Classificação de páginas na internet

José Martins Junior

Orientador: *Prof. Dr. Edson dos Santos Moreira*

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências de Computação e Matemática Computacional.

USP – São Carlos
Fevereiro/2003

“VERSÃO REVISADA APÓS A DEFESA”

Data da Defesa:	11/04/2003
Visto do Orientador:	

Agradecimentos

Este espaço dedica-se ao reconhecimento de todas as pessoas que contribuíram direta ou indiretamente para a realização do presente trabalho. Agradeço primeiramente a Deus por ter me suprido a boa saúde e a coragem de que tanto precisei nessa etapa de minha vida. Agradeço a todos os meus familiares que inspiraram e apoiaram com carinho a realização deste trabalho, e souberam suportar os momentos mais difíceis de minha ausência em seu convívio.

Agradeço ao meu orientador, professor Edson dos Santos Moreira, por ter acreditado e investido em meu potencial, dando-me assim a oportunidade de aprender com sua ampla sabedoria e sua notória experiência na área de estudos envolvida. Sou também grato pela ajuda prestada por meus amigos do grupo de pesquisas do Laboratório Intermídia e do projeto DEEPSIA, e pelas orientações e dicas de estudo que recebi do professor André Carlos Ponce de Leon Ferreira de Carvalho e da doutoranda Cristiane Yae-Mi Imamura.

Por fim, agradeço a todos aqueles que me incentivaram, que por conseqüente reforçaram a minha crença de que iria alcançar tal objetivo.

À minha esposa Rita, aos meus filhos Thalles,
Fernanda e Augusto e aos meus pais José e Myltes

Índice

1	INTRODUÇÃO	1
1.1	PROBLEMÁTICA.....	2
1.2	OBJETIVOS	4
1.3	HISTÓRICO DO GRUPO DE PESQUISAS	4
1.4	ESTRUTURA DA DISSERTAÇÃO	5
2	COMÉRCIO ELETRÔNICO.....	6
2.1	O COMÉRCIO ELETRÔNICO NA WEB	7
2.2	O PROJETO DEEPSIA	12
2.3	CONSIDERAÇÕES FINAIS	17
3	AGENTES	18
3.1	DEFINIÇÃO DE AGENTE	18
3.2	TIPOS DE AGENTES	19
3.3	UTILIZAÇÃO DE AGENTES NO COMÉRCIO ELETRÔNICO	22
3.4	CONSIDERAÇÕES FINAIS	23
4	ONTOLOGIA	24
4.1	DEFINIÇÃO DE ONTOLOGIA	25
4.2	UTILIZAÇÃO DE ONTOLOGIA NO COMÉRCIO ELETRÔNICO	28
4.3	WEB SEMÂNTICA (<i>SEMANTIC WEB</i>)	31
4.4	CONSIDERAÇÕES FINAIS	33
5	CLASSIFICAÇÃO DE TEXTOS	34
5.1	CLASSIFICAÇÃO AUTOMÁTICA DE TEXTOS	36
5.2	PROCESSO GERAL DA CLASSIFICAÇÃO AUTOMÁTICA DE TEXTOS	37
5.2.1	<i>Obtenção de documentos .....</i>	<i>37</i>
5.2.2	<i>Pré-processamento dos documentos obtidos .....</i>	<i>38</i>
5.2.3	<i>Extração do conhecimento .....</i>	<i>40</i>
5.2.4	<i>Avaliação do conhecimento .....</i>	<i>41</i>
5.3	SUPPORT VECTOR MACHINES	44
5.3.1	<i>VC Dimension e Structural Risk Minimization .....</i>	<i>45</i>
5.3.2	<i>SVM na Classificação de Textos .....</i>	<i>47</i>
5.3.3	<i>SVM^{light} – Uma Implementação do Algoritmo.....</i>	<i>48</i>
5.4	APLICAÇÃO DE TC NA CLASSIFICAÇÃO DE PÁGINAS NA WEB	50
5.5	CONSIDERAÇÕES FINAIS	51
6	A SOLUÇÃO ADOTADA	52
6.1	METODOLOGIA.....	53
6.1.1	<i>Análise do problema.....</i>	<i>53</i>
6.1.2	<i>Escolha do método de classificação de textos .....</i>	<i>56</i>
6.1.3	<i>Desenvolvimento dos modelos de representação .....</i>	<i>56</i>
6.1.4	<i>Amostragem do conjunto problema.....</i>	<i>66</i>
6.1.5	<i>Aplicação do método e geração dos resultados parciais .....</i>	<i>68</i>
6.2	TECNOLOGIA UTILIZADA NA IMPLEMENTAÇÃO DO EXPERIMENTO.....	72
6.3	CONSIDERAÇÕES FINAIS	73
7	RESULTADOS OBTIDOS	74
8	CONCLUSÕES E PROPOSTA PARA TRABALHOS FUTUROS.....	75
9	REFERÊNCIAS BIBLIOGRÁFICAS	76

Lista de Figuras

Figura 1 – Arquitetura do sistema DEEPSIA.....	13
Figura 2 – Abstração de um domínio.....	24
Figura 3 – Exemplo de estrutura de uma ontologia.....	27
Figura 4 – Exemplo de classificação no modelo UNSPSC.....	30
Figura 5 – Arquitetura da Web Semântica.....	32
Figura 6 – Representação dos hiperplanos no espaço vetorial.....	44
Figura 7 – Subconjuntos aninhados de funções, ordenados pela <i>VC dimension</i>	47
Figura 8 – Exemplo de código HTML.....	53
Figura 9 – Código HTML interpretado pelo browser.....	54
Figura 10 – Exemplo positivo.....	55
Figura 11 – Exemplo negativo.....	55
Figura 12 – (a) Conjunto das páginas brasileiras, e (b) arquivo de entrada do SVM ^{light}	57
Figura 13 – Obtenção de documentos.....	58
Figura 14 – Classe <i>DocumentExample</i>	59
Figura 15 – O processo <i>parsing</i>	60
Figura 16 – Classe <i>HtmlParser</i>	60
Figura 17 – O processo de extração de palavras.....	61
Figura 18 – Classe <i>TextTokenizer</i>	62
Figura 19 – Classe <i>Dictionary</i>	63
Figura 20 – Classe <i>FeaturesVectors</i>	64
Figura 21 – Classe <i>FeaturesFrequencies</i>	65
Figura 22 – Classe <i>FeaturesIDF</i>	65
Figura 23 – Classe <i>FeaturesDF</i>	66

Lista de Tabelas

Tabela 1 – Matriz de confusão para um modelo com classe bi-valorada	42
Tabela 2 – Conjuntos de exemplos amostrados.....	67
Tabela 3 – Amostras obtidas pelo <i>Crawler Agent</i>	67
Tabela 4 – Exemplos obtidos das amostras do <i>Crawler Agent</i>	67
Tabela 5 – Resultados de testes realizados em partes do conjunto de treinamento (1)	69
Tabela 6 – Resultados de testes realizados em partes do conjunto de treinamento (2)	69
Tabela 7 – Resultados de testes realizados em partes do conjunto de treinamento (3)	70
Tabela 8 – Avaliação dos resultados apresentados pelo <i>Crawler Agent</i>	70
Tabela 9 – Resultados dos testes realizados sobre as amostras do <i>Crawler Agent</i>	71
Tabela 10 – Resultados de testes realizados em outros conjuntos de exemplos (1).....	71
Tabela 11 – Resultados de testes realizados em outros conjuntos de exemplos (2).....	71
Tabela 12 – Sumário da avaliação dos resultados obtidos na realização dos testes	74

Resumo

O grande crescimento da Internet ocorreu a partir da década de 1990 com o surgimento dos provedores comerciais de serviços, e resulta principalmente da boa aceitação e vasta disseminação do uso da Web. O grande problema que afeta a escalabilidade e o uso de tal serviço refere-se à organização e à classificação de seu conteúdo. Os engenhos de busca atuais possibilitam a localização de páginas na Web pela comparação léxica de conjuntos de palavras perante os conteúdos dos hipertextos. Tal mecanismo mostra-se ineficaz quando da necessidade pela localização de conteúdos que expressem conceitos ou objetos, a exemplo de produtos à venda oferecidos em sites de comércio eletrônico. A criação da *Web Semântica* foi anunciada no ano de 2000 para esse propósito, visando o estabelecimento de novos padrões para a representação formal de conteúdos nas páginas Web. Com sua implantação, cujo prazo inicialmente previsto foi de dez anos, será possível a expressão de conceitos nos conteúdos dos hipertextos, que representarão objetos classificados por uma ontologia, viabilizando assim o uso de sistemas, baseados em conhecimento, implementados por agentes inteligentes de software. O projeto DEEPSIA foi concebido como uma solução centrada no comprador, ao contrário dos atuais *Market Places*, para resolver o problema da localização de páginas Web com a descrição de produtos à venda, fazendo uso de métodos de classificação de textos, apoiados pelos algoritmos *k-NN* e *C4.5*, no suporte ao processo decisório realizado por um agente previsto em sua arquitetura, o *Crawler Agent*. Os testes com o sistema em sites brasileiros denotaram a necessidade pela sua adaptação em diversos aspectos, incluindo-se o processo decisório envolvido, que foi abordado pelo presente trabalho. A solução para o problema envolveu a aplicação e a avaliação do método *Support Vector Machines*, e é descrita em detalhes.

Abstract

The huge growth of the Internet has been occurring since 90's with the arrival of the internet service providers. One important reason is the good acceptance and wide dissemination of the Web. The main problem that affects its scalability and usage is the organization and classification of its content. The current search engines make possible the localization of pages in the Web by means of a lexical comparison among sets of words and the hypertext's contents. In order to find contents that express concepts or object, such as products for sale in electronic commerce sites such mechanisms are inefficient. The proposition of the Semantic Web was announced in 2000 for this purpose, envisioning the establishment of new standards for formal contents representation in the Web pages. With its implementation, whose deadline was initially stated for ten years, it will be possible to express concepts in hypertext's contents, that will fully represent objects classified into an ontology, making possible the use of knowledge based systems implemented by intelligent software's agents. The DEEPSIA project was conceived as a solution centered in the purchaser, instead of current Market Places, in order to solve the problem of finding Web pages with products for sale description, making use of methods of text classification, with k-NN and C4.5 algorithms, to support the decision problem to be solved by an specific agent designed, the Crawler Agent. The tests of the system in Brazilian sites have denoted the necessity for its adaptation in many aspects, including the involved decision process, which was focused in present work. The solution for the problem includes the application and evaluation of the Support Vector Machines method, and it is described in detail.

1 Introdução

A Internet foi projetada na década de 1970 para possibilitar a comunicação entre computadores de diversos centros de pesquisa dos EUA, sendo expandida para outros países a partir da década de 1980 e seu uso então estendido às universidades, instituições governamentais e organizações. Mas foi a partir da década de 1990, quando provedores comerciais passaram a integrar a rede mundial e permitir seu acesso por pessoas comuns e empresas, que a Internet literalmente “explodiu”. Em janeiro de 2002, mais de 140 milhões de computadores no mundo inteiro podiam desfrutar dos serviços por ela prestados (fonte: *Internet Software Consortium*¹).

Em 1990, Tim Berners-Lee havia proposto um novo tipo de aplicação voltada à exibição de documentos. Os hipertextos, dessa forma denominados por associarem referências (ou ligações) a outros documentos no próprio texto, eram exibidos por uma ferramenta gráfica chamada “*WorldWideWeb*”, criada por Berners-Lee. A “*Web*”, como passou a ser conhecida tal aplicação, foi a principal responsável pelo enorme interesse pela Internet e seu conseqüente crescimento. Documentos de hipertexto foram então produzidos em larga escala pelo crescente número de usuários da Internet, descrevendo em seus conteúdos os mais diversos assuntos. Um estudo desenvolvido por [AGG01] estimou a existência de mais de 1 bilhão de páginas na Web, e uma taxa de crescimento de aproximadamente 1 milhão por dia.

Nos últimos anos, muitos recursos foram adicionados à Web permitindo a inclusão e exibição em hipertextos de outros tipos de mídia, como figuras, filmes e sons, tornando a interação nesse universo muito mais atraente e agradável a seus usuários. Mas o grande problema que hoje afeta sua escalabilidade e uso não se refere ao formato ou à aparência das páginas de hipertexto, e sim à organização e à classificação adequadas de seus conteúdos.

Vários mecanismos de busca hoje disponíveis na Internet, como Altavista, Google e Yahoo possibilitam a localização de páginas baseada em palavras-chave fornecidas pelo usuário. A eficiência desses agentes depende da escolha de um conjunto de palavras pelo usuário que provavelmente constarão da descrição textual do conteúdo procurado. Em outras palavras, tais agentes estabelecem comparações léxicas do conjunto de palavras com o conteúdo das páginas na Internet.

---

¹ <http://www.isc.org/ds/WWW-200201/>

Uma forma alternativa seria dotar tais agentes de “inteligência” para permitir a busca de páginas que expressam os conceitos do objeto procurado pelo usuário. O maior obstáculo para esse avanço refere-se à inadequação da principal linguagem de composição de hipertextos utilizada atualmente, a HTML (*Hyper Text Markup Language*), para representar formalmente conceitos.

Para esse propósito, Tim Berners-Lee anunciou em 2000 a criação da Web Semântica. Tal iniciativa, instituída pelo *WWW Consortium*², visa o desenvolvimento de padrões, como o XML (*Extensible Markup Language*) e o RDF (*Resource Description Framework*), para representação de conceitos de objetos em páginas de hipertexto, que constituirão uma ontologia global de classes a ser compartilhada na Web. Infelizmente trata-se de uma solução de médio a longo prazo, tendo em vista o desenvolvimento e a disseminação do uso dos padrões e recursos. O prazo inicialmente previsto foi de dez anos.

Na busca por soluções para a extração de conceitos do atual conteúdo da Web, diversos trabalhos envolvendo a aplicação de métodos de classificação automática de textos foram desenvolvidos nos últimos anos, e resultaram em soluções satisfatórias, dos quais destacam-se: classificação de documentos da Web utilizando aprendizado Bayesiano [NIG99], construção de bases de conhecimento sobre domínios específicos da Web [CRA99] e classificação de textos de páginas Web em categorias predefinidas utilizando *Support Vector Machines* [JOA98].

1.1 Problemática

Um dos mais importantes tipos de serviços na Web atualmente relaciona-se ao desenvolvimento de sistemas *online* aplicados a negócios, ou *e-Business*. A atual onda de globalização da economia encontrou na Internet um meio apropriado para a divulgação e comercialização de bens e serviços. Nichos de mercado, antes inacessíveis por limitações geográficas, agora podem ser alcançados pela vasta abrangência da “rede mundial de computadores”. A realização de transações comerciais através da Internet instituiu uma nova forma de operação, o comércio eletrônico, ou *e-Commerce*.

---

² <http://www.w3c.org/2001/sw/>

A fim de solucionar o problema da divulgação de negócios, muitos portais, ou *Marketplaces*, começaram a surgir. Tais portais estabelecem associações com sites de venda, mediante contrato, e possibilitam a realização pelos usuários da Internet da busca por produtos oferecidos naquele domínio restrito. Sem dúvidas, consistem de um ótimo negócio para fornecedores de bens e serviços.

Porém, tal solução é precária quando analisado o real interesse de um possível cliente em sua busca por bens e serviços que deseja adquirir. A principal intenção de uma pessoa ou empresa, ao buscar por ofertas de bens e serviços, é avaliar qual negócio oferecido melhor lhe favorece, em uma relação custo / benefício. A limitação do domínio na busca reduz as opções de escolha, constituindo a desvantagem do comprador em relação ao fornecedor nesse tipo de negócio proposto pelos portais.

Sistemas como o *Dynamic on-line Internet Purchasing System based on Intelligent Agents* (DEEPSIA) visam apresentar uma solução mais adequada ao cliente, ou comprador, através de processos de classificação de páginas Web em um catálogo, que representa uma ontologia de produtos à venda.

