanescentes das páginas HTML;
c) Utilizar outras técnicas de extração de dados aplicadas com a ferramenta TreeTagger, como por exemplo, ontologias;
d) Utilizar outras técnicas de contextualização para comparar com a pesquisa, como por exemplo, modelo contextual.

REFERÊNCIAS

ALBERTO FILHO, Luiz. **Aplicação de Técnicas de Mineração de Dados e Textos no Apoio à Tomada de Decisão em Segurança Pública.** 2009. 79 f. Dissertação (Pós-graduação) - Universidade Federal Do Pará, Belém, 2009.

ÁLVARES, Alberto Cáceres. **Extração de Informação de Artigos Científicos:** uma abordagem baseada em indução de regras de etiquetagem. 2007. 131 f. Dissertação (Mestrado) - Universidade de São Paulo, São Carlos, 2007.

ARAUJO, José Marcelo Pereira de. **Processo de Descoberta de Conhecimento em Dados Não-Estruturados: Estudo de Caso para a Inteligência Competitiva.** 2007. 180 f. Dissertação (Pós-graduação) - Universidade Católica de Brasília, Brasília, 2007.

CALDAS, Paracelso De Oliveira. **Geração de Regras de Extração de Dados em Páginas HTML.** 2003. 63 f. Dissertação (Mestrado) - Universidade Federal Do Rio Grande Do Sul, Porto Alegre, 2003.

CARDOSO, Rafael Cunha. **Um Sistema de Recuperação e Extração de Informação Utilizando Conceitos da Web Semântica.** 2004. 120 f. Dissertação (Mestrado) - Universidade Federal de Pernambuco, Recife, 2004.

CARVALHO, Alan. **HTML 4 para trainees.** Rio de Janeiro: Book Express, 2001. 183 p.

DATE, C. J. **Introdução a sistemas de bancos de dados.** 7. ed. Rio de Janeiro: Ed. Campus, 2000. 802 p.

FEITOSA, Daniela Soares et al. **Banco de Dados Objetos-Relacionados (BDOR)**. Salvador, BA, jan. 2008.

GRAHAM, Ian S. **HTML, A referência completa para HTML 3.2 e das extensões HTML.** Rio de Janeiro: Ed. Campus, 1998. 640 p.

GUANDELINE, Eidy Leandro Tanaka. **Integração materializada na web:** um Estudo de Caso. 2002. 94 f. Dissertação (Mestrado) - Universidade Federal Do Rio Grande Do Sul, Porto Alegre, 2002.

LOH, Stanley. **Abordagem Baseada em Conceitos para Descoberta de Conhecimento em Textos.** 2001. 110 f. Tese (Doutorado) - Universidade Federal Do Rio Grande Do Sul, Porto Alegre, 2001.

MANFREDINI, Vitor Hugo. **Proposta de uma Técnica de Extração de Informação de Arquivos de Log de Servidores Proxy.** 2003. 53 f. Trabalho de Conclusão de Curso (Bacharel) - Universidade Federal de São Paulo, Londrina, 2003.

MARANGON, Sílvio Luís. **Análise de métodos para programação de Contextualização Dissertação.** 2006. 134 f. Dissertação (Mestrado) - Universidade de São Paulo, São Paulo, 2006.

MORAIS, Edison Andrade Martins. **Contextualização de Documentos em Domínios Representados por Ontologias Utilizando Mineração de Textos.** 2007. 113 f. Dissertação (Mestrado) - Universidade Federal de Goiás, Goiânia, 2007.

OLIVEIRA, Luis Henrique Gonçalves de. **Extração de Metadados utilizando uma ontologia de domínio.** 2009. 67 f. Dissertação (Mestrado) - Universidade Federal do Rio Grande do Sul, Porto Alegre, 2009.

RODRIGUES, Edmilson Faria. **APRENDIZAGEM ESTATÍSTICA PARA RECUPERAÇÃO DA INFORMAÇÃO.** 2008. 63 f. Dissertação (Mestre) - Universidade De Brasília, Brasília, 2008.

SCARINCI, Rui Gureghian. **SES - Sistema de Extração Semântica de Informações.** 1997. 166 f. Dissertação (Mestrado) - Universidade Federal do Rio Grande do Sul, Porto Alegre, 1997.