Um importante agente da Arquitetura DEEPSIA, apresentada em detalhes no Capítulo 2, é o *Crawler*. Ele realiza a primeira seleção das páginas coletadas da Web, encaminhando para o agente *Miner* apenas aquelas cujo conteúdo textual descreva algum produto à venda.

O agente *Crawler* foi desenvolvido e treinado inicialmente pela UNINOVA (Portugal) para realizar os processos de obtenção e de decisão sobre o conjunto de páginas Web européias. Para tal processo de decisão foram utilizados métodos de classificação automática de textos apoiados pelos algoritmos *k-NN* e *C4.5*.

Uma das metas do consórcio DEEPSIA, que está sendo realizada com a participação da USP, é a adaptação do sistema para o Brasil. O resultado dos testes realizados com o sistema nas buscas em sites de venda brasileiros motivou o grupo de pesquisas pela realização de diversos estudos, inclusive para a aplicação de outro método de classificação de textos no processo de decisão do agente *Crawler*.

1.2 Objetivos

O presente trabalho tem como objetivo geral o estudo da classificação automática de textos e a aplicação de seus métodos em conteúdos de páginas Web coletadas de sites brasileiros.

Como objetivo específico buscou-se escolher, aplicar e testar um novo algoritmo de classificação de textos para o problema de decisão enfrentado pelo agente *Crawler* do sistema DEEPSIA sobre o conjunto de páginas Web brasileiras.

Não é o foco deste trabalho apresentar formas de extração de dados de bases textuais. Como resultado final, buscou-se obter um nível satisfatório na precisão quando da classificação de páginas da Web brasileiras, associando-os a duas categorias distintas: documentos que contêm a descrição de produtos à venda e documentos que não a contêm.

1.3 Histórico do Grupo de Pesquisas

O grupo de pesquisas do Laboratório Intermídia do Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo (USP), tem aplicado estudos ao desenvolvimento de projetos em três frentes importantes na área da computação.

A primeira, tendo iniciado seus trabalhos no início da década de 1990 com projetos relacionados ao gerenciamento de redes de computadores [ODA94; CIC94; LIE95; MOR95; MOR97], desenvolve atualmente pesquisas na área de Segurança Computacional com sistemas detectores de intrusões que se utilizam de redes neurais para o reconhecimento de padrões de ataques [CAN97; BON98; TAV99], e aplicam a tecnologia de agentes móveis para gerenciamento da segurança [REA98] e verificação de anomalias [BER99]. Pesquisas sobre ambientes servidores para agentes móveis [FIL01] e tecnologias que podem conferir inteligência a tais agentes [AMB01], e estudos para implementação de sistemas detectores de intrusão em dispositivos de segmentação de redes (switches) [TAV02], foram também desenvolvidos pelo grupo.

A segunda frente de trabalho desenvolve atualmente pesquisas em sistemas multimídia distribuídos, aplicando estudos em padrões para a representação de informações e metadados na identificação de fluxos de mídia contínua e técnicas adequadas para transmissão e distribuição de vídeo na Internet [GOU98].

A partir do segundo semestre de 2001 o Laboratório Intermídia passou a integrar, juntamente com o Núcleo de Manufatura Avançada e amparados pelo CNPq, a iniciativa brasileira pela cooperação com o projeto internacional DEEPSIA, estabelecido por um consórcio entre diversas instituições e empresas européias, com o apoio da *Information Society Technologies* (IST). A participação em tal projeto motivou o grupo pela instituição de uma nova frente de trabalho relacionada ao estudo e ao desenvolvimento de soluções otimizadas para sistemas de Comércio Eletrônico, como a aplicação de técnicas apropriadas à busca, classificação, armazenamento e recuperação de informações de produtos a serem utilizadas em processos de compra na Web. A experiência do grupo em pesquisas na área de segurança computacional também colaborou com estudos na avaliação do sistema DEEPSIA.

1.4 Estrutura da Dissertação

O Capítulo 2 aborda os principais conceitos e estudos atuais na área de aplicação do trabalho proposto, o Comércio Eletrônico. Descrições sobre os modelos de negócios aplicados à Web e sobre a arquitetura do sistema DEEPSIA também são encontradas nesse capítulo.

No Capítulo 3 são apresentados diversos conceitos encontrados na literatura sobre agentes, e são abordados aspectos da utilização de agentes no Comércio Eletrônico.

O Capítulo 4 descreve ontologia e apresenta uma abordagem sobre a sua aplicação em ambientes multi-agentes. Também é apresentado e analisado o modelo UNSPSC para o Comércio Eletrônico, instituído pela ECCMA e adotado pelo projeto DEEPSIA. Uma breve descrição das principais características da Web Semântica encerra o capítulo.

O Capítulo 5 contém a descrição detalhada sobre as técnicas de aprendizado de máquina aplicáveis à classificação automática de textos, bem como a metodologia para o processo geral de classificação de textos, abordando os métodos e algoritmos atualmente utilizados. A descrição detalhada sobre vários aspectos do método SVM, bem como os trabalhos publicados sobre a aplicação da classificação de textos em páginas Web são apresentados na sequência.

No Capítulo 6 encontra-se a descrição completa da solução adotada para a escolha e a aplicação do método de classificação de textos na solução do problema apresentado.

Os resultados obtidos pelo presente trabalho são apresentados no Capítulo 7, já o Capítulo 8 estabelece as conclusões extraídas de tais resultados e de todo o estudo realizado, trazendo também as sugestões para trabalhos futuros.

2 Comércio Eletrônico

Apesar de ter atualmente um significado relacionado aos processos compra e venda pela Internet, o comércio eletrônico (ou *e-Commerce = Electronic Commerce*) consiste, de forma geral, de qualquer transação de compra ou venda, envolvendo bens ou serviços, através de um meio eletrônico. Uma definição mais apropriada à ciência da computação é apresentada em [MEI00] como o “conjunto de técnicas e tecnologias computacionais utilizadas para facilitar e executar transações comerciais de bens e serviços físicos ou virtuais”.

O comércio eletrônico está inserido em um contexto maior que abrange todos os tipos de negócios eletrônicos (ou *E-Business = Electronic Business*). Vários modelos de negócios eletrônicos podem ser definidos, analisando-se a relação entre as partes envolvidas, sendo os tipos seguintes mais identificados nas atividades de comércio eletrônico atuais:

- *Business-to-Business (B2B)*: caracterizam transações bilaterais de negócios envolvendo empresas. O comércio eletrônico em tal modelo pode prover maior integração da cadeia de suprimento, com menor custo para troca de informação entre parceiros e maior velocidade;
- *Business-to-Consumer (B2C)*: definem as relações de negócios entre empresas e consumidores finais. Neste modelo são aplicadas atividades de comércio eletrônico envolvendo a venda direta ao consumidor.

A atividade do comércio eletrônico implica na adoção ou adaptação de diversos processos presentes no modelo de negócios tradicional:

- *Processos de Marketing*: classificam-se neste grupo o fornecimento de informações sobre preço, condições e características de produtos à venda, bem como a administração dos canais de fornecimento e as campanhas de publicidade, como realização de promoção online;
- *Processos de Venda sob Encomenda*: que incluem o funcionamento ininterrupto do negócio online, a manutenção e atualização de catálogos adequados à rápida localização de produtos e a administração geral do negócio (como prazos e logística);

- *Processos de Obtenção e Compra*: combinam ações relacionadas à manutenção de catálogos de fornecedores e à busca de informações neles contidas, para otimizar o processo de compra perante os canais de suprimento;
- *Processos de Distribuição e Logística*: envolvem questões relacionadas à logística e regulamentações para a distribuição de produtos em mídia eletrônica, através da rede;
- *Serviços ao Consumidor*: consistem de canais de atendimento ao consumidor para suportar sua opinião sobre o negócio/produto (críticas, reclamações e sugestões), solicitações de auxílio, acompanhamento de pedidos e devolução de produtos.

2.1 O Comércio Eletrônico na Web

Em 1990, Tim Berners-Lee propôs um novo tipo de aplicação voltada à exibição de documentos [BEL89]. Os hipertextos, dessa forma denominados por associarem referências (ou ligações) a outros documentos no próprio texto, eram exibidos por uma ferramenta gráfica chamada WorldWideWeb, criada anteriormente por Berners-Lee. A “Web”, como passou a ser conhecida tal aplicação, foi a principal responsável pelo enorme interesse pela Internet e seu conseqüente crescimento. Alguns anos depois, quando provedores comerciais passaram a integrar a rede mundial e permitir o acesso de pessoas comuns e empresas [ZAK02], que o uso da Internet literalmente “explodiu”. Documentos de hipertexto foram então produzidos em larga escala pelo crescente número de usuários da Internet, descrevendo em seus conteúdos os mais diversos assuntos. Em estudo recente [AGG01] foi estimada a existência de mais de 1 bilhão de páginas na Web, e uma taxa de crescimento de aproximadamente 1 milhão por dia.

O uso da Web também evoluiu. As instituições, principalmente aquelas que visam o lucro pela venda de bens e serviços, perceberam na Web um veículo eficiente para a divulgação de seus produtos. Esse novo modelo de negócios aplicado à Internet, ou e-Business, tem motivado empresas ao redor do mundo a investirem na adaptação à Web de seus sistemas de informação, para assim disputarem mercados promissores.

Nos últimos anos, pesquisadores atuantes na área de Engenharia de Software aplicaram esforços para a concepção de métodos adequados ao desenvolvimento de sistemas de informação para a Web [ISA95; SCH99; ENG99], instituindo assim uma nova sub-área de estudos, a *Web Engineering*, cujas iniciativas são apoiadas pelo *WWW Consortium*, consistindo temas das *WWW Conferences* mais recentes.

Sistemas de informação para suporte à realização de negócios *online* (*e-Business*) estão em franca expansão na Web. Alguns fatores colaboraram para isso, como a interface agradável e simples das aplicações Web, que possibilita o uso de conteúdo multimídia para realçar a própria apresentação de produtos, e a disponibilidade de acesso à Internet. Tais fatores, associados à atual onda de globalização da economia, tornaram a Web um meio apropriado para a divulgação e comercialização de bens e serviços. A evolução da infraestrutura de apoio, como o suporte logístico, as operadoras de cartão de crédito e a integração dos sistemas bancários, fez com que nichos de mercado, antes inacessíveis por limitações geográficas, pudessem ser alcançados pela vasta abrangência da “rede mundial de computadores”.

Diversos trabalhos foram recentemente publicados, abordando sistemas de informação para *e-Business* em vários aspectos: criação de métodos formais [SON02]; definição de arquiteturas [MAG02] e infra-estrutura [COU02; MIL02]; utilização de agentes [MAG02] e ontologia [HER02]; aplicação de técnicas de recuperação de informações [BAD02].

Existem atualmente muitos tipos de aplicações de negócios na Web. Uma taxonomia de modelos de negócios aplicados à Web [RAP02] é apresentada a seguir.

Brokerage Model – *Brokers* (corretores ou intermediários) são realizadores de negócios, pois colocam compradores e vendedores juntos e facilitam as transações. Tal modelo inclui:

- *Marketplace Exchange* – um grande número de serviços de apoio ao processo de transação para uma determinada indústria ou consórcio. (Exs.: Orbitz, ChemConnect)
- *Buy/Sell Fulfillment* – clientes especificam os pedidos de compra e venda para um bem ou serviço, incluindo detalhes como preço e distribuição, e o corretor cobra uma taxa por transação. (Exs.: CarsDirect, Respond.com)
- *Demand Collection System* – o comprador informa o preço máximo por um bem ou serviço e o intermediário faz a busca. (Ex.: Priceline.com)
- *Auction Broker* – representa negócios para vendedores, cobrando taxas de comissão por transações. (Ex.: eBay)
- *Transaction Broker* – intermedia o processo de pagamento em transações entre compradores e vendedores. (Exs.: ayPal, Escrow.com)
- *Bounty Broker* – oferece recompensas para encontrar pessoas, coisas e idéias. Normalmente cobra uma taxa ou porcentagem sobre os itens encontrados. (Ex.: BountyQuest)

- *Distributor* – realiza serviço de catálogo, conectando um grande número de fabricantes ao varejo. (Ex.: Questlink)
- *Search Agent* – agente (ou “robot”) usado para encontrar preço e disponibilidade de bens e serviços para o comprador. (Ex.: MySimon, DealTime)
- *Virtual Mall* – mercado online virtual, opera como intermediário em transações comerciais, representando vendedores cadastrados. (Ex.: ChoiceMall)

Advertising Model – é uma extensão do modelo de difusão tradicional. Sistemas desse tipo publicam informações sobre bens e serviços de terceiros. Pertencem a este modelo:

- *Portal* – é um ponto de entrada na Web, normalmente um search engine com conteúdo e serviços diversificados. (Ex.: Yahoo!)
- *Personalized Portal* – permite a personalização da interface e do conteúdo. (Ex.: MyYahoo!)
- *Niche Portal* – tem um público-alvo bem definido, (um site que atrai compradores domésticos, ou jovens, p.e.). (Ex.: iVillage.com)
- *Classifieds* – possibilita o anúncio de classificados (itens à venda ou procurados). (Exs.: Monster.com, Match.com)
- *Registered Users* – site que oferece acesso gratuito a conteúdos de interesse, mas condicionados a um registro do usuário. (Exs.: NYTimes Digital)
- *Query-based Paid Placement* – vende a divulgação de sites pela alocação privilegiada de links ou *banners*. (Exs.: Google, Overture, MyWay)
- *Contextual Advertising* – propaganda automática (*banners* e janelas *pop-up*) condicionada ao uso de alguns aplicativos *freeware*. Algumas ferramentas selecionam links baseadas no comportamento do usuário. (Exs.: Gator, eZula)
- *Ultramercials* – propaganda online interativa, que requer uma resposta do usuário antes de exibir o conteúdo desejado. (Ex.: Salon em cooperação com a Mercedes-Benz)

Infomediary Model – mantém dados sobre consumidores e seus hábitos, bem como fabricantes e seus produtos, e atuam como agentes de informação intermediários.

- *Advertising Networks* – serviço que alimenta dados em *banners* para uma rede de sites e coleta informações sobre a eficiência de propagandas, através de *cookies*. (Ex.: DoubleClick)
- *Audience Measurement Services* – agências de pesquisa online sobre o mercado. (Ex.: Nielsen//Netratings)

- *Incentive Marketing* – programa de fidelidade do consumidor. Provê incentivos aos consumidores, como cupons e descontos. (Exs.: Coolsavings, MyPoints, Greenpoints)
- *Metamediary* – facilita transações entre compradores e vendedores, provendo informações sobre produtos. (Ex.: Edmunds)

Merchant Model – atacadistas e varejistas de bens e serviços. Vendas são realizadas através de listas de preços ou consulta.

- *Virtual Merchant* – ou “e-tailer”, mercado que opera sobre a Web. (Ex.: Amazon.com)
- *Catalog Merchant* – serviço de encomendas (carta, telefone e online) apoiado por catálogos Web. (Ex.: Lands’ End)
- *Click and Mortar* – armazéns varejistas tradicionais, mas com uma interface Web. (Ex.: Barnes & Noble)
- *Bit Vendor* – mercado que opera exclusivamente com a distribuição de produtos e serviços digitais através da Web. (Ex.: Eyewire)

Manufacturer Model – um modelo que aproveita o poder da Web para permitir que um fabricante acesse consumidores diretamente, encurtando o canal de distribuição. O contato direto pode ser fundamentado pela eficiência no provimento do serviço ou no melhor entendimento das necessidades e preferências do consumidor. (Ex.: Apple Computer)

- *Brand Integrated Content* – a forma tradicional de propaganda utiliza-se de textos, músicas, filmes e figuras em meios de divulgação maciça. A Web veio permitir que um fabricante produzisse seu próprio comercial, podendo oferecê-lo associado a um conteúdo de interesse (como um filme, p.e.). (Ex.: BMW)

Affiliate Model – em contraste com um portal genérico, que atrai um tráfego enorme para um único site, este modelo provê oportunidade de compra, através de ofertas, nos sites (conveniados) que um usuário está visitando. (Exs.: Barnes & Noble, Amazon.com)

Community Model – baseia-se na lealdade do usuário, que normalmente contribui com o seu conteúdo e/ou dinheiro. O próprio conteúdo e os negócios publicados por seu usuários geram rendimentos para a manutenção do site.

- *Voluntary Contributor Model* – baseia-se na criação de uma comunidade de usuários que mantém o site através de doações voluntárias. Incluem-se no modelo os sites de organizações sem fins lucrativos. (Ex.: The Classical Station (WCPE.org))

- *Knowledge Networks* – sites de especialistas que provêm uma fonte de informações sobre áreas profissionais e outros assuntos. Funcionam normalmente como fóruns de debates e mantém listas de discussões sobre assuntos específicos. (Ex.: AllExperts)

Subscription Model – usuários são tarifados pelo uso de serviços especiais oferecidos em sites. É comum a combinação deste modelo com o *advertising model*.

- *Content Services* – além de periódicos e revistas, a Web tem possibilitado o uso de tal modelo para música e vídeo também. (Exs.: Listen.com, Netflix)
- *Person-to-Person Networking Services* – são canais para a distribuição de informações de usuários cadastrados. (Ex.: Classmates)
- *Trust Services* – um agente independente que intermedia sociedade entre desconhecidos em transações e negócios. Usuários devem concordar com um código de conduta além de pagar a taxa de inscrição. (Ex.: Truste)
- *Internet Services Providers* – oferece acesso a Internet e outros serviços para usuários cadastrados. (Ex.: America Online)

Utility Model – também chamado “sob demanda”, baseia-se na medição do uso dos serviços. Faz alusão aos modelos tradicionais de prestação de serviço (como fornecimento de energia elétrica e telefonia).

- *Metered Subscription* – neste caso, os usuários cadastrados pagam pelo acesso ao conteúdo medido em porções (número de páginas acessadas, p.e.). (Ex.: Slashdot)

Um modelo de negócios comum e de grande sucesso atualmente na Web é o *Marketplace Exchange*. Alguns portais brasileiros, como o *BuscaPé*, o *BondFaro* e o *MercadoBR* mostram-se eficientes na localização de produtos para compradores. O grande problema é que tais portais estabelecem associações com sites de vendedores, mediante contrato, e possibilitam a realização pelos usuários da Internet da busca por produtos oferecidos em um domínio restrito. Esse tipo de solução consiste de um ótimo negócio para vendedores de bens e serviços.

Infelizmente, a limitação do domínio da busca desfavorece um lado envolvido no negócio, o comprador. Em qualquer negócio, a intenção de um comprador (pessoa ou empresa) é encontrar a oferta que melhor lhe favoreça, em uma relação custo X benefício. O modelo *Marketplace* reduz as opções de escolha pelo comprador ao grupo de fornecedores conveniados.

2.2 O Projeto DEEPSIA

O projeto *Dynamic on-line Internet Purchasing System based on Intelligent Agents* (DEEPSIA³ – IST-1999-20483) está sendo desenvolvido no âmbito do programa europeu de pesquisa denominado *Information Society Technologies* (IST). O consórcio DEEPSIA é formado por empresas e institutos de pesquisa de vários países europeus, nomeadamente: empresa ComArch (Polónia), UNINOVA (Portugal), Universite Libre de Bruxelles (Bélgica), University of Sunderland (Inglaterra), empresa Zeus (Grécia) e empresa Comunicación Interativa (Espanha). A USP representa a participação brasileira em tal projeto.

O DEEPSIA visa fomentar a participação das Pequenas e Médias Empresas (PMEs) na área do Comércio Eletrônico. Neste contexto, a abordagem adotada pelo DEEPSIA é desenvolver uma infra-estrutura computacional baseada em um sistema multi-agente que suporte o processo de compra eletrônica (via Internet) efetuado por uma PME, através da criação de Catálogos de Produtos armazenados localmente na própria PME.

A estratégia do DEEPSIA, baseada em uma solução centrada no comprador, visa facilitar a entrada das PMEs no Comércio Eletrônico fornecendo ofertas de produtos a venda adequadas às necessidades específicas de cada PME. A infra-estrutura computacional a ser desenvolvida pelo DEEPSIA oferecerá uma interface amigável baseada em um catálogo personalizado de produtos, o qual será automaticamente atualizado com a informação (sobre os produtos) recolhida dos portais eletrônicos existentes na Internet. O catálogo poderá conter também dados de produtos fornecidos por vendedores específicos cadastrados no comprador. Este catálogo personalizado fornecerá ao comprador um conjunto de ofertas apropriadas em termos de qualidade, diversidade, aplicabilidade e prazos de entrega.

A arquitetura do sistema proposto [GAR01] é composta basicamente de três módulos ou subsistemas, conforme mostrada na Figura 1: *Dynamic Catalogue* (DC), *Portal Interface Agent* (PIA) e *Multi-Agent System* (MAS).

---

³ Endereço do projeto DEEPSIA na Internet: <http://www.deepsia.com/>

Figura 1 – Arquitetura do sistema DEEPSIA

Uma breve descrição de cada subsistema é apresentada a seguir.

- *Dynamic Catalogue (DC)*: Consistirá da interface do usuário, e será responsável pela manutenção e apresentação dos dados coletados pelos multi-agentes contactados, com base nas preferências do usuário fornecidas. Além de tais dados, o sistema possibilitará o acesso a informações de sites visitados, e a configuração da ontologia (representando o perfil individual da expectativa de compra) pelo usuário. A empresa ComArch detém a responsabilidade de seu desenvolvimento;
- *Portal Interface Agent (PIA)*: Será responsável por criar uma interface privilegiada entre os portais da Web e o *Multi-Agent System (MAS)*. O agente atuará como um facilitador ao acesso a dados contidos nos sites e Bancos de Dados de fornecedores. Sua utilização pelos fornecedores (opcional), pode garantir que as informações sobre seus produtos serão acessadas pelo MAS, independente dos resultados das buscas efetuadas pelo *Web Crawler Agent*;
- *Multi-Agent System (MAS)*: Descrito como um sistema autônomo para coleta de dados e um processo semi-automático de atualização do catálogo, composto de um conjunto de agentes, com tarefas específicas. Ao comportamento individual de cada agente será atribuída uma base inicial de conhecimento, que reflete os objetivos estratégicos das PMEs. A responsabilidade de seu desenvolvimento é da UNINOVA. Os módulos internos deste sistema, que caracterizam os tipos especialistas de agentes, são descritos como:

- *Web Crawler Agent (WCA)*: Sua função principal é a busca de páginas Web com dados de interesse pelo usuário, baseando-se em um processo recursivo a partir de uma “semente”, sendo previamente treinado em um processo off-line. Poderá receber informações explícitas do usuário sobre URLs para pesquisa. Executará a primeira seleção e classificação das páginas para posterior processamento do *miner agent*. Para cada página obtida o *WCA* deve decidir se existe a descrição de produto(s) à venda em seu conteúdo textual e, para tanto, utiliza-se de técnicas de aprendizado supervisionado de máquina com os métodos de classificação de textos *k-NN* (*k nearest neighbours*) e *C4.5*;
- *Miner Agent (MA)*: Executará a análise das páginas Web encontradas pelo *WCA* e armazenará as informações relevantes em um Banco de Dados. Ontologias de conceitos (definidas pelo usuário) e um conjunto de regras para relações dos conceitos comporão a base de conhecimento do *MA* para descrição dos dados relevantes, que deverá ser atualizada a cada requisição. Contará também com um processo de validação de alterações das URLs no catálogo, promovido pelo *facilitator agent*, e poderá sobrescrever a entradas de URLs classificadas pelo *WCA*;
- *Human Agent (HA)*: Integrará a interface do usuário para validação sobre recursos e sites pelo *facilitator agent*, e possibilitará também a adição de sites pelo usuário ao *Miner input*. É o único agente a estabelecer contato direto com o usuário, os demais serão executados através da interface com o catálogo;
- *Facilitator Agent (FA)*: Delimitará a interface entre o catálogo e o conhecimento obtido pelo *Miner* e/ou *Collector Agent*. Possibilitará a configuração da periodicidade da produção e atualização de dados para o catálogo, e manterá níveis de confiança sobre as informações que permitem decidir quando validá-las através do *Human Agent*. Receberá solicitações por aquisição de informações do Catálogo e as redirecionará ao *Collector Agent*;
- *Collector Agent (CA)*: Será a interface direta com o Banco de Dados que mantém os sites inscritos pelo PIA. Consistirá de uma aplicação JAVA (JATLite) que utilizará uma metodologia similar ao DNS no DB (compatível com o driver JDBC) para direcionar solicitações do *Facilitator* ao PIA, estabelecendo associações *fornecedores inscritos X usuários compradores*. Receberá mensagens do catálogo com solicitações por atualização e manterá a relação de usuários online, possibilitando também o acesso à informação de fornecedores inscritos.

Outros agentes internos compõem a arquitetura do MAS, executando funções operacionais do modelo e possibilitando sua escalabilidade (existência de vários agentes do mesmo tipo operando colaborativamente) e especialização (possibilitando a existência de agentes especialistas em determinadas características de produtos, regiões geográficas, e outras):

- *Tutor Agent (TA)*: Automatizará o processo de treinamento dos agentes, pelo uso de técnicas de aprendizagem supervisionada. Tais técnicas baseiam-se em mecanismos de *feedback*, promovendo alterações e atualizações nas bases de conhecimento dos agentes;
- *Postman Agent (PA)*: Atuará como um cache para o *miner agent*, com o propósito de aumentar sua eficiência;
- *Pillar Box Agent (PBA)*: Servirá como um repositório inteligente de páginas Web, armazenadas fisicamente em sites ftp associados;
- *Router Agent (RA)*: Será responsável por fornecer a localização de agentes (endereços IP) para a eficiente comunicação entre eles.

O Fluxo de páginas Web:

- 1) Cada vez que o *Crawler Agent* encontrar uma página:
  - a) A página será classificada e enviada ao servidor FTP;
  - b) Uma mensagem KQML será enviada ao *PBA* com sua localização.
- 2) Cada vez que o *PostMan Agent* encontrar uma mensagem:
  - a) Efetuará o download da página e armazenará no *file system* do *MA*;
  - b) Enviará uma mensagem ao *Miner Agent* descrevendo a página e seu endereço.
- 3) O *Miner Agent* poderá *proceder*, de acordo com o resultado da classificação:
  - a) Se for de baixa credibilidade, enviará uma mensagem KQML sobre a página e a própria página para a interface humana (para análise pelo usuário);
  - b) Caso contrário, a informação da página irá para o catálogo diretamente.

A Participação da USP no Projeto DEEPSIA

A USP São Carlos, através do Grupo de Pesquisas em Segurança Computacional do Laboratório Intermídia do ICMC (Instituto de Ciências Matemáticas e de Computação) e do NUMA/EESC (Núcleo de Manufatura Avançada), representa a participação brasileira no projeto, sendo a ele integrada a partir do segundo semestre de 2001 com apoio financeiro do CNPq.

O compromisso do NUMA para o projeto refere-se à realização de “Pesquisa Sócio-econômica” e de levantamento dos “Requisitos dos Usuários”, que descreverão a forma do processo de compra e os tipos de produtos procurados pelas PMEs brasileiras. Com base no estudo, o grupo também definirá e especificará a ontologia dos produtos.

O grupo de pesquisas do ICMC teve como objetivo inicial auxiliar a especificação e a implementação do *Miner Agent* previsto pela arquitetura do MAS. Em uma segunda etapa, baseado nos requisitos levantados, mediante demonstração do sistema e pesquisa efetuada com um grupo de PMEs brasileiras pelo NUMA, aplicou esforços na adaptação do DEEPSIA:

- Tradução da interface e da ontologia para a língua portuguesa;
- Desenvolvimento de novas ferramentas de busca e personalização na interface do sistema.

A observação da versão do sistema desenvolvida pelos parceiros europeus, e a tentativa de operá-la em buscas nos sites de venda brasileiros, deixou clara a necessidade por estudos para a sua melhoria, nos seguintes pontos:

- Processo de decisão do *Crawler Agent* (se o conteúdo da página descreve produto(s) à venda): utilização de outros métodos para a classificação de conteúdos textuais das páginas Web. Os resultados dos testes realizados pelo grupo são apresentados e comparados no Capítulo 6;
- Processo de recuperação de informações do Catálogo: aplicação de técnicas de *data mining* e *data warehousing* para a obtenção de dados históricos e estatísticos sobre produtos armazenados e acesso por usuários do sistema;
- Processo de extração de informações das páginas Web: estudo de técnicas de *text mining* aplicáveis aos conteúdos das páginas Web brasileiras;
- Questões de segurança do sistema: avaliação dos pontos de risco do DEEPSIA em diversos aspectos, desde a comunicação realizada entre os agentes, até a plataforma operacional e a tecnologia utilizada em sua concepção.

2.3 Considerações finais

Neste capítulo buscou-se descrever sucintamente os conceitos básicos envolvidos no Comércio Eletrônico, sua localização histórica e sua inclusão no ambiente Web. Foram também apresentados os principais modelos de negócio eletrônico (*e-Business*) aplicados à Web atualmente, bem como o projeto DEEPSIA, uma solução centrada no comprador, que motivou este trabalho.

A literatura atual aponta para o uso de agentes e ontologia para a integração de sistemas de informação para *e-Business* no ambiente distribuído da Web. Uma descrição sobre agentes e ontologia será apresentada nos próximos capítulos.

3 Agentes

Agentes, agentes inteligentes e sistemas baseados em agentes têm atraído um considerável interesse de muitos campos da ciência da computação [LIN96]. A tecnologia de agentes vem sendo aplicada academicamente nos mais diversos campos, principalmente em inteligência artificial, sistemas distribuídos e engenharia de software, e recentemente em algumas aplicações comerciais, como:

- Suporte para aplicações baseadas em fluxo de trabalho ou *workflow* [NIC98];
- Gerenciamento de redes e serviços de telecomunicações [COR98];
- Análise de informações em aplicações de *data mining*;
- *Layout* de circuitos eletrônicos [MOW97];
- Segurança computacional [REA98];
- Comércio eletrônico [DOO97];
- Computação móvel [LIN96].

3.1 Definição de Agente

Infelizmente não existe um consenso dos pesquisadores sobre a exata definição de agente. Uma definição baseada em Inteligência Artificial é apresentada por Selker: “Agentes são programas de computadores que simulam o relacionamento humano, por fazerem exatamente o que uma outra pessoa faria por você” [SEL94]. Minsk define um agente como “um sistema que pode servir como um mensageiro, pelo motivo de possuir algumas habilidades de especialista” [MIN94]. Para Genesereth & Ketchpel agentes são “componentes de software que comunicam com seus pares por troca de mensagens em uma linguagem de comunicação de agentes” [GEN94].

Em termos gerais, um agente pode ser definido como um software capaz de executar uma tarefa complexa em nome de um usuário [END98].

Franklin e Graesser após fazer várias comparações, definem o conceito de agente autônomo: “Um agente autônomo é um sistema situado em um ambiente, e parte do mesmo, percebendo este ambiente e agindo sobre o mesmo no decorrer do tempo, de acordo com sua própria agenda e de modo a afetar o que ele perceberá no futuro” [FRA96]. Como forma de ajudar na classificação de um agente, Franklin e Graesser ainda apresentam um conjunto de propriedades que podem ser observadas a seguir:

- Reativo (ou percepção e ação): responde conforme as mudanças no ambiente;
- Autônomo: exerce controle sobre suas próprias ações;
- Orientado a metas (com propósito pró-ativo): não age simplesmente em função do ambiente;
- Contínuo: é um processo executado continuamente;
- Comunicativo (socialmente capaz): comunica-se com outros agentes, possivelmente com humanos;
- Inteligente (adaptável): muda seu comportamento com base na sua experiência anterior;
- Móvel: capaz de se mover de uma máquina para outra;
- Flexível: ações não são definidas através de scripts;
- Caráter: personalidade e estado emocional.