SCHMID, Helmut. **Probalistic Part-of-Speech Tagging Using Decision Trees**. Stuttgart, p.1-9, dez. 1996.

SEGUNDO, Alonso de Oliveira; ROCHA, Gabriel Almeida; NEVES, Nilton Aquino Das. **Banco de Dados Objeto-Relacional (BDOR)**. Salvador, BA, jan. 2008.

SILBERSCHATZ, Abraham; KORTH, Henry F.; SUDARSHAN, S. **Sistema de banco de dados.** Rio de Janeiro: Elsevier, 2006. 781 p.

SILVA, Edilberto Magalhães. **Descoberta de Conhecimento com o uso de Text Mining:** Cruzando o Abismo de Moore. 2002. 175 f. Dissertação (Mestrado) - Universidade Católica de Brasília, Brasília, 2002.

SILVA, Tércio De Morais Sampaio. **EXTRAÇÃO DE INFORMAÇÃO PARA BUSCA SEMÂNTICA NA WEB BASEADA EM ONTOLOGIAS.** 2003. 93 f. Dissertação (Mestrado) - Universidade Federal de Santa Catarina, Florianópolis, 2003.

SILVEIRA, Fábio Fagundes. **Técnicas de Banco de Dados para a World Wide Web.** Universidade da Região da Campanha, Bagé, 2009.

VIVAN, Orlando Miguel. **Usando base de dados relacionais para geração semi-automática de ontologias destinada à extração de dados.** 2003. 79 f. Dissertação (Mestrado) - Universidade Federal Do Rio Grande Do Sul, Porto Alegre, 2003.

WIVES, Leandro Krug. **Um estudo sobre Agrupamento de Documentos Textuais em Processamento de Informações não Estruturadas Usando Técnicas de "Clustering".** 1999. 84 f. Dissertação (Mestrado) - Universidade Federal Do Rio Grande Do Sul, Porto Alegre, 1999.

WIVES, Leandro Krug; LOH, Stanley; OLIVEIRA, José Palazzo M. de. **Concept-Based Knowledge Discovery in Texts Extracted from the Web.** Porto Alegre, p.29-39, 10 jul. 2000.

WIVES, Leandro Krug. **Utilizando conceitos como descritores de textos para o processo de identificação de conglomerados (clustering) de documentos.** 2004. 136 f. Tese (Doutorado) - Universidade Federal Do Rio Grande Do Sul, Porto Alegre, 2004.

ZAMBENEDETTI, Christian. **Extração de Informação sobre Bases de Dados Textuais.** 2002. 144 f. Dissertação (Mestrado) - Universidade Federal do Rio Grande do Sul, Porto Alegre, 2002.

APÊNDICE A – ARTIGO CIENTÍFICO

Analise de Técnicas de Extração de Dados Não Estruturados em Páginas HTML para Utilização no Armazenamento e Manipulação em Banco de Dados Objeto Relacional

Felipe Cogorni Mauricio[1], Paracelso de Oliveira Caldas[2]

[1]Acadêmico do Curso de Ciência da Computação – Unidade Acadêmica de Ciências, Engenharias e Tecnologias – Universidade do Extremo Sul Catarinense (UNESC) – Criciúma – SC

[2] Professor do Curso de Ciência da Computação – Unidade Acadêmica de Ciências, Engenharias e Tecnologias – Universidade do Extremo Sul Catarinense (UNESC) – Criciúma – SC

cogorni@hotmail.com, poc@unesc.net

Abstract. *The Internet today is the place where most accesses are done to conduct a research, but due to lack of standardization in the building of sites, specifically in HTML pages, there is a very significant loss of data which could be better utilized, for instance the data unstructured of HTML pages. To use these data in a relevant way, there are techniques that help with the extraction of such data. This article demonstrates the use of TreeTagger tool that uses some of these techniques and classifies grammatically texts HTML assisting in the extraction of data in a contextualized, and then this data is stored in a bunch of data for better manipulation.*