Segundo uma definição genérica de Franklin e Graesser, um agente autônomo sempre possui as quatro primeiras propriedades apresentadas. Adicionar outras propriedades incrementa a aplicabilidade dos agentes.

Outros autores, como Genesereth & Ketchpel, argumentam que agentes precisam necessariamente ser comunicativos: “uma entidade é um software agente se e somente se comunicar-se corretamente através de uma linguagem de comunicação de Agentes” [GEN94].

Conforme observado em Reami [REA98], outros trabalhos propõem a definição de um conceito forte de agente. Esta abordagem leva à concepção ou implementação através do uso de conceitos que são normalmente aplicados a seres humanos. Por exemplo, é comum na comunidade de inteligência artificial encontrar agentes caracterizados por suas crenças, desejos e intenções, ou seja, através de uma noção mental de estado. Uma revisão mais ampla das idéias e de sistemas que adotam essa abordagem pode ser encontrada no trabalho de Jennings e Wooldridge [JEN95].

3.2 Tipos de Agentes

Nwana [NWA96] apresenta várias dimensões para classificar os agentes existentes:

Primeiramente, agentes podem ser classificados por sua mobilidade, ou seja, por sua habilidade de mover através da rede de computadores. Desta forma, os agentes podem ser *estáticos* ou *móveis*.

A segunda dimensão refere-se ao modo de funcionamento deliberativo ou reativo do agente. Um agente deliberativo possui um modelo de raciocínio simbólico interno, que lhe permite planejar e negociar tarefas com outros agentes a fim de coordenar uma execução. Agentes reativos, por outro lado, apresentam um comportamento do tipo estímulo/resposta, ou seja, o agente responde ao estado atual do ambiente no qual está imerso, não possuindo o modelo de raciocínio simbólico.

A terceira dimensão envolve classificação de acordo com as diversas propriedades ideais e fundamentais que um agente deveria apresentar. Em seu trabalho, Nwana cita uma lista de três propriedades: autonomia, aprendizagem e cooperação.

A quarta dimensão é relacionada com o papel assumido pelo agente na aplicação, por exemplo, agentes de informação para WWW.

A quinta dimensão é a categoria dos agentes híbridos, que combinam duas ou mais das outras dimensões citadas em um único agente.

Em sua apresentação no mundo real e nas pesquisas, os agentes podem aparecer em várias das dimensões descritas anteriormente. Em seu trabalho, Nwana reduz a combinação dessas várias dimensões, que não seriam possíveis de representação gráfica simples, a uma lista arbitrária, mas que cobre a maioria dos tipos de agentes em investigação atualmente. Desta forma, Nwana identifica sete tipos de agentes:

- Agentes colaboradores – enfatizam autonomia e cooperação com outros agentes para executar tarefas para seus donos. Exemplos desses agentes incluem agentes baseados no paradigma crenças-desejos-intenções (do inglês, *BDI – belief-desires-intentions – agents*, parte importante das linhas de pesquisa em agentes que utilizam conceitos da Inteligência Artificial clássica) sugerido por muitos autores da área de inteligência artificial distribuída;
- Agentes de interface – enfatizam autonomia e aprendizagem e a metáfora chave que dá suporte a esses agentes é a do assistente pessoal que colabora com o usuário em um mesmo ambiente de trabalho. O objetivo principal é migrar da manipulação direta da interface pelo usuário para a delegação de tarefas aos agentes de interface, o que facilitaria a utilização por usuários novatos, uma vez que muitas tarefas são muito complexas ou trabalhosas e exigem treinamento intensivo para sua execução correta;

- Agentes móveis – são processos capazes de “vagar” por grandes redes como a WWW, interagindo com máquinas, coletando informações e retornando após executar os deveres ajustados pelo usuário. Apesar de mobilidade não ser uma condição nem necessária, nem suficiente para o conceito de agente, agentes móveis apresentam uma série de vantagens sobre os similares estáticos, por exemplo: redução dos custos de comunicação, independência da limitação imposta pelo uso de recursos locais, coordenação mais fácil, computação assíncrona, ambiente de desenvolvimento natural para serviços de comércio, arquitetura flexível para computação distribuída e o fato de fornecerem uma nova abordagem atrativa e radicalmente diferente para o processo de projeto de aplicações;
- Agentes de informação (para Internet) – têm o papel de gerenciar, manipular e ordenar informações oriundas de várias fontes distribuídas. A necessidade para agentes deste tipo específico advém da quantidade enorme de informações disponíveis e que precisa ser pesquisada. Agentes deste tipo utilizam-se de outras tecnologias, tais como índices de busca on-line (*Altavista, Yahoo, Google* e outros), para compilar informação sobre algum assunto requisitado pelo usuário e retornar o resultado ordenado para ele;
- Agentes reativos – representam uma categoria especial de agentes que não possuem qualquer forma de representação simbólica interna dos seus ambientes e agem de acordo com o estado atual do ambiente em que estão imersos através de estímulos e respostas. Estes agentes normalmente são simples e se comunicam de maneiras básicas, no entanto, padrões de comportamento complexos emergem da interação entre os diversos agentes. São compostos por módulos autônomos com tarefas bem específicas e lidam com representações próximas do nível físico (dados de sensores) ao invés de representações simbólicas de alto nível. Algumas vantagens seriam robustez, flexibilidade e adaptação em contraposição à inflexibilidade, lentidão e fragilidade dos sistemas inteligentes baseados nas técnicas clássicas;
- Agentes híbridos – esses agentes baseiam-se na hipótese de que maiores benefícios resultam da utilização integrada das várias filosofias existentes. Por exemplo, um agente que implementa tanto o comportamento reativo, quanto o deliberativo, poderia apresentar respostas mais rápidas a mudanças no seu ambiente aplicando o comportamento reativo, enquanto outros problemas orientados a metas e que não exigem tal agilidade poderiam ser tratados pelo comportamento deliberativo;

- Agentes inteligentes – seriam agentes capazes de apresentar todas as características citadas possíveis de autonomia, aprendizagem e cooperação, mas que, na realidade, representam apenas aspiração dos pesquisadores atualmente.

Entretanto, podem ser observadas algumas aplicações que utilizam dois ou mais tipos de agentes. Nwana refere-se a estas aplicações como sistemas de agentes heterogêneos.

3.3 Utilização de Agentes no Comércio Eletrônico

O grande crescimento da Web acarretou um outro problema para o uso de sistemas de Comércio Eletrônico: como localizar precisamente as informações sobre produtos procurados?

A maioria dos agentes de busca, tais como *Altavista*, *Google* e *Yahoo* demonstram grande eficiência na busca por conjuntos de palavras em páginas da Web. O processo normal de busca baseia-se na comparação de informações no nível léxico, obrigando o usuário do sistema a utilizar sua intuição ao fornecer conjuntos apropriados de palavras que possivelmente constarão nas páginas da Web que descreverão o objeto procurado.

Tal prática pode tornar-se frustrante quando um usuário busca por informações qualificadas ou específicas como, por exemplo, um produto que deseja adquirir. Uma forma alternativa de busca tem sido amplamente explorada para esse fim, com a utilização de agentes intermediários, e consiste de sites especializados como *BuscaPé* e *Shopper.com*, que funcionam como portais para outros sites de venda conveniados, ou *Marketplaces*. Além disso, são limitados a domínios de produtos de interesse de usuários individuais (pessoas), como alimentos, vestuário, eletrodomésticos, livros, CD-ROMs e outros. A busca em portais de venda será restrita ao domínio de produtos oferecidos pelos sites de venda conveniados.

A inexistência de soluções aplicáveis às empresas para a localização de produtos à venda como, por exemplo, matéria prima a ser utilizada em seus processos produtivos, motivou a realização do projeto DEEPSIA, baseado em um sistema multi-agentes.

Kephart [KEP99] descreve um tipo de agente, o *shopbot*, como uma família de programas que executam buscas automáticas na Internet por informações sobre preços e qualidades de bens e serviços. O autor afirma que tais agentes são capazes de aprender sobre domínios distintos de produtos e formas de comercialização utilizadas em sites de venda, criando um significado semântico às informações classificadas. Avaliando os resultados apresentados nota-se que, apesar de altamente adaptativos, os agentes necessitam da predefinição e especialização no domínio desejado. Um exemplo de sua aplicação é discutido em [DOO97], onde *shopbots* foram utilizados com sucesso em domínios compostos por softwares e CD-ROMs oferecidos à venda em sites da Web.

3.4 Considerações finais

Este capítulo apresentou os conceitos de agentes obtidos de diversas fontes da literatura, bem como uma análise dos diferentes tipos de agentes, baseada em aspectos diversos. O Comércio Eletrônico, dada a grande dimensão de seu ambiente, a Web, pode valer-se do uso de agentes para a automação de diversas funções, como a busca por informações de produtos à venda.

Infelizmente, os agentes de busca atualmente disponíveis não são capazes de associar descrições textuais das páginas a conceitos de produtos procurados, e trazem volumes enormes de informação pouco significativa. O uso de sistemas multi-agentes que operam conceitos descritos em uma ontologia pode ser a solução parcial para tal problema. Uma revisão sobre ontologia é apresentada no Capítulo 4.

Outro problema que afeta o desenvolvimento de sistemas para busca de produtos é a falta de recursos para a expressão de conceitos na linguagem formal (HTML) de edição de hipertextos. A *Web Semântica*, também apresentada no Capítulo 4, visa solucionar tal problema a médio e longo prazo. Uma alternativa de solução para o momento consiste da aplicação de técnicas de Classificação de Textos para o reconhecimento de padrões textuais em páginas Web que descrevem produtos à venda. Este assunto é o foco principal deste trabalho e será descrito com maiores detalhes no Capítulo 5.

4 Ontologia

No princípio, os computadores foram concebidos como máquinas que processam dados. Os dados expressam estruturas manipuláveis por computadores que não exprimem individualmente qualquer valor semântico, mas representam elementos de um ambiente real ou domínio. Tarefas inerentes à produção de informação a partir do inter-relacionamento de tais dados eram executadas por profissionais especializados em Centros de Processamento de Dados (CPDs).

Em um segundo estágio, sistemas de informação (SIs) foram desenvolvidos visando a automatização do processo de manipulação dos dados para geração da informação. Técnicas aprimoradas de análise permitiram a modelagem prévia de tais sistemas em que requisitos de informação eram traduzidos em processos destinados à obtenção, armazenamento e manipulação de dados. Os sistemas computacionais foram então dotados de interfaces mais adequadas à interpretação humana, permitindo a obtenção de informação por usuários comuns através de funcionalidades previamente analisadas e estabelecidas.

O ser humano encara a informação como a percepção de coisas ou eventos de um domínio do discurso e a associa a conceitos que formam a base de seu conhecimento. Tais conceitos expressam a interpretação das informações e seus relacionamentos. As etapas da geração do conhecimento a partir dos vários níveis de abstração em um domínio são mostradas na Figura 2.

Figura 2 – Abstração de um domínio

O desenvolvimento em massa de SIs individuais ocasionou a geração de enormes quantidades de informações sobre diversos domínios. Individualmente, cada sistema define regras e formas distintas para o acesso à informação. Tal panorama gerou a necessidade da representação abstrata de um nível superior à informação, o conhecimento, independente da forma como a informação é apresentada, mas relacionado ao conceito que ela expressa. A conceitualização sobre domínios específicos do discurso constitui os objetos a serem representados e relacionados em uma ontologia.

Atualmente, muitos esforços estão sendo aplicados na definição de ontologias para os mais diversos domínios, como medicina [ABA00], meio ambiente [PER01], biologia molecular⁴ e comércio eletrônico⁵. Outras organizações trabalham na definição de padrões para:

- Uso de ontologias por agentes inteligentes (FIPA – *Foundation for Intelligent Physical Agents*⁶);
- Representação de conceitos na Web (W3C – *World Wide Web Consortium*⁷).

4.1 Definição de Ontologia

Segundo o dicionário Michaelis da Língua Portuguesa:

“on.to.lo.gi.a *sf* (onto+logo²+ia¹) **1** Ciência do ser em geral. **2** **Filos** Parte da metafísica que estuda o ser em geral e suas propriedades transcendentais.”

Gruber [GRU93] define ontologia como a especificação explícita da conceitualização de objetos. Em sistemas baseados em conhecimento a ontologia é aplicada à representação abstrata dos objetos (ou coisas) e das relações existentes entre eles, definindo um vocabulário de termos da linguagem humana.

Grüninger [GRÜ96] alerta que para a correta utilização de ontologias na representação semântica de objetos em modelagens de problemas e domínios, devem ser aplicados conjuntos apropriados de axiomas na descrição de tais objetos. A axiomatização adequada de objetos, independente de contexto ou língua, pode ser útil para a reutilização dos conceitos descritos pela ontologia, e importante para a comunicação de conhecimentos entre agentes inteligentes.

⁴ <http://boree.cines.fr:8104/textes/IMGTindex/ontology.html>

⁵ <http://www.eccma.org/>

⁶ <http://www.fipa.org/>

⁷ <http://www.w3c.org/RDF/>

Um tutorial sobre Engenharia Ontológica, elaborado por Gómez-Pérez [GÓM99] e disponível no site *Ontology.org*, reúne diversas definições do conceito de ontologia:

- “Uma ontologia define os termos básicos e relações que constituem o vocabulário de uma área do assunto, assim como as regras para combinar termos e relações definindo extensões ao vocabulário” – Neches et al⁸ apud [GÓM99];
- “Uma ontologia é uma especificação explícita de uma conceitualização” – Gruber⁹ apud [GÓM99];
- “Uma ontologia é um conjunto de termos estruturado hierarquicamente para descrever um domínio que pode ser usado como um esqueleto para uma base de conhecimentos” – Swartout et al¹⁰ apud [GÓM99];
- “Uma ontologia provê os meios para descrever explicitamente a conceitualização por trás do conhecimento representado em uma base de conhecimentos” – Bernaras et al¹¹ apud [GÓM99];
- “Uma especificação explícita e formal de uma conceitualização compartilhada” – Studer et al¹² apud [GÓM99].

O autor analisa também, nesta última descrição, a relação individual de alguns termos nela contidos a princípios aplicados à Engenharia do Conhecimento e computabilidade:

- Formal – relaciona-se ao formato legível por máquinas;
- Especificação explícita – está relacionada aos conceitos, propriedades, relações, funções, limitações e axiomas explicitamente definidos;
- Compartilhada – refere-se ao conhecimento adotado de forma consensual;
- Conceitualização – modelo abstrato de algum fenômeno no mundo.

Huhns & Singh [HUH97] apresentam em seu artigo a definição de quatro esquemas de classificação diferentes, cada qual com uma capacidade distinta para a representação da semântica em mensagens entre agentes:

- Palavras-chave (*keywords*): constituem a forma mais simples de localização de informação útil por agentes;

---

⁸ Neches, R.; Fikes, R.; Finin, T.; Gruber, T.; Patil, R.; Senator, T.; Swartout, W.R. *Enabling Technology for Knowledge Sharing*. AI Magazine. Winter 1991. 36-56.

⁹ Gruber, T. *A translation Approach to portable ontology specifications*. Knowledge Acquisition. Vol 5. 1993. 199-220.

¹⁰ Swartout, B.; Patil, R.; Knight, K.; Russ, T. *Toward Distributed Use of Large-Scale Ontologies*. Ontological Engineering. AAAI-97 Spring Symposium Series. 1997. 138-148.

¹¹ Bernaras, A.; Laresgoiti, I.; Corraera, J. *Building and Reusing Ontologies for Electrical Network Applications*. ECAI96. 12th European conference on Artificial Intelligence. Ed. John Wiley & Sons, Ltd. 298-302.

¹² Studer, Benjamins; Fensel. *Knowledge Engineering: Principles and Methods*. Data and Knowledge Engineering. 25 (1998) 161-197.