Resumo. *A internet hoje é o local que mais se acessa para a realização de pesquisas, porém devido a falta de padronização na construção de sites, especificamente nas páginas HTML, existe uma perda bastante significativa de dados que poderiam ser melhores aproveitados, são os casos dos dados não estruturados em páginas HTML. Para poder utilizar esses dados de uma forma relevante existem técnicas que auxiliam na sua extração. Este artigo demonstra a utilização da ferramenta TreeTagger que utiliza de algumas dessas técnicas e classifica gramaticalmente os textos HTML auxiliando na extração dos dados de uma forma contextualizada, e posteriormente esses dados são armazenados em um banco de dados para uma melhor manipulação.*

1. Introdução

Considerando o crescente aumento na quantidade de dados estruturados e não estruturados em diversas bases de dados podem ser observadas duas situações. Em se tratando de sistemas que contem dados estruturados, existe hoje uma infinidade de ferramentas para lidar com esses dados, muito exploradas pela área de TI. Porém, no contexto de sistemas que apresentam dados não estruturados, ainda se tem uma carência, sendo de difícil tratamento e prejudicando a manipulação para obtenção de informações.

Sistemas não estruturados não possuem uma estrutura prévia para tratar os dados. São exemplos desses dados: email, fax, documentos de texto, algumas páginas HTML, imagens, entre outros.

Analisando a complexidade e a grande quantidade de dados não estruturados no meio eletrônico, principalmente em páginas web, observa-se a necessidade de extrair e modelar informações contidas nesse meio para que possam ser manipuladas por técnicas e ferramentas já utilizadas em banco de dados estruturados (CALDAS, 2003).

Para obter uma extração dos dados existem técnicas, cada uma com suas características, que podem, de maneira efetiva, extrair os dados relevantes. Porém, para cada caso, se tem a necessidade de pesquisar melhor cada técnica dependendo de sua finalidade. Entre elas encontramos: técnica baseada em Processamento de Linguagem Natural, baseada em ontologias, entre outras. (SILVEIRA, 2009).

2. Técnicas para Extração de Dados Não estruturados

Técnicas de extração da informação podem ser usadas para extrair desde textos estruturados como também sem estrutura nenhuma. Cabendo salientar que a definição de uma técnica é muito importante para cada determinado tipo de documento (ÁLVARES, 2007).

2.1. Mineração de Dados

A Mineração de Dados é o processo para extrair informações válidas, previamente desconhecidas e de muito valor a partir de grandes bases de dados. Permite muito mais que uma simples consulta a um banco de dados, pois possibilita o usuário explorar e inferir informação útil a partir dos dados, descobrindo relacionamentos escondidos (ARAUJO, 2007).

Existem vários modelos de MD os quais são classificados de acordo com a finalidade ou com a classe de aplicação para qual são usados. Cada classe possui um conjunto de algoritmos para extração de relações de uma base de dados. Os principais métodos de MD são Classificação, Associação, Agrupamento e Previsão de séries Seqüenciais – Temporais (ARAUJO, 2007).

2.2. Mineração de Texto

O processo de Mineração de Texto (MT) tem como objetivo coletar informações significantes e conhecimentos relevantes em documentos textuais. Procedimento que implica em um grau de dificuldade grande, pois geralmente em documentos textuais é usado linguagem natural, sem a preocupação com a padronização ou estruturação dos dados (ALBERTO FILHO, 2009).

MT é o conjunto de técnicas e ferramentas inteligentes que auxiliam na análise de grandes volumes de dados textuais, com intuito de filtrar conhecimento útil, em qualquer domínio que utiliza textos não estruturados (ARAUJO, 2007).

A área de MT envolve quatro etapas principais: a Coleta de Dados, a de Pré-processamento ou preparação dos dados, a etapa de Análise dos Dados e Extração do Conhecimento e a etapa de Pós-processamento (ARAUJO, 2007).

2.3. Indução de Wrappers

Wrappers são ferramentas de consulta e atualização de dados em uma fonte. Basicamente ele faz uma seleção de dados relevantes para serem extraídos e planeja sua estrutura para serem apresentados. As operações de um wrapper são feitas através de uma linguagem de extração específica (CALDAS, 2003).

Sistemas wrappers localizam informações relevantes em textos estruturados, utilizando esses dados, por exemplo, em um banco de dados. Porém, no contexto da web, o

propósito de um wrapper é converter informações implícitas armazenadas em páginas HTML em informações explícitas estruturadas, para posterior processamento (ÁLVARES, 2007).