- Thesauri: um thesaurus oferece uma estrutura mais completa que as palavras-chave para a organização e associação de informações, classificando termos descritivos em categorias distintas, próximas ou relacionadas;
- Taxonomias: provêm estruturas de classificação capaz de representar a hierarquia entre os termos, através de relações de generalização e especialização;
- Ontologias: uma ontologia é um modelo computacional de alguma porção do mundo. A ontologia pode ser fisicamente estruturada como uma rede semântica ou um grafo, onde os nós são conceitos ou objetos individuais e os arcos representam relações ou associações entre os conceitos, conforme exemplo na Figura 3. Propriedades, atributos, limitações, funções e regras são acrescentados a tal rede para a caracterização e descrição do comportamento dos conceitos.

Figura 3 – Exemplo de estrutura de uma ontologia

Os mesmos autores descrevem que, formalmente, uma ontologia expressa um acordo sobre uma conceitualização compartilhada, implementando a estrutura para a modelagem do conhecimento de um domínio e dos acordos sobre a representação de teorias particulares sobre domínios específicos. As definições associam os nomes das entidades de um universo de discurso (por exemplo, classes, relações, funções e outros objetos) a textos em linguagem natural que descrevem o significado dos nomes, e a axiomas formais que delimitam a interpretação.

Huhns & Singh ainda afirmam que, para sistemas de informação, ou para a Internet, ontologias podem ser utilizadas para organizar palavras-chave e conceitos de bancos de dados através da captura das relações significativas entre palavras-chave ou entre as tabelas e campos em um banco de dados. Tais relações oferecem aos usuários uma visão abstrata de um espaço da informação para o domínio de interesse. Ontologias geralmente representam e suportam tipos de relações entre classes significativas, como:

- *Generalização e herança* – expressa a relação entre classes que compartilham similaridades, mas mantêm particularidades específicas. A generalização relaciona uma classe a uma ou diversas outras versões refinadas dela própria. Cada subclasse herda características de sua superclasse e adiciona a si outras características. Um exemplo de generalização pode ser visto na relação “Conversível é um tipo (Subclass) de Carro” da Figura 3;
- *Agregação* – classes que representam objetos são relacionadas como partes de um todo e associadas a uma nova classe, que representará o conjunto como um único objeto. Algumas propriedades da classe composta podem ser propagadas para as classes agregadas (componentes). A relação “Carro possui (hasPart) Roda” da Figura 3 é um exemplo de agregação;
- *Instanciação* – é a relação entre uma classe e cada um dos indivíduos que a constituem.

4.2 Utilização de Ontologia no Comércio Eletrônico

De *Ontology.org* também pode ser obtida uma abrangente motivação para o uso de ontologia no Comércio Eletrônico:

- Espaços de informação em larga escala na Internet serão uma proeminente, senão determinante, base para muitas atividades empresariais, governamentais e pessoais no futuro;
- A padronização de modelos de negócio, processos e arquitetura do conhecimento é crítica para a obtenção do prometido retorno de investimentos do comércio na Internet;
- A principal barreira ao comércio eletrônico encontra-se na necessidade das aplicações em compartilhar informações significativas, e não na confiabilidade ou segurança da Internet. Isto é porque existe uma grande variedade de sistemas empresariais e de comércio eletrônico disponibilizados por negociantes com formas distintas de configuração e uso;
- Problemas de interoperabilidade são particularmente agudos quando um grande número de parceiros comerciais busca concordar e definir padrões interoperacionais, ainda que concorde pela maximização do retorno de investimento.

Embora seja útil esforçar-se pela adoção de um padrão comum sobre domínios específicos para conteúdos e transações, sua obtenção é geralmente difícil, principalmente em iniciativas entre indústrias concorrentes. Em adição:

- Práticas comerciais variam muito e nem sempre podem ser alinhadas, devido a razões técnicas, práticas, organizacionais ou políticas;
- A complexidade da forma como descrever organizações, seus produtos & serviços, separadamente ou combinados, e as interações entre eles é uma tarefa formidável;
- É praticamente impossível estabelecer, a priori, regras (técnicas ou procedimentais) para controlar a participação no mercado eletrônico;
- A adoção de um padrão comum pode impor limites aos modelos de negócio adotados por parceiros comerciais e reduzir potencialmente suas habilidades para a participação plena no comércio eletrônico;

Uma estratégia alternativa é desenvolver ontologias fundamentais compartilhadas, e usá-las como a base para a interoperacionalidade entre parceiros comerciais em mercados eletrônicos. Uma abordagem baseada em ontologia tem o potencial para acelerar significativamente a penetração do comércio eletrônico em setores verticais da indústria, possibilitando a interoperacionalidade no nível de negócio, reduzindo a necessidade por padronização no nível técnico. Isso possibilitará a adaptação de serviços às mudanças bruscas do ambiente online.

Organizações como a ECCMA¹³ (*Electronic Commerce Code Management Association*), fundada em 1999, estão em constante atividade pela elaboração de padrões de conteúdo para o comércio eletrônico na Internet.

A Ontologia ECCMA

A ECCMA definiu o padrão UNSPSC (*Universal Standard Products and Services Classification*) a fim de instituir uma estrutura ontológica para a classificação de bens e serviços (*commodities*).

O consórcio do projeto DEEPSIA adotou o padrão UNSPSC da ECCMA como o modelo de ontologia para a representação e a classificação de bens e de serviços extraídos das páginas Web.

---

¹³ <http://www.eccma.org/>

O UNSPSC apresenta uma classificação hierárquica de quatro níveis. Cada nível contém um valor numérico de dois caracteres e uma descrição textual. Com exceção do nível *Segmento* (*Segment*), que pode ser usado para distinguir produtos de serviços, a ordenação dentro de cada nível não é significativa. A ordem nos níveis *Família* (*Family*), *Classe* (*Class*) e *Mercadoria* (*Commodity*) não é significativa, e a ordem das palavras em um título não implica hierarquia ou importância.

A representação hierárquica do UNSPSC, utilizando os pares de caracteres, é mostrada a seguir:

XX Segmento (*Segment*)

A agregação lógica de famílias para propósitos analíticos.

XX Família (*Family*)

Um grupo reconhecido de categorias de mercadorias inter-relacionadas.

XX Classe (*Class*)

Um grupo de mercadorias que compartilham um uso comum ou função.

XX Mercadoria (*Commodity*)

Um grupo de produtos ou serviços substituíveis.

A Figura 4 apresenta um exemplo de classificação hierárquica segundo o modelo proposto pela ECCMA. As instâncias de classificação são associadas às folhas da estrutura, que é preestabelecida pelo padrão UNSPSC. Interpreta-se, por exemplo, a classificação da primeira instância na figura, como: o *Commodity* é um “Screwdriver”, que é um tipo pertencente à *Class* “Wrenches and drivers”, pertencente à *Family* “Hand tools”, no *Segment* “Tools and General Machinery”.

Figura 4 – Exemplo de classificação no modelo UNSPSC

Alguns problemas podem ser notados no modelo UNSPSC:

- Apesar de suportar o endereçamento de até cem milhões¹⁴ de *commodities* diferentes, a estrutura hierárquica dispõe de um limite máximo de 100 filhos por prole (2 dígitos na chave numérica de identificação), que pode ser insuficiente para categorias populosas;
- O UNSPSC não especifica outros atributos na estrutura, além do próprio nome de cada nó, o que pode ocasionar interpretações duvidosas quando da classificação, conforme exemplo mostrado na Figura 4;
- As únicas relações possíveis de representação no modelo são generalização e herança, o que o torna comparável a uma taxonomia, conforme as definições apresentadas por Huhns & Singh [HUH97].

4.3 Web Semântica (*Semantic Web*)

Durante o evento “XML2000”, Tim Berners-Lee, diretor do *WWW Consortium*¹⁵ anunciou a instituição do projeto Web Semântica. O principal objetivo desse trabalho refere-se à implantação, nos próximos dez anos, de uma nova filosofia para o desenvolvimento e utilização da Web tradicional.

Na Web tradicional os conteúdos de hipertextos fazem sentido à interpretação humana pois são expressos em HTML, que dispõe de recursos para a apresentação dos dados, mas não detém um formalismo apropriado à representação de conceitos manipuláveis por computadores.

Berners-Lee [BEL01] descreve que os padrões XML (*Extensible Markup Language*) e RDF (*Resource Description Framework*) foram então desenvolvidos para o suprimento de tal deficiência. O padrão XML estabelece uma linguagem que possibilita a seus usuários a adição de estruturas arbitrárias através da criação de suas próprias *tags* (etiquetas ou rótulos), mas nada especifica sobre o significado das estruturas.

Já o padrão RDF busca expressar significados sobre coisas ou objetos, codificados por um conjunto de triplas na linguagem XML. Cada tripla consiste de um sujeito (*subject*), um verbo (*verb*) e um objeto (*object*) de uma sentença elementar.

¹⁴ Obs.: quando da elaboração deste trabalho, mais de 16000 mercadorias estavam inseridas no UNSPSC.

¹⁵ <http://www.w3c.org/>

Em RDF, um documento descreve que coisas individuais (pessoas, páginas Web, p.e.) têm propriedades (como “é irmã de” ou “é autor de”) com certos valores (outra pessoa, outra página Web). Sujeitos e objetos são identificados individualmente por um URI (*Universal Resource Identifier*). Os verbos também são identificados por URIs, o que possibilita a definição de um novo conceito, um novo verbo, através da associação de um novo URI em algum lugar na Web.

Outro componente básico da Web Semântica, cuja arquitetura¹⁶ é mostrada na Figura 5, é a ontologia. Como já abordado, uma ontologia pode expressar classes de objetos e as relações existentes entre elas. Classes, subclasses e relações entre entidades consistem uma ferramenta muito poderosa para uso na Web. Agentes inteligentes poderão extrair as regras lógicas do conteúdo das páginas Web, composto de termos expressos em XML, consultando o significado individual e as relações entre eles descritos no vocabulário da ontologia.

Figura 5 – Arquitetura da Web Semântica

Berners-Lee [BEL01] ainda descreve que o poder real da Web Semântica será atingido quando programas que coletam conteúdo Web de fontes diversas processarem a informação e trocarem resultados com outros programas, salientando assim o importante papel dos agentes nas aplicações da Web Semântica.

¹⁶ Fonte: originalmente apresentada no evento “XML2000”

Muitos trabalhos foram desenvolvidos para a definição de aplicações utilizando os padrões propostos pela arquitetura da Web Semântica como, por exemplo, [KLA01] e [CHA01]. Mas o grande problema é que a maioria do atual conteúdo da Web não segue a padronização XML/RDF na composição e representação de seus conteúdos.

4.4 Considerações finais

Neste capítulo foi abordado e desenvolvido o conceito de ontologia e apresentado o modelo geral para a sua representação lógica. Organizações, como a ECCMA, trabalham na definição de ontologias para domínios específicos (no caso da ECCMA, o Comércio Eletrônico).

O padrão UNSPSC da ECCMA, adotado pelo consórcio DEEPSIA, apesar de divulgado como um tipo de ontologia, apresenta características de uma taxonomia.

A Web Semântica, instituída em 2000, visa apresentar uma solução, a médio e longo prazo, para a representação adequada de conteúdos nas páginas Web, sendo estes associados então, a uma ontologia que descreverá regras e relacionamentos para o uso por agentes.

Infelizmente, o conteúdo atual das páginas Web, em sua maioria, é descrito em HTML, o que impossibilita a aplicação dos novos padrões em desenvolvimento pela *WWW Consortium*. Uma alternativa para o problema é o uso de técnicas de Classificação de Textos para o reconhecimento de padrões de conteúdo, e consiste do foco deste trabalho. Uma descrição mais detalhada sobre o assunto é apresentada no Capítulo 5.

5 Classificação de Textos

Conforme apresentado no capítulo anterior, o uso de ontologias para a representação de conceitos pode fornecer uma forma apropriada à manipulação de grandes quantidades de informações distribuídas em sistemas amplos, como a Internet, ou mais especificamente, a Web. O desenvolvimento da Web Semântica, também abordado no capítulo anterior, surge como uma solução de médio a longo prazo (aproximadamente dez anos) para os problemas de representação dos conteúdos da Web, através da aplicação de padrões como o XML e o RDF, possibilitando a operação de conceitos, expressos por classes de uma ontologia, através do uso de agentes inteligentes.

A questão é: *O que se pode fazer para a interpretação semântica do conteúdo atual da Web?* E mais: *Quando a Web Semântica tornar-se realidade, todo e qualquer conteúdo na Web representará conceitos? E para qualquer domínio?*

Atualmente, as páginas Web não possuem uma forma adequada para a representação de conceitos. O atual padrão HTML (*HyperText Markup Language*) limita-se a solucionar a formatação e ligação de documentos, através de uma linguagem formal baseada no uso de *tags* (etiquetas) predefinidas, permitindo a associação de textos e outros elementos na composição de páginas Web, como figuras, filmes, sons e referências a outros documentos. Um hipertexto pode ser dividido em duas partes:

- Uma cabeça (*HEAD*): que entre outras informações de menor relevância, suporta a descrição de um título (*TITLE*), o qual pode expressar o assunto sobre o conteúdo da página. As ferramentas gráficas utilizadas para a visualização de hipertextos normalmente exibem o conteúdo do título no topo da janela de interface com o usuário;
- Um corpo (*BODY*): delimita o conteúdo de uma página Web.

Outra característica da Web atual é que todas as páginas são indexadas por um URL (*Uniform Resource Locator*), que representa um caminho completo para o acesso a um documento (página) na Web. O conteúdo de um URL é textual, e normalmente descreve informações como: protocolo (HTTP, FTP, p.e.) padrão para o serviço a ser acessado; nome de domínio que endereça o *host* da aplicação servidora; endereço (ou número) da porta que referencia a aplicação servidora; caminho no sistema de arquivos; nome do arquivo (ou objeto) a ser acessado, dentre outras.

Portanto, pode-se dizer que existem três componentes textuais em uma página Web que expressam significados a seres humanos: o conteúdo do corpo da página, seu título e o URL que a referencia. Dos três componentes, o único que expressa significado para as aplicações da Web, portanto compreensível por computadores, é o URL. Infelizmente, os conceitos descritos no texto de uma página Web não se relacionam ao URL.

Uma solução seria abstrair os conceitos descritos em conteúdos textuais para um formato compreensível por máquinas, classificando-os corretamente em objetos significativos a humanos. Isto consiste o objeto de estudos da Classificação de Textos (TC – *Text Classification* ou *Text Categorization*¹⁷). Joachims [JOA97] define a classificação de textos como o processo de agrupamento de documentos em diferentes categorias ou classes. Sebastiani [SEB99] associa a classificação de textos à atividade de rotular textos em linguagem natural com categorias temáticas de um conjunto predefinido.

Estudos em TC tiveram início na década de 1960, mas foi a partir da década de 1980 que o aprimoramento das técnicas de programação e o avanço do hardware tornaram possíveis a implementação e o uso em computadores de diversos métodos e algoritmos. Tais métodos, de uma forma geral, implicam em um conjunto de atividades, que visam a obtenção de uma representação apropriada de textos previamente classificados ou rotulados (exemplos), compatível com um método de classificação. Este representará o algoritmo capaz de classificar novas instâncias de documentos textuais, comparando-se à representação obtida para cada classe.

Durante a década de 1980, o principal objetivo em TC foi o estudo de técnicas para a representação manual das regras de composição dos textos, para a correta análise e classificação de novas instâncias de documentos por sistemas computadorizados. Ou seja, visava-se apenas a automatização do processo de classificação a partir de regras previamente e manualmente estabelecidas. O processo que envolve a elaboração de tais regras é muito trabalhoso e depende do conhecimento de especialistas na área do discurso que abrange os conceitos a serem descritos nas classes.

---

¹⁷ Nas diversas fontes consultadas notou-se o uso das duas expressões, ou de uma delas, para denotar o mesmo sentido.

A partir da década de 1990, estudos na área de IA (*Inteligência Artificial*) buscaram formas mais efetivas da aplicação de TC na obtenção de conhecimento expresso em bases textuais. Nesse contexto, o paradigma do Aprendizado de Máquina (AM) foi considerado como uma solução para a automatização dos processos de TC [SEB99]. AM consiste de uma sub-área de pesquisas em IA, e objetiva o estudo de métodos computacionais para adquirir novos conhecimentos, novas habilidades e novos meios de organizar o conhecimento já existente [MON97].