2.4. Baseado em Ontologias

Uma ontologia é utilizada para representar o conhecimento de um determinado domínio, ou seja, representam os objetos, as relações entre eles, as restrições de integridade, os aninhamentos, os tipos de dados e a cardinalidade. Para isso tem-se a necessidade de um perito no domínio de ontologia. Ela é utilizada para compartilhar informações por diversas aplicações (CALDAS, 2003).

Um banco de dados é populado a partir da extração de dados em documentos através de regras pré-estabelecidas pela ontologia. Após a extração é possível ter uma consulta mais precisa com relacionamentos existentes no domínio da aplicação (GUANDELINE, 2002).

2.5. Processamento de Linguagem Natural

O Processamento de Linguagem Natural (PLN) é usado por diversas ferramentas para aprender regras de extração a fim de extrair informações úteis em documentos. Estas regras são baseadas em restrições sintáticas e semânticas, aplicando técnicas para relacionar frases e elementos da oração (CALDAS, 2003).

PLN consiste no desenvolvimento de métodos computacionais para a análise e manipulação ou codificação de informações expressas em língua natural. O objetivo é compreender a língua, por exemplo, sumarizando uma grande quantidade de texto ficará muito melhor de utilizá-lo de forma relevante. Algumas das principais tarefas de PLN são reconhecimento de contexto, análise sintática, semântica, léxica e morfológica, sumarização e tradução de textos (ÁLVARES, 2007).

As técnicas de PLN são amplamente utilizadas em documentos com pouca ou nenhuma estruturação. O objetivo é compreender textos em alguma língua natural, a fim de encontrar dados que sejam relevantes para extraí-los (MANFREDINI, 2003).

2.6. Técnicas Estatísticas de Aprendizado de Máquina

A área de aprendizado de máquina estuda construir programas que possam aprender e melhorar seus desempenhos a partir de experiências. Então quando seu desempenho aumenta em relação a determinadas tarefas ele aprende com essa experiência (OLIVEIRA, 2009).

No Aprendizado de Maquina é aplicado indução sobre um conjunto particular de exemplos (experiências) para se obter conclusões genéricas. Existem dois tipos principais de aprendizado indutivo: supervisionado e não-supervisionado (OLIVEIRA, 2009).

No supervisionado, ou com professor, é feito por meio de exemplos rotulados no formato entrada/saída desejada. O algoritmo do Aprendizado de Maquina gera um modelo correto para rotular novas entradas (OLIVEIRA, 2009).

No aprendizado não-supervisionado os exemplos rotulados não existem. O algoritmo deve, através de uma medida de qualidade, rotular os dados de entrada por grupos e conforme sua representação (OLIVEIRA, 2009).

Estas técnicas estão sendo utilizadas especialmente para modelo de uma estrutura a partir de dados e como usar da melhor forma dados rotulados e não rotulados. Um exemplo dessa técnica é o *Hidden Markov Models*, são sistemas determinísticos de aprendizado de regras que extraem informações de textos não estruturados e transformam em um registro estruturado (ÁLVARES, 2007).

2.7. Baseada em Modelo

Ferramentas que usam dessa técnica utilizam objetos criados por usuários que servem como modelos. A técnica procura por porções de dados na web que se compare com o modelo pré-definido e o ajusta na estrutura (CALDAS, 2003).

Essa técnica se baseia na aquisição de dados na web que se assemelham a estrutura de objetos de interesse, ou seja, um padrão de objeto que deve ser procurado em uma página (CARDOSO, 2004).

A estrutura é provida conforme um conjunto de primitivas de modelagem (como tuplas, por exemplo) que estão de acordo com um modelo de dados. Depois são aplicados algoritmos que procuram por objetos que estão de acordo com a estrutura padrão definida (CARDOSO, 2004).

3. Extração e Manipulação de Dados Não Estruturados

Define-se Recuperação de Informação como um processo de busca em documentos relevantes em resposta a uma consulta que favoreça as necessidades de informação de um usuário, por exemplo, usando métodos estatísticos para o cálculo de similaridade entre dois documentos (ÁLVARES, 2007).

Ainda que a web seja rica e grande em relação a documentos, ela também é bastante confusa. Assim para amenizar essa confusão a RI desempenha um papel importante para a recuperação de documentos relevantes, porém frustram as expectativas dos usuários, grande parte das vezes, em pesquisas na web, retornando informações fora do contexto (CARDOSO, 2004).

Para tratar desse problema foram pesquisados alguns métodos para contextualizar a extração de dados em páginas HTML ao assunto de interesse do usuário.

3.1. Consulta por Termos

É a forma básica de consulta fornecida pelos mecanismos de busca, buscam por um ou mais termos, utilizando de várias formas de consulta: consulta booleana, consulta por proximidade, etc (MARANGON, 2006).

3.2. Contextualização pela URL

A utilização da URL para o contexto de uma página. É a analise das informações contidas no esquema de localização e acesso dentro da URL (MARANGON, 2006).

3.3. Contextualização por Referência Direta

Os *hyperlinks* das páginas webs formam um grafo com sentido único, porém se usados em sentido invertido poderia ser muito utilizado, por exemplo, para *clipping* de notícias para uma determinada empresa, analisando páginas que fazem referência à empresa (MARANGON, 2006).

3.4. Modelo Contextual

Loh (2001) indica em seu trabalho um modelo contextual que também visa a contextualização para a representação de um conceito. É utilizada uma lista de termos chamados positivos e negativos para comparação. Para se considerar o conceito, todos os termos positivos devem estar presentes e nenhum termo negativo pode aparecer. Por exemplo, no domínio medico, o conceito “alcoolismo” pode ser representado pelas regras:

1. álcool –nega;

2. hálito etílico.

Sendo que o símbolo “-“ indica um termo negativo. O “-nega” serve para eliminar frases do tipo “o paciente nega uso de álcool”.

3.5. Modelo Espaço-Vetorial

O conceito é definido com um conjunto de palavras que devem ser encontradas dentro de documentos. Esse conjunto é representado por um vetor de termos, que tem um grau de afinidade com o conceito.

Foi selecionado dentre os métodos para contextualização, a etiquetagem morfossintática de texto que utiliza, dentre outras, a técnica de PLN para classificar textos gramaticalmente conforme o seu contexto (WIVES, 2004).

4. Etiquetagem morfossintática de textos

A etiquetagem morfossintática de um texto implica em utilizar de etiqueta ou rótulo (*tag*), com etiquetas já predefinidas (*tagset*), para definir palavras, símbolos de pontuação, palavra estrangeira, ou fórmula matemática presente em um texto, conforme o seu contexto (ÁLVARES, 2007).

Os etiquetadores podem ser classificados em três tipos: simbólica, probabilística e híbrida. Na simbólica os etiquetadores se baseiam em regras, restrições e árvores de decisões não probabilísticas. Na abordagem probabilística, são fundamentadas através de técnicas como Processamento de Linguagem Natural, redes neurais, máxima entropia, Modelo de Markov e árvores de decisões probabilísticas (ÁLVARES, 2007).

A Figura 1 mostra um exemplo de uma etiquetagem.

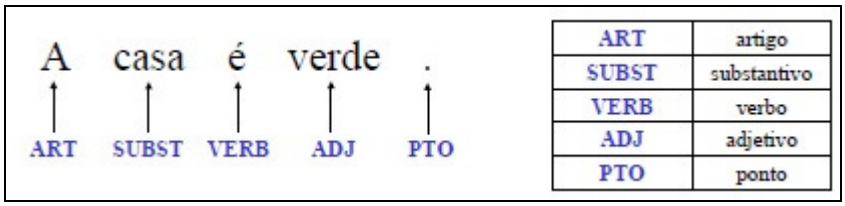


Figura 1. Exemplo de etiquetagem

O etiquetador pode ser induzido automaticamente, usando de regras que analisam as palavras e/ou seu contexto (análise das palavras ou etiquetas que vem antes e depois da palavra em foco analisada) (ÁLVARES, 2007).