5.1 Classificação Automática de Textos

A classificação automática de textos refere-se à construção automática de classificadores pelo uso de processos indutivos. Ou seja, um processo indutivo, também chamado de aprendiz (*learner*), constrói automaticamente um classificador para uma categoria c_i “observando” as características de um conjunto de documentos, previamente (e manualmente) classificados sob c_i por um especialista no domínio [SEB99]. Tal abordagem denota um tipo de aprendizado de máquina denominado *supervisionado*, onde um documento novo é classificado de acordo com a comparação com um classificador construído e treinado a partir de documentos rotulados.

Vários métodos automáticos de classificação foram desenvolvidos, e podem ser classificados quanto aos paradigmas [MON97]:

- *Simbólico*: caracteriza os sistemas que buscam representar simbolicamente o aprendizado na forma de alguma expressão lógica, árvore de decisão, regras de produção ou rede semântica. Os métodos mais comuns aplicam a indução de classificadores em árvores de decisão, árvores de regressão e regras de decisão;
- *Estatístico*: de forma geral, os sistemas nesse paradigma tendem a focar tarefas em que todos os atributos têm valores contínuos ou ordinais. Aplicam-se aqui também os métodos paramétricos, que visam constituir modelos a partir de parâmetros obtidos da análise dos dados (por exemplo, a constituição de um classificador linear a partir da combinação linear dos atributos). Exemplos de métodos desenvolvidos sob tal paradigma são: *Naive-Bayes*, *Rocchio* e *Support Vector Machines*;

- *Instance-based*: os sistemas nesse paradigma utilizam a “experiência” obtida em classificações de documentos já realizadas para classificar um novo documento, portanto baseiam-se em exemplos de classificação anteriores. A grande maioria dos métodos de classificação automática de textos relaciona-se a tal paradigma, mas o método *k-NN* (*k nearest neighbours*) é o que melhor exemplifica;
- *Conexionista*: qualifica os sistemas que envolvem unidades altamente interconectadas, fazendo uma analogia às conexões neurais do sistema nervoso humano. Os métodos desenvolvidos neste paradigma utilizam-se da aplicação de *redes neurais*, que consistem construções matemáticas simples inspiradas no modelo biológico do sistema nervoso humano.

5.2 Processo geral da Classificação Automática de Textos

O termo *Text Mining* (Mineração de Textos) tem sido utilizado na literatura [WEI98; SEB99] para definir atividades de recuperação e extração de informação (IR – *Information Retrieval* e IE – *Information Extraction*) de bases textuais, fazendo-se uma analogia ao significado do termo *Data Mining* (Mineração de Dados) em relação aos RDBMS (*Relational Data Base Management Systems*). Nesse contexto, TC tem sua aplicação para a identificação e a classificação dos conteúdos descritos nas bases textuais.

[IMA01] descreve de forma geral as atividades que constituem um processo de *Text Mining*. Tal metodologia será apresentada nas próximas seções deste capítulo, onde serão descritas as seguintes atividades:

- Obtenção de documentos;
- Pré-processamento dos documentos;
- Extração do conhecimento;
- Avaliação do conhecimento.

5.2.1 Obtenção de documentos

A obtenção de documentos denota uma atividade pela qual são estabelecidos os meios para a seleção de documentos textuais que constituirão os conjuntos para treinamento, teste e classificação. Conjuntos de treinamento e teste, apesar de utilizados em fases distintas, são basicamente constituídos de pares: documentos (d_i) e classes que os representam (c_i) [SEB99].

A obtenção de documentos textuais contidos em páginas Web deve ser apoiada por ferramentas apropriadas para navegação e busca. Diversas ferramentas como *Web Crawlers*, *Search Engines*, *Subject Directories* e outras podem ser utilizadas nesta atividade.

Web Crawlers são ferramentas (*robots*) que normalmente realizam a busca em profundidade, a partir de um URL dado, perseguindo as referências a outros documentos a partir da página inicial. Alguns critérios, como a restrição da busca dentro de um endereço de domínio da Internet específico e a não exploração de sites que proíbem *robots*, podem ser considerados. Implementações de código livre, como o *WebSPHINX*¹⁸ e guias¹⁹ para a criação de *Web Crawlers* encontram-se disponíveis na Internet.

Algumas ferramentas comerciais, como o *WebSite Extractor*²⁰ e o *Get Right*²¹, possibilitam a busca em profundidade e a reprodução completa dos diretórios e arquivos em um domínio da Internet especificado em um URL.

Como a composição dos conjuntos iniciais de treinamento e testes depende da prévia classificação de cada página, uma busca mais seletiva pode ser obtida com o uso de *Search Engines* e *Subject Directories*.

- *Search Engines* são mecanismos de busca normalmente disponíveis em sites especializados da Internet, como o *Altavista*²², o *Google*²³ e o *Yahoo*²⁴. Tais ferramentas disponibilizam um formulário para que seus usuários submetam conjuntos de palavras, que serão comparadas lexicalmente aos conteúdos textuais de páginas Web armazenadas em seus repositórios;
- *Subject Directories* consistem de uma forma de busca disponibilizada por sites (como os acima mencionados) orientada a diretórios. Tais diretórios destinam a agrupar páginas por assuntos em categorias hierárquicas, e normalmente são organizados com intervenção humana.

5.2.2 Pré-processamento dos documentos obtidos

A próxima etapa relaciona-se ao pré-processamento dos documentos obtidos e visa a extração das características principais de um texto visando sua representação sucinta, de forma manipulável por computadores, a ser utilizada pelos algoritmos de classificação de textos. Duas abordagens podem ser aplicadas na seleção dos termos que representarão o conteúdo textual [SEB99]:

---

¹⁸ <http://www-2.cs.cmu.edu/~rcm/websphinx/>

¹⁹ <http://developer.java.sun.com/developer/technicalArticles/ThirdParty/WebCrawler/>

²⁰ <http://www.esalesbiz.com/extra/>

²¹ <http://www.getright.com/>

²² <http://www.altavista.com/>

²³ <http://www.google.com/>

²⁴ <http://www.yahoo.com/>

- *Sintática*: visa a análise das frases perante a gramática da linguagem utilizada. Tal prática depende da aplicação de algoritmos para a manipulação sintática de textos em linguagem natural, o que na prática não é tão trivial, tornando esta abordagem impraticável para domínios abrangentes;
- *Estatística*: estabelece a análise e a representação de frases através de conjuntos/seqüências de palavras que nelas ocorrem com alta frequência. Tal abordagem é utilizada amplamente pelos métodos atuais de classificação.

A abordagem estatística visa obter uma representação atributo-valor para o conteúdo textual. Essa representação é normalmente referenciada na literatura como *bag of words* [MLA98; SEB99] e descreve a representação vetorial de atributos e pesos a eles associados.

Para a obtenção da representação é necessária a execução de algumas tarefas como identificação dos atributos, atribuição de pesos e redução da representação [IMA01].

Para a tarefa de identificação de atributos algumas operações devem ser realizadas:

- Remoção de *tags* HTML e de códigos de programação (*JavaScript*, p.e.) e substituição de marcadores HTML por símbolos especiais (caracteres acentuados, p.e.). Tal operação é normalmente executada com a ajuda de um *parser* (analisador) léxico;
- Reconhecimento e extração de palavras individuais ou compostas do conteúdo textual. Bibliotecas específicas (*tokenisers*) normalmente são usadas para este fim;
- Uso de uma lista de palavras (*stop list* ou lista de *stopwords*) a serem desconsideradas, como artigos, preposições e outras;

Para a atribuição de pesos aos atributos, são utilizadas duas abordagens de cálculo:

- *Booleano*: os valores 0 e 1 são utilizados para representar, respectivamente, a ausência ou a presença do termo (atributo) no conteúdo textual;
- *Numérico*: representam medidas estatísticas baseadas na frequência dos termos (*TF*) no documento. Uma variação desta abordagem, denominada *TFIDF* (*Term Frequency / Inverse Document Frequency*), sugere a consideração da frequência de cada termo em todos os documentos, como medida inversa da sua capacidade de representar especificamente cada documento [JOA97].

5.2.3 Extração do conhecimento

A extração do conhecimento envolve a aplicação de algoritmos que implementam os diversos métodos de classificação automática de textos. A função principal de tais métodos é a indução automática de classificadores, através do aprendizado de máquina sobre conjuntos apropriados de treinamento.

A seguir, serão apresentados alguns dos métodos mais aplicados atualmente na classificação automática de textos, e suas particularidades [SEB99].

- *Probabilísticos*: classificadores probabilísticos analisam o $CSV_i(d_j)$ (*Categorization Status Value* para o documento d_j) em função da probabilidade $P(c_i/d_j)$ de um documento representado por um vetor $d_j = \langle w_{1j}, \dots, w_{rj} \rangle$ de termos booleanos ou numéricos (*binary or weighted*) coincidir com uma categoria c_i . Tal probabilidade é normalmente computada pela adaptação do teorema de Bayes à classificação de textos, denominada *Naïve Bayes*;
- *Árvores de decisão (Decision trees)*: tal método consiste da criação de uma árvore tal que seus *nós* internos são rotulados pelos termos, os *ramos* que deles partem são rotulados por testes sobre o peso que o termo possui na representação do documento, e as *folhas* são rotuladas por categorias. O processo de classificação de um documento é feito pelo teste recursivo dos pesos dos nós internos em relação à representação no documento. Algoritmos consagrados, como o ID3 e o C4.5, são amplamente utilizados nestes métodos;
- *Regras de decisão*: este método prevê que um classificador seja construído pelo aprendizado indutivo de regras, e consistirá da conjunção de fórmulas condicionais (*cláusulas*) que comporão uma regra lógica na *forma normal disjuntiva* (DNF – *Disjunctive Normal Form*). É também conhecido por ser tão poderoso quanto o método baseado em árvores de decisão;
- *Rocchio*: baseia-se na adaptação da fórmula de Rocchio para a obtenção da relevância em modelos vetor-espço (*Relevance Feedback in the Vector-space Model*) para a classificação de textos. Trata-se de um método clássico que teve origem no tradicional estudo de IR. Nesse método, um vetor protótipo é construído para cada classe c_i e um documento d_j é classificado através do cálculo da distância do vetor que o representa a cada vetor protótipo. O vetor protótipo é construído a partir da média de todos os vetores do conjunto de treinamento para uma determinada classe. O cálculo da distância pode ser efetuado através do produto vetorial ou da medida de similaridade de Jaccard;

- *Redes neurais*: um classificador baseado em rede neural constitui uma rede de unidades, onde as unidades de entrada da rede geralmente representam os termos (atributos), enquanto que as unidades de saída representam as categorias de interesse. Os pesos são indicados nos limites que conectam as unidades e representam as relações de dependência condicional;
- *Baseados em exemplos (Example-based)*: tais classificadores não constroem uma representação explícita da categoria de interesse, mas baseiam-se nos julgamentos utilizados por especialistas para a classificação dos conjuntos de treinamento. Um dos algoritmos utilizados neste método é o *k-NN* (*k nearest neighbours*), que busca classificar um documento pela similaridade a, pelo menos, *k* documentos que foram classificados em c_i no conjunto de treinamento;
- *Support Vector Machines (SVM)*: trata-se de uma abordagem recente introduzida por Vapnik e implementada por Joachims [JOA98], para reconhecimento de padrões em problemas envolvendo duas classes. O SVM consiste de um método definido sobre o espaço vetorial, onde o problema é encontrar uma superfície de decisão que melhor separa os exemplos positivos e negativos, devidamente representados. O processo visa maximizar a margem entre a superfície de decisão e os dados específicos de cada classe no conjunto de treinamento (princípio da minimização do risco estrutural);
- *Comitês (Committees)*: tal método baseia-se na idéia que, dada uma tarefa que requer conhecimento de especialistas para ser executada, *k* especialistas podem ser melhores que um, se os seus julgamentos individuais forem apropriadamente combinados. Vários critérios de decisão podem ser adotados para a combinação, como a *votação majoritária* (MV – *Majority Voting*) e a combinação linear de pesos (WLC – *Weighted Linear Combination*) dos classificadores.

5.2.4 Avaliação do conhecimento

Um fator importante na etapa de avaliação do conhecimento é a interação de todos os elementos participantes do processo, uma vez que ela pode detectar possíveis erros em alguma parte do processo, ou mesmo se a aplicação de outros métodos poderia fornecer resultados melhores. Se fatos como estes forem detectados, algumas das etapas anteriores devem ser retomadas; caso contrário, os resultados serão validados e poderão ser utilizados pelo usuário final em sua aplicação.

Em AM, a avaliação de um modelo de conhecimento é comumente realizada sobre conjuntos de exemplos, chamados conjuntos de teste. Os métodos usados são o *hold-out* e o *cross-validation*. Em ambos os métodos os conjuntos de teste são utilizados para estimar a qualidade média do modelo gerado, através de várias amostragens. No *hold-out* um conjunto de teste de tamanho predefinido é aleatoriamente escolhido, e o restante dos exemplos do conjunto é usado para treinamento do modelo. No método *cross-validation* o conjunto de dados é dividido em tantos conjuntos quanto o número de realizações da fase de teste, e em cada interação um deles é selecionado como de teste e os demais formam o conjunto de treinamento.

Nas tarefas de classificação que envolvem classes de valores discretos, a maioria das estimativas de qualidade pode ser calculada por meio de uma matriz de confusão gerada a partir dos resultados da classificação. Na matriz de confusão, são confrontados os valores reais dos exemplos com os preditos pelo modelo. Na Tabela 1 é apresentado um modelo de classificação bi-valorada, na qual *tp* é o número de exemplos que pertencem à classe positiva e que foram classificados como positivos (positivo verdadeiro), *fp* é o número de exemplos da classe negativa que foram classificados como positivos (falso positivo), *fn* é o número de exemplos positivos que foram classificados como negativos (falso negativo) e *tn* é o número de exemplos negativos que foram classificados corretamente (negativo verdadeiro).

Tabela 1 – Matriz de confusão para um modelo com classe bi-valorada

	Predito como positivo	Predito como negativo
Exemplos positivos	<i>tp</i>	<i>fp</i>
Exemplos negativos	<i>fn</i>	<i>tn</i>

Precision, Recall e Accuracy

A precisão (*precision*) de um modelo é a porção de exemplos positivos que foram corretamente classificados. Para um problema de classe bi-valorada, utilizando a informação da matriz de confusão apresentada na Tabela 1, a precisão pode ser calculada pela Equação 1.

$$Precision = \left(\frac{tp}{tp + fp} \right) \quad (1)$$

A medida *Recall* é definida como a porção classificada corretamente como exemplos positivos. Para o mesmo caso do problema anterior, tal medida pode ser estimada por meio da Equação 2.

$$Recall = \left(\frac{tp}{tp + fn} \right) \quad (2)$$

A medida *Accuracy* denota a proporção de predições corretas, com base na Equação 3.

$$Accuracy = \left(\frac{tp + tn}{tp + fp + tn + fn} \right) \quad (3)$$

As medidas *Precision*, *Recall* e *Accuracy* são bastante utilizadas, principalmente pela simplicidade. Porém, existem situações em que um classificador pode predizer apenas uma classe para qualquer exemplo e assim os valores dessas medidas podem se tornar altos, embora não representem uma boa estimativa. Para contornar esse tipo de problema em IR, é utilizada a medida *F-measuri*.