Segundo Álvares (2007) a tarefa do etiquetador pode ser dividida em três módulos: escrutinador léxico, classificador gramatical e desambiguizador. O escrutinador léxico é responsável por separar (*tokenization*) e identificar as orações no texto. O classificador gramatical classifica os termos do texto com o auxilio, geralmente, de um léxico. O desambiguizador utiliza informações do contexto para determinar a etiqueta de termos que podem pertencer a mais de uma classe gramatical (ÁLVARES, 2007).

Considerando isso, esta pesquisa consiste na utilização da etiquetagem morfossintática de textos para classificar gramaticalmente os textos das páginas HTML e também classificar uma pesquisa de um usuário feita em uma ferramenta de busca na web, a fim de compará-los e extrair informações relevantes e dentro do contexto da busca realizada. Para isso é utilizada uma ferramenta chamada TreeTagger que implementa a etiquetagem morfossintática de textos.

5. Ferramenta Tree Tagger

Segundo Schmid (1996) a ferramenta *TreeTagger* utiliza de técnicas de PLN para fazer a etiquetagem morfossintática do texto. A etiquetagem utiliza de árvores de decisão para tratar as probabilidades de transição. Baseado nesse método o *TreeTagger* alcançou 96,36% de precisão.

Os possíveis contextos não são apenas trigramas, bigramas e unigramas, mas também contexto usando a árvore de decisão binária para estimar as probabilidades de transição. A probabilidade de um trigrama, por exemplo, é determinado pela sequencia do caminho correspondente, como mostra a Figura 2.

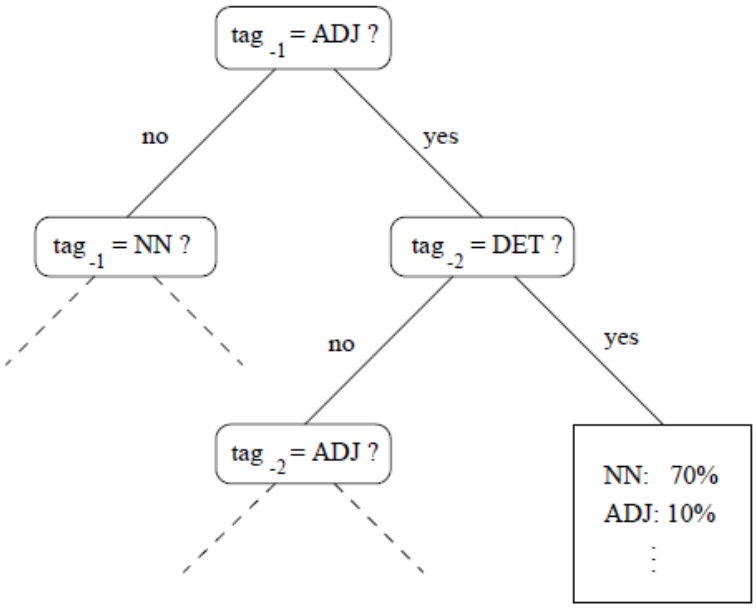


Figura 2. Uma simples árvore de decisão

O *TreeTagger* possui também um léxico com as probabilidades de *tags* primárias para cada palavra. O léxico possui três partes: *fullform lexicon*, *suffix lexicon* e *default entry*. O *fullform lexicon* é a palavra armazenada na sua forma primária. É o primeiro a ser procurado, se a palavra for encontrada a *tag* correspondente é retornada. Se a palavra não é achada entra na segunda parte, o *suffix léxico* que faz a verificação dos sufixos das palavras, mostrado na Figura 3. O *default entry* é utilizado caso o *suffix lexicon* falhar. São verificados os sufixos mais freqüentes e devolvido a *tag* mais semelhante.