F-measuri

F-measuri é uma medida que combina *Precision* e *Recall* e adiciona um parâmetro para indicar a importância relativa. Como pode ser observado na Equação 4, o parâmetro que quantifica tal importância é representado por B .

$$F_B(M, c_i) = \left(\frac{(1 + B^2) \times precision(M, c_i) \times recall(M, c_i)}{B^2 \times precision(M, c_i) + recall(M, c_i)} \right) \quad (4)$$

O parâmetro B pode ter valores entre 0 e infinito e é definido pelo usuário. Quando o valor 0 for assumido, F assumirá apenas o valor da medida *Precision*, e se o valor for igual a infinito essa medida assumirá o valor de *Recall*. Geralmente, são usados valores como 0,5, que significa que o valor de *Precision* tem mais relevância que *Recall*; 1,0, que atribui a mesma relevância para ambos; ou 2,0, que indica que *Recall* terá mais relevância que *Precision*. Levando-se em consideração a matriz de confusão mostrada pela Tabela 1, a medida F pode ser estimada pela Equação 5.

$$\left(\frac{(1 + B^2) \times tp}{(1 + B^2 tp + fp + B^2 fn)} \right) \quad (5)$$

5.3 Support Vector Machines

O método *Support Vector Machines* (SVM) foi desenvolvido por Vapnik [VAP95] baseado no princípio da Minimização do Risco Estrutural (*Structural Risk Minimization* – SRM) da teoria do aprendizado estatístico. O SVM é um método utilizado para reconhecimento de padrões definido sobre o espaço vetorial, onde o problema consiste em encontrar uma superfície de decisão que melhor separe os dados em duas classes (positivos e negativos). Pode ser aplicado à solução de problemas de classificação e estimativas de regressão e densidade.

Quando utilizado para a classificação, o SVM admite como entrada conjuntos de exemplos para treinamento e teste, assumidos como “iid” (*independently drawn and identically distribution*),

$$S_n = (x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$$

de tamanho n para uma distribuição $\Pr(x, y)$ desconhecida, que descreve a tarefa de aprendizado. x_i representa o conteúdo do documento e $y_i \in \{-1, +1\}$ indica a classe.

Na sua forma básica, os classificadores SVM aprendem regras binárias lineares

$$h(x) = \text{sign}(w \cdot x + b) = \{+1, \text{ se } w \cdot x + b > 0; -1, \text{ caso contrário}\}$$

descritas por um vetor de pesos w e um ponto inicial b . Dependendo de qual lado do hiperplano o vetor de atributos x incidir, será classificado nas classes $+1$ ou -1 . A idéia da minimização do risco estrutural é encontrar uma hipótese h que possa garantir a menor probabilidade de erro. Para SVMs, Vapnik afirma que o objetivo pode ser traduzido em encontrar o hiperplano h com máxima margem separando os dados de classes distintas (exemplos positivos e negativos) representados no espaço vetorial, isto é mostrado na Figura 6 (obtida de *MLnet.org*). Os exemplos representados juntos ao hiperplano h (marcados com círculos) são chamados *Support Vectors*.

Figura 6 – Representação dos hiperplanos no espaço vetorial

Burges [BUR98] descreve que para problemas de reconhecimento de padrão, SVMs têm sido utilizadas no reconhecimento de caligrafias e de objetos, na identificação de vozes, na detecção de faces em imagens e na categorização de textos [JOA98].

O tutorial de Burges [BUR98] fornece um bom embasamento teórico sobre o uso de SVM para o reconhecimento de padrões, sendo a principal fonte de pesquisas utilizada para a composição da próxima seção.

5.3.1 VC Dimension e Structural Risk Minimization

Dado um conjunto de l observações. Cada observação consiste de um par: um vetor $x_i \in R^n$, $i = 1, \dots, l$ e a verdade associada y_i . No caso de TC, o vetor x_i será composto de palavras e seus respectivos pesos, e y_i será 1, para um exemplo positivo, e -1 para um exemplo negativo de uma classe em questão. Assume-se que exista uma distribuição probabilística desconhecida $P(x, y)$ para qual os dados são representados, ou seja, assume-se que os dados são “iid”.

Agora supõe-se que exista uma máquina cuja tarefa consiste em aprender o mapeamento $x_i \mapsto y_i$. Tal máquina é então definida por um conjunto de possíveis mapeamentos $x_i \mapsto f(x, \alpha)$, assumindo-se ser determinística: para cada x dado, e uma escolha de α , ela sempre fornecerá o mesmo resultado $f(x, \alpha)$. Uma escolha específica de α , indicará o treinamento da máquina.

A expectativa do erro para o teste da máquina treinada é dada pela Equação 6.

$$R(\alpha) = \int \frac{1}{2} |y - f(x, \alpha)| dP(x, y) \quad (6)$$

O risco empírico, definido como a taxa de erro incidente sobre o conjunto de treinamento (para um número fixo e finito de observações) é dado pela Equação 7.

$$R_{emp}(\alpha) = \frac{1}{2l} \sum_{i=1}^l |y_i - f(x_i, \alpha)| \quad (7)$$

Nota-se que nenhuma probabilidade aparece aqui. $R_{emp}(\alpha)$ é um número fixo para uma escolha particular de α e para um conjunto de treinamento específico $\{x_i, y_i\}$.

A quantidade $1/2 |y_i - f(x_i, \alpha)|$ é chamada perda (*loss*), que para este caso, pode assumir apenas os valores 0 e 1.

Vapnik [VAP95] demonstra que, se escolhido η para as perdas, tal que $0 \leq \eta \leq 1$, com probabilidade $1 - \eta$, obtém-se a relação indicada pela Equação 8.

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\left(\frac{h(\log(2l/h) + 1) - \log(\eta/4)}{l} \right)} \quad (8)$$

Onde h é um inteiro não negativo chamado *Vapnik Chervonenkis (VC) dimension*, e é uma medida de capacidade mencionada a seguir. O lado direito da Equação 8 é chamado limite de risco (*risk bound* ou *guaranteed risk*). O segundo termo no lado direito da Equação 8 é chamado *VC confidence*. Tal relação implica que, dado um conjunto de diferentes treinamentos de máquina, e escolhendo-se um valor fixo e suficientemente pequeno para η , dado pela máquina que minimizar o lado direito da equação, escolher-se-á naturalmente a máquina que fornece o limite mais baixo para a expectativa de erro. Esse é o princípio para a escolha de uma máquina treinada para uma certa tarefa, e é a idéia essencial da minimização do risco estrutural.

VC Dimension

VC dimension é uma propriedade de um conjunto de funções $\{f(\alpha)\}$, e pode ser definida por várias classes de função f . Para o caso de reconhecimento de padrões (duas classes) o conjunto de funções é definido por $f(x, \alpha) \in \{-1, 1\} \forall x, \alpha$. Dado um conjunto de l pontos, que podem ser representados em qualquer das 2^l formas, e se para cada representação associa-se um membro de $\{f(\alpha)\}$, pode-se dizer que o conjunto de pontos foi totalmente atingido pelo conjunto de funções. A *VC dimension* para um conjunto de funções $\{f(\alpha)\}$ é definido como o número máximo de pontos de treinamento que pode ser atingido por $\{f(\alpha)\}$.

Structural Risk Minimization (SRM)

Como descrito anteriormente, o termo *VC confidence* da Equação 8 depende da escolha do conjunto de funções, visto que o risco empírico e a expectativa de erro dependem de uma função particular escolhida durante o processo de treinamento. Vapnik [VAP95] descreve que SRM consiste em encontrar o subconjunto de funções que minimize o limite da expectativa de erro. Isso pode ser realizado pelo simples treinamento de uma série de máquinas, uma para cada subconjunto, onde para um dado subconjunto o objetivo do treinamento é simplesmente minimizar o risco empírico. A escolha será feita pela máquina que apresentar a menor soma do risco empírico com *VC confidence*.

A idéia de estrutura vem da forma como é escolhido o subconjunto de funções a partir do conjunto original: dividindo-se o conjunto em subconjuntos aninhados, conforme apresentado na Figura 7 (reproduzida de [BUR98]), onde h_n representa a *VC dimension* de cada conjunto.

Figura 7 – Subconjuntos aninhados de funções, ordenados pela *VC dimension*

5.3.2 SVM na Classificação de Textos

O uso de *Support Vector Machines* para a classificação de textos foi introduzido por Joachims [JOA98]. Em seu trabalho, apresentou um estudo comparativo entre SVM e os algoritmos *Bayes*, *Rocchio*, *C4.5* e *k-NN*, para a classificação de dois conjuntos de testes, representados pelas coleções *ModApte* (extraída da *Reuters*) e *Oshumed corpus*. O resultado médio da medida *Precision/Recall* obtido com o uso do SVM foi superior ao dos demais algoritmos.

O objetivo da classificação de textos é associar documentos a um número fixo de categorias. Para facilitar o aprendizado, cada categoria é tratada como um problema separado de classificação binária [JOA99a].

Cada documento é representado por um vetor atributo-valor. Cada palavra (atributo) corresponde a uma característica com $TF(w_i, x)$, onde *TF* significa a frequência do termo (*Term Frequency*) e é o número de vezes (valor) em que a palavra w_i aparece no documento x .

Joachims justifica a boa performance do método SVM analisando certas propriedades presentes em documentos textuais:

- Espaço dimensional de entrada muito grande: ao treinar classificadores textuais, nota-se uma grande quantidade de características (*features*). SVM utiliza *overfitting protection*, e não depende necessariamente da quantidade de características;
- Poucas características irrelevantes: testes realizados sobre o conjunto da *Reuters* demonstraram que classificadores que utilizam grande parte das características extraídas do texto têm melhor performance do que os que operam sobre um conjunto de características refinado agressivamente por um processo seletivo;
- Vetores de documentos são esparsos: em cada documento, o respectivo vetor contém poucas entradas diferentes de zero;

- A maioria dos problemas de classificação de textos são linearmente separáveis: o que corresponde à forma básica do método SVM.

Um ponto interessante descrito por Joachims [JOA98] refere-se ao critério geral adotado para a seleção de características (palavras) de cada documento textual. A regra geral consiste em considerar como características válidas somente as palavras que aparecerem 3 vezes, pelo menos, no conjunto de treinamento, e que não constem de uma lista de *stop-words* (como *e*, *ou*, etc.).

Outro ponto de destaque no trabalho de Joachims [JOA98] é a afirmação de que o uso de IDF (*Inverse Document Frequency*) para a normalização das dimensões dos vetores de características pode melhorar a performance. O cálculo de $IDF(w_i)$ para uma palavra w_i envolve a prévia obtenção de $DF(w_i)$, que consiste do número de documentos onde a palavra w_i ocorre. Tal cálculo é mostrado pela Equação 9, onde n é o número total de documentos.

$$IDF(w_i) = \log\left(\frac{n}{DF(w_i)}\right) \quad (9)$$

5.3.3 SVM^{light} – Uma Implementação do Algoritmo

SVM^{light} é uma implementação do método *Support Vector Machines* desenvolvida por Joachims [JOA99b] em linguagem C, e disponível, nas plataformas Solaris, Windows, Cygwin e Linux, para o livre uso em pesquisas científicas²⁵.

A versão atual do software (5.00) contempla diversas funcionalidades:

- Otimização rápida do algoritmo: seleção de conjuntos baseada em *steepest feasible descent*; escolha de heurística; *caching* de avaliações do *kernel* e uso de *folding* no caso linear;
- Soluciona problemas de classificação, regressão e *ranking*;
- Calcula a estimativa *XiAlpha* da taxa de erro, da *precision* e do *recall*;
- Calcula eficientemente a estimativa *Leave-One-Out* da taxa de erro, da *precision* e do *recall*;
- Inclui um algoritmo para treinamento de *large transductive SVMs (TSVMs)* [JOA99a];
- Pode treinar SVMs com modelos de custo;
- Manipula milhares de *support vectors*;

- Suporta centenas de milhares de exemplos de treinamento;
- Suporta o uso de funções de *kernel* padrão (polinomial e RBF, para SVMs não lineares), permitindo a definição de um *kernel* próprio pelo usuário;
- Utiliza a representação de vetores esparsos.

O SVM^{light} consiste basicamente de dois módulos: *svm_learn* e *svm_classify*.

svm_learn

É o módulo de treinamento do SVM^{light}. A forma geral de chamada do módulo é:

```
svm_learn [options] example_file model_file
```

Onde:

- *options*: são opções de configuração que alteram o funcionamento do algoritmo em relação à: forma do aprendizado (classificação, regressão ou *ranking*; fator de custo; entre outras), estimativa de performance (*leave-one-out*, *XiAlpha*), operação com *transdction*, tipos de *kernel* (polinomial, RBF) e outras otimizações;
- *example_file*: é o nome do arquivo de entrada, que contém os exemplos de treinamento. A primeira linha do arquivo pode conter comentários e será ignorada, caso inicie com “#”. Cada uma das linhas seguintes representará um exemplo de treinamento. Para o caso de classificação em SVMs lineares o formato de tais linhas será o seguinte (obs.: os pares *feature/value* devem estar dispostos em ordenação crescente em relação ao valor de *feature*):

```
<line> .=. <target> <feature>:<value> ... <feature>:<value>
```

onde:

```
<target> .=. +1 | -1 (exemplo positivo ou negativo)
```

```
<feature> .=. <integer> (índice no arquivo words, descrito abaixo)
```

```
<value> .=. <float> (é o valor ou peso associado à feature; TF, TFIDF, ou outro)
```

Exemplo: indicação de um exemplo negativo que contém as *features* 1, 3 e 9284, com pesos 0.43, 0.12 e 0.2, respectivamente.

```
-1 1:0.43 3:0.12 9284:0.2
```

- *model_file*: é o nome do arquivo de saída, onde o *svm_learn* armazenará o modelo de classificação obtido pelo treinamento sobre os exemplos contidos no *example_file*.

---

²⁵ http://www.cs.cornell.edu/People/tj/svm_light/

svm_classify

É o módulo de classificação do SVM^{light}. A forma geral de chamada do módulo é:

```
svm_classify [options] example_file model_file output_file
```

Onde:

- `options`: são opções de configuração da execução que incluem: ajuda, nível de exibição de mensagens e a escolha do formato da saída;
- `example_file`: é o nome do arquivo de entrada, que contém os exemplos de teste, de formato idêntico ao arquivo de entrada do `svm_learn`;
- `model_file`: é o nome do arquivo com o modelo de classificação obtido no treinamento realizado pelo `svm_learn`;
- `output_file`: é o nome do arquivo de saída onde serão armazenados os valores da função de decisão para cada exemplo classificado (um por linha, na mesma ordem dos exemplos, no arquivo de entrada). Para o caso de classificação, o sinal (positivo ou negativo) de cada valor indicará a predição da respectiva classe.

O `svm_classify` exibe, como resultado em *stdout*, a medida *accuracy* obtida na classificação do conjunto de testes.

words

Trata-se de um arquivo texto contendo todas as palavras relevantes (que apareçam pelo menos 3 vezes em todo o conjunto de documentos amostrados, e que não constem da lista de *stop-words* [JOA98]) que compõem o dicionário para o domínio das classes a serem analisadas. Cada linha no arquivo deve conter uma palavra inédita, que será referenciada pelo SVM^{light} como uma *feature* distinta, através de um índice numérico inteiro que corresponde ao número da linha no arquivo.

5.4 Aplicação de TC na classificação de páginas na Web

Vários autores descrevem estudos realizados nos últimos anos, envolvendo o uso e a avaliação de métodos de classificação automática de textos em conteúdos de páginas Web, dos quais destacam-se:

Mladenic [MLA98] utiliza e compara em seus experimentos métodos de aprendizado probabilísticos (*Naïve Bayes*) e baseados em exemplos (*k-NN*) na classificação de assuntos em páginas Web visitadas por usuários. Seu trabalho foi inspirado na ferramenta *WebWatcher*.

Nigam [NIG99] descreve em seu trabalho a aplicação de um método probabilístico (*Naïve Bayes*) e de uma versão alterada deste, pela utilização de técnicas de EM (*Expectation-Maximization*), para a classificação de documentos rotulados e não rotulados (referindo-se à supervisão do método de aprendizado) consistidos de páginas Web.

Joachims [JOA98] descreve a utilização do método SVM para a classificação de páginas Web em um número predeterminado de categorias e apresenta um estudo comparativo de sua eficiência perante os algoritmos *Naïve Bayes*, *Rocchio*, *C4.5* e *k-NN*.

Wang & Hu [WAN02] utilizaram SVM para o reconhecimento de padrões de tabelas em páginas Web.