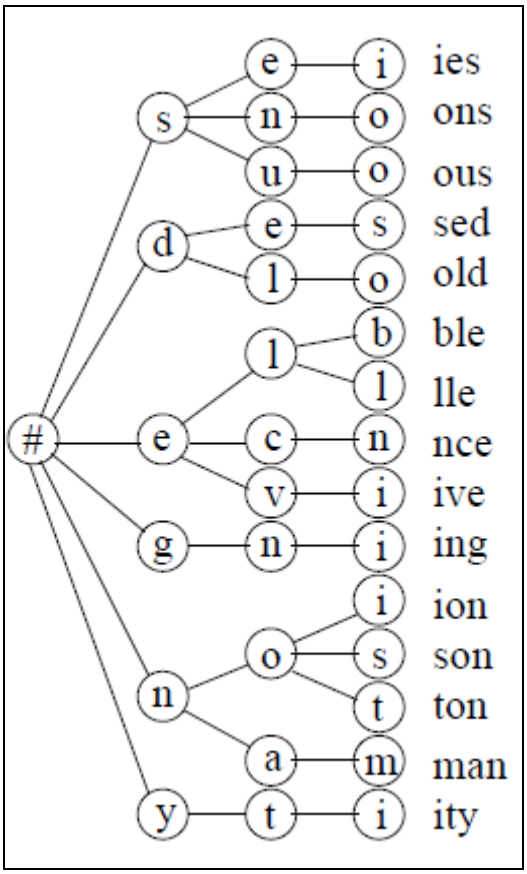


Figura 3. Uma simples árvore de sufixo de tamanho 3

A ferramenta então faz a classificação gramatical de cada palavra no texto, usando o espaço entre elas para defini-las. Segue abaixo as etiquetas definidas pelo autor da implementação da língua portuguesa na ferramenta:

l) ADJ: adjetivo;

m) ADV: advérbio;

n) DET: determinante;

o) CARD: número cardinal/ordinal;

p) NOM: nome comum/próprio;

q) P: pronome;

r) PREP: preposição;

s) V: verbo;

t) I: interjeição;

u) VIRG: separadores dentro da oração;

v) SENT: separadores de orações.

Existem também combinações: PREP+DET, por exemplo, nas palavras “do”, “das”, etc.; V+P, por exemplo, nas palavras “levou-me”, “disse-lhe”, etc.

A ferramenta também já classifica gramaticalmente conforme o contexto da frase, por exemplo, a frase “Eu prego as boas atitudes.” Será classificada da seguinte maneira:

g) Eu: P (pronome);

h) prego: V (verbo);

i) as: DET (determinante);

j) boas: ADJ (adjetivo);

k) atitudes: NOM (nome comum);

l) .: SENT (separador de oração).

Enquanto que a frase “Esse prego é grande” será classificada assim:

f) Esse: DET (determinante);

g) prego: NOM (nome comum);

h) é: V (verbo);

i) grande: ADJ (adjetivo);

j) .: SENT (separador de oração).

Observa-se que a palavra “prego” foi classificada de maneira diferente nas duas frases conforme sua contextualização. Na primeira frase, “prego” vem do verbo “pregar” e na segunda “prego” vem do substantivo.

6. Resultados

Os resultados foram obtidos utilizando as métricas de avaliação precisão, cobertura e F-measure (em particular F-1), como forma de estimar a qualidade de extração das informações das páginas HTML.

Segundo Álvares (2007) Precisão (precision) é definida como a quantidade de informações corretamente extraídas sobre todas as informações extraídas, ou seja, refere-se a quanto de informações confiantes foram extraídas. Cobertura (recall) é definida como a quantidade de informações corretamente extraídas sobre todas as informações extraídas, ou seja, refere-se o quanto de informação relevante foi corretamente extraída.

Para outra avaliação utilizando as medidas de cobertura e precisão utiliza-se outra medida chamada F-measure que avalia o sistema de extração (Figura 4).

$$F - measure = \frac{(\beta^2 + 1) * C * P}{\beta^2 * (C + P)}$$

Figura 4. Formula *F-measure*

Onde o parâmetro β quantifica a preferência da cobertura sobre a precisão.

Para a avaliação dos resultados foram feitas três pesquisas na web, as frases utilizadas foram “acidente no morro dos cavalos”, “manifestação de bombeiros no rio” e “greve dos professores”. Foi escolhida uma página de cada assunto retornada pela ferramenta de busca na web *Google*.

Por meio da métrica utilizada nesta pesquisa os resultados apresentaram os seguintes valores para cada pesquisa citada respectivamente:

a) Precisão = 66%, Cobertura = 100% e F1 = 79%;

b) Precisão = 66%, Cobertura = 100% e F1 = 79%;

c) Precisão = 100%, Cobertura = 75% e F1 = 85%.

A média obtida pelo *F-measure* foi de 81%.

7. Considerações Finais

Existem algumas técnicas para extração de dados não estruturados na web, entre elas está a de Processamento de Linguagem Natural. Com essa técnica é possível fazer uma analise sintática, semântica, morfológica e contextual de textos. Com essa análise se consegue extrair textos de forma contextualizada e de relevância mesmo não contendo estrutura nenhuma.

Com isso esta pesquisa fundamentou-se no entendimento da extração de dados não estruturados de forma contextualizada, utilizando a ferramenta TreeTagger que implementa a etiquetagem morfossintática de textos e utiliza de técnicas de Processamento de Linguagem Natural, probabilidades e árvores de decisão para classificar gramaticalmente o texto de páginas HTML assim como classificar também a pesquisa de um usuário em uma ferramenta de busca na web, a fim de compará-los aumentando a precisão dos resultados, para posteriormente armazenar os dados em um banco de dados para sua melhor manipulação.

Após a compreensão do método e a implementação do protótipo, foram executados alguns testes os quais demonstraram resultados satisfatórios com informações relevantes da pesquisa realizada.

Referências

ALBERTO FILHO, Luiz. Aplicação de Técnicas de Mineração de Dados e Textos no Apoio à Tomada de Decisão em Segurança Pública. 2009. 79 f. Dissertação (Pós-graduação) - Universidade Federal Do Pará, Belém, 2009.

ÁLVARES, Alberto Cáceres. Extração de Informação de Artigos Científicos: uma abordagem baseada em indução de regras de etiquetagem. 2007. 131 f. Dissertação (Mestrado) - Universidade de São Paulo, São Carlos, 2007.

ARAUJO, José Marcelo Pereira de. Processo de Descoberta de Conhecimento em Dados Não-Estruturados: Estudo de Caso para a Inteligência Competitiva. 2007. 180 f. Dissertação (Pós-graduação) - Universidade Católica de Brasília, Brasília, 2007.

CALDAS, Paracelso De Oliveira. Geração de Regras de Extração de Dados em Páginas HTML. 2003. 63 f. Dissertação (Mestrado) - Universidade Federal Do Rio Grande Do Sul, Porto Alegre, 2003.

CARDOSO, Rafael Cunha. Um Sistema de Recuperação e Extração de Informação Utilizando Conceitos da Web Semântica. 2004. 120 f. Dissertação (Mestrado) - Universidade Federal de Pernambuco, Recife, 2004.

GUANDELINE, Eidy Leandro Tanaka. Integração materializada na web: um Estudo de Caso. 2002. 94 f. Dissertação (Mestrado) - Universidade Federal Do Rio Grande Do Sul, Porto Alegre, 2002.

LOH, Stanley. Abordagem Baseada em Conceitos para Descoberta de Conhecimento em Textos. 2001. 110 f. Tese (Doutorado) - Universidade Federal Do Rio Grande Do Sul, Porto Alegre, 2001.

MANFREDINI, Vitor Hugo. Proposta de uma Técnica de Extração de Informação de Arquivos de Log de Servidores Proxy. 2003. 53 f. Trabalho de Conclusão de Curso (Bacharel) - Universidade Federal de São Paulo, Londrina, 2003.

MARANGON, Sílvio Luís. Análise de métodos para programação de Contextualização Dissertação. 2006. 134 f. Dissertação (Mestrado) - Universidade de São Paulo, São Paulo, 2006.

OLIVEIRA, Luis Henrique Gonçalves de. Extração de Metadados utilizando uma ontologia de domínio. 2009. 67 f. Dissertação (Mestrado) - Universidade Federal do Rio Grande do Sul, Porto Alegre, 2009.

SCHMID, Helmut. Probalistic Part-of-Speech Tagging Using Decision Trees. Stuttgart, p.1-9, dez. 1996.

SILVEIRA, Fábio Fagundes. Técnicas de Banco de Dados para a World Wide Web. Universidade da Região da Campanha, Bagé, 2009.

WIVES, Leandro Krug. Utilizando conceitos como descritores de textos para o processo de identificação de conglomerados (clustering) de documentos. 2004. 136 f. Tese (Doutorado) - Universidade Federal Do Rio Grande Do Sul, Porto Alegre, 2004.