Craven [CRA99] provou que é possível reconhecer conceitos de objetos de uma ontologia em descrições textuais de páginas Web, através do uso de aprendizado indutivo sobre exemplos de tais descrições. Utilizou-se de métodos probabilísticos (*Naïve Bayes*) para a classificação dos conceitos nas categorias da ontologia, e da análise de *hyperlinks* para descrever as relações existentes entre as classes. Apesar de funcional, o experimento realizado por Craven limitou-se a um domínio específico (páginas que descrevem setores, funcionários, professores, aulas e alunos de uma universidade) e valeu-se da organização já existente dos assuntos em tais páginas.

5.5 Considerações finais

Este capítulo abordou o assunto Classificação de Textos, e contém uma descrição detalhada sobre as técnicas de aprendizado de máquina aplicáveis à classificação automática de textos. Uma metodologia para o processo geral de classificação de textos foi apresentada, contendo uma descrição dos métodos e algoritmos atualmente utilizados.

Atenção especial foi dada ao método SVM, cuja descrição detalhada abordou aspectos sobre sua teoria, representação e uso da implementação SVM^{light}.

Estudos envolvendo a aplicação da classificação de textos em páginas Web são mencionados no final do capítulo.

6 A Solução Adotada

Como apresentado no Capítulo 2, o Comércio Eletrônico consiste de uma atividade de grande expansão atualmente na Web. Muitos modelos de negócio eletrônico foram desenvolvidos para operação em tal ambiente, mas nota-se que os de maior sucesso e também mais comuns, como os *Market Places*, favorecem o lado do vendedor na transação, dada a restrição do domínio da busca por produtos a um grupo de fornecedores cadastrados.

Os agentes de busca como *Yahoo* e *Google*, apresentados no Capítulo 3, estabelecem comparações léxicas das palavras fornecidas pelos seus usuários com os conteúdos textuais de páginas Web. O uso de tais mecanismos para tarefas relacionadas à pesquisa por produtos à venda pode se tornar frustrante ao usuário interessado, uma vez que ele necessitaria prever as possíveis palavras constantes das páginas pretendidas.

A Web Semântica, instituída em 2000, aparece como uma solução futura para tal problema. Evoluções na linguagem formal de edição de hipertextos, a exemplo dos padrões XML e RDF, permitirão a expressão de conceitos de objetos e suas relações no conteúdo das páginas Web. Esses conceitos serão devidamente classificados em uma ontologia a ser compartilhada por agentes na Web.

O projeto DEEPSIA, apresentado no Capítulo 2, visa o desenvolvimento de uma solução centrada no comprador para a busca por produtos à venda na Web. Como também descrito, os testes realizados com o sistema DEEPSIA na localização de produtos à venda em páginas brasileiras denotaram a necessidade, dentre outras, da avaliação de outros métodos para suporte à decisão do *Crawler Agent*. Tal agente assume um papel importante no sistema, pois realiza a busca de páginas Web (*crawling*) e decide, através de técnicas de aprendizado de máquina, se contém a descrição de produtos à venda, ou não.

No Capítulo 5 foram descritos os principais conceitos envolvidos na aplicação de métodos de classificação automática de textos, suportados por técnicas de aprendizado de máquina. Muitos dos métodos apresentados são referenciados na literatura atual em estudos envolvendo a classificação de conteúdos de páginas Web.

Baseando-se no exposto, o presente trabalho visa a aplicação de um dos métodos descritos para a classificação de páginas Web, baseada em seus conteúdos textuais, como suporte ao problema de decisão do *Crawler Agent* do sistema DEEPSIA.

6.1 Metodologia

A metodologia aplicada à solução do problema envolveu as seguintes etapas, as quais são descritas em detalhes nas próximas seções:

- Análise do problema;
- Escolha do método de classificação de textos;
- Desenvolvimento dos modelos de representação;
- Amostragem do conjunto problema;
- Aplicação do método e geração de resultados parciais.

6.1.1 Análise do problema

O conjunto que descreve o problema estudado neste trabalho é composto pelas páginas Web de sites brasileiros. Uma característica comum notada nesse conjunto é o uso do HTML para a composição do conteúdo das páginas.

Uma página Web escrita em HTML, como descrito anteriormente, possui geralmente os componentes *HEAD* e *BODY*, delimitados por *tags* com os respectivos nomes, conforme mostrado na Figura 8.

Figura 8 – Exemplo de código HTML

O *HEAD*, contém informações sobre o tipo de conteúdo da página (*metadados*), que são úteis para a sua interpretação pelos Web browsers, das quais o título (*TITLE*) representa maior significado para o usuário. O título da página é exibido pelo Web browser no topo da janela gráfica, conforme exemplo na Figura 9.

Figura 9 – Código HTML interpretado pelo browser

No *BODY* é descrito todo o conteúdo da página Web que será exibido ao usuário, conforme apresentado na Figura 9. Tal conteúdo pode apresentar, além dos diversos *tags* HTML de formatação e ligação, outros que são utilizados para descrição de código JavaScript (*SCRIPT*), definição de estilos de formatação (*STYLE*) e comentários que não fazem parte do hipertexto (`<!-- -->`), conforme o exemplo na Figura 8.

Portanto, o conteúdo textual que distingue as páginas Web é delimitado pelos *tags* de formatação e ligação, e pelo *tag TITLE*. Isso pode ser claramente observado no resultado da interpretação do código HTML apresentado ao usuário pelo Web browser, conforme apresentado na Figura 9.

Dado o problema de decisão do *Crawler Agent* pode-se subdividir o conjunto composto pelas páginas Web brasileiras em dois subconjuntos:

- Conjunto de exemplos positivos: tal conjunto é formado pelas páginas Web brasileiras que contêm a descrição textual de produto(s) à venda. Uma característica notada com grande frequência nos elementos de tal conjunto é a presença de sentenças que exprimem valores monetários (preços) dentre outras palavras comumente utilizadas em seus conteúdos. Um exemplo positivo pode ser observado na Figura 10;
- Conjunto de exemplos negativos: este conjunto é composto das páginas Web brasileiras que não contêm a descrição textual de produto(s) à venda. Um elemento de tal conjunto é apresentado na Figura 11.

Figura 10 – Exemplo positivo

Figura 11 – Exemplo negativo

Portanto, observa-se um caso típico de classificação binária (positivos/negativos) para uma classe única. Ou seja, uma página Web pode ser predita como um exemplo positivo se contiver um padrão de descrição textual associado à classe em questão, e como um exemplo negativo caso contrário.

6.1.2 Escolha do método de classificação de textos

Baseando-se na afirmação de que as páginas Web podem ser classificadas em um dos tipos acima mencionados através da observação de seus conteúdos, pode-se dizer que é viável a obtenção de conjuntos de documentos rotulados para cada caso. Isso torna possível a aplicação da classificação automática de textos pela indução de classificadores a partir do aprendizado de máquina supervisionado.

Tal técnica é também aplicada na concepção de métodos sob o paradigma estatístico que estão entre os mais utilizados atualmente para problemas de classificação de textos, conforme mencionado no Capítulo 5, o que sugere a adoção de um deles para a solução do problema apresentado.

Para este trabalho decidiu-se pela aplicação do método *Support Vector Machines* (SVM) que não havia sido testado para o problema de decisão do *Crawler Agent*. A aplicação e a avaliação de tal método sobre o problema em questão definiram os objetivos específicos do trabalho. Outros motivos influenciaram tal escolha (descritos no Capítulo 5):

- Adequação da aplicação do método para o problema de classificação exposto;
- Casos de sucesso e bom desempenho, quando da comparação de diversos métodos na solução dos mesmos problemas, citados na literatura;
- Grande quantidade de material literário sobre os conceitos envolvidos e a aplicação prática do método;
- Implementação do algoritmo SVM bem documentada e de código aberto, disponível para o uso em pesquisa científica.

6.1.3 Desenvolvimento dos modelos de representação

Para a concepção do modelo geral da solução a ser implementado, primeiramente buscou-se analisar o formato dos elementos dos conjuntos a serem amostrados, comparando-o ao formato de representação dos dados de entrada da implementação SVM^{light}. Em uma segunda etapa, o processo geral de classificação de textos, descrito no Capítulo 5, foi adaptado para suportar os requisitos obtidos na análise citada.

Os elementos pertencentes ao conjunto problema constituem, como já descrito, de páginas Web de sites brasileiros, que são compostas por código HTML e conteúdo textual expresso em língua portuguesa, como representado na Figura 12(a). Cada exemplo, obtido de tal conjunto pela referência ao seu URL, constituirá de um arquivo HTML distinto.

O algoritmo SVM^{light} requer, como entrada nos casos de treinamento e teste, a representação do conteúdo textual de cada exemplo (positivo ou negativo) no formato de uma linha composta de um conjunto de pares $\langle feature \rangle : \langle frequency \rangle$, com indicação do tipo do exemplo (1 para positivo e -1 para negativo) logo na primeira posição. Onde cada *feature* representa uma palavra significativa do texto e *frequency* é o valor da frequência (TFIDF) de tal palavra. A representação do arquivo texto de entrada, contendo tantas linhas quantos forem os exemplos, é mostrada na Figura 12(b).

Figura 12 – (a) Conjunto das páginas brasileiras, e (b) arquivo de entrada do SVM^{light}

Ficou clara e evidente a necessidade da implementação de mecanismos que possibilitassem a obtenção dos exemplos, o armazenamento organizado destes e a transformação entre os modelos de representação distintos.

Esta parte do trabalho apoiou-se na adaptação das etapas de obtenção e de pré-processamento dos documentos, abordadas no processo geral de classificação de textos (Capítulo 5). Atividades de análise e de modelagem foram aplicadas na solução dos problemas envolvidos em cada etapa, e são descritas a seguir:

Obtenção de documentos

O principais problemas enfrentados nesta etapa relacionam-se à recuperação das páginas Web, indexadas por URLs, e o armazenamento apropriado delas em repositórios locais, destinados à separação dos dois tipos de exemplos: positivos e negativos. Uma representação geral de tais procedimentos é encontrada na Figura 13.

O URL de uma página não tem significado local, portanto uma estratégia distinta de indexação foi utilizada, onde cada exemplo é armazenado em um arquivo, cujo nome segue a associação seqüencial de números naturais (1, ... , n) mais a extensão “.html”. O URL é armazenado, tanto na primeira linha do arquivo (no formato de comentário HTML), como em um arquivo índice (“index”), juntamente com o número que representa o nome do arquivo HTML.

Utilizando de recursos do próprio sistema de arquivos, e visando-se viabilizar a obtenção de exemplos para categorias distintas, foi adotada uma solução que consiste em armazenar separadamente os exemplos em diretórios nomeados pela categoria. Os exemplos positivos e negativos são indexados e armazenados em subdiretórios distintos, “positive” e “negative”, respectivamente.

Figura 13 – Obtenção de documentos

A solução para tais problemas foi obtida pela concepção da classe *DocumentExample*, mostrada na Figura 14. Tal classe, instanciada pelos parâmetros “nome da categoria” e “localização do diretório”, administra os repositórios de exemplos para as categorias, criando os diretórios ou simplesmente abrindo-os, caso existam. Isso possibilita realizar parcialmente a tarefa de obtenção de documentos.

Cada página Web é obtida pela chamada do método *insertExample*, tendo como parâmetros a respectiva URL e o tipo de exemplo (“positive” ou “negative”). Para executar tal tarefa, a classe *DocumentExample* obtém uma instância da classe *WebPage*, concebida especialmente para tal, conforme representado na Figura 14.

As exceções obtidas quando da realização de tais procedimentos são encapsuladas pela classe *DocumentExampleException*.

Figura 14 – Classe *DocumentExample*

Pré-processamento dos documentos

Nesta etapa buscou-se obter soluções para a extração do conteúdo textual das páginas Web e a sua transformação para um modelo de representação adequado e requerido pelo algoritmo SVM^{light}.

Conforme abordado no Capítulo 5, os seguintes procedimentos descrevem as principais atividades necessárias à solução dos problemas acima citados:

Parsing:

O principal objetivo deste procedimento é a extração do conteúdo textual do documento, pela correta interpretação do código HTML, implicando tarefas de:

- Remoção de *tags*, comentários, definições de estilo (*STYLE*) e código JavaScript (*SCRIPT*);
- Interpretação de códigos utilizados na representação de caracteres especiais (ISO-8859-1).

Objetivando a consistência da indexação dos exemplos e a disponibilização do resultado do processo para as demais etapas, optou-se por armazenar o conteúdo textual obtido em um arquivo de extensão “.txt” no mesmo diretório e com o mesmo nome arquivo HTML, conforme indicado na Figura 15.

Figura 15 – O processo *parsing*

A solução para o *parsing* foi fornecida pelo projeto da classe *HtmlParser*, a qual dispõe de métodos adequados ao tratamento de *tags* e à extração do conteúdo textual, e é mostrada na Figura 16. Para o problema da interpretação dos códigos de caracteres especiais, foi desenvolvida a classe *Charset*, que contém a associação dos códigos ISO-8859-1 para todos os caracteres especiais, encontrados em páginas Web brasileiras. A classe *HtmlReader* foi concebida da necessidade pela realização da leitura seqüencial, caracter por caracter, do arquivo HTML armazenado. A fim de reduzir a diversidade de formatos das palavras no texto, todos os caracteres são transformados em seus respectivos “minúsculos” (quando aplicável) antes do armazenamento.

Para facilitar o processo geral, o *parsing* é efetuado em cada página, sendo acionado após a obtenção pela classe *DocumentExample*, conforme representado na Figura 14.

Figura 16 – Classe *HtmlParser*

Tokenising:

Por este processo visa-se reconhecer no arquivo texto cada palavra significativa que poderá compor a representação estatística do exemplo. Uma palavra é considerada significativa se não contiver números e não constar de uma *stop list*. Como o objeto de estudo deste trabalho são descrições de produtos oferecidos à venda, é desejável reconhecer a presença de sentenças numéricas que indiquem a descrição de preços.

Procedimentos especiais também devem ser elaborados para a retirada de caracteres que não representam letras (“parênteses”, “aspas”, pontuações, números e outros) dos extremos das palavras.

A Figura 17 representa o processo, de forma geral.

Figura 17 – O processo de extração de palavras

A classe *TextTokenizer* (Figura 18) foi projetada para o processo de extração sequencial das palavras significativas do conteúdo textual. Através do método *getNextWord* obtém-se a próxima palavra do arquivo texto, que é:

- Validada quanto à presença na *stop list*: procedimento este executado pelo método *isValidWord*, utilizando a instância de uma classe *StopList*, que contém a lista de palavras reservadas;
- Comparada ao formato de um preço: através do método *isPrice*, e em caso positivo, uma palavra comum (“pricestring”) identificará cada ocorrência e será retornada pelo método *getNextWord*;
- Processada, para a retirada de caracteres que não representam letras (“parênteses”, “aspas”, pontuações, números e outros) de seus extremos, através do método *handleWord*.

Figura 18 – Classe *TextTokenizer*

Composição do dicionário:

A aplicação de métodos estatísticos abordada requer a representação do texto como um conjunto de palavras e seus respectivos pesos. Para limitar o número de palavras encontradas no domínio estudado faz-se necessária a definição de um vocabulário, ou dicionário, específico, obtido sobre a amostragem do conjunto problema.

A composição de tal dicionário deve ser realizada durante o processo de aprendizado supervisionado ou, mais especificamente, na etapa de treinamento do algoritmo. O motivo disso relaciona-se ao fato de que as amostras utilizadas para o treinamento são previamente classificadas pela entidade humana como partes integrantes do conjunto problema.

A fim de se compor um dicionário, e também atender os requisitos do algoritmo SVM^{light} , foram notados os seguintes aspectos nesta fase:

- O dicionário deve ser gerado a partir das amostras de treinamento, ou seja, toda vez que o conjunto de treinamento for alterado um novo dicionário deve ser construído;
- Devem participar do dicionário todas as palavras válidas (regra descrita no processo de *tokenising*) que aparecerem em 3 documentos (pelo menos), conforme sugestão do autor do algoritmo, mencionada no Capítulo 5;

- As palavras do dicionário serão armazenadas, uma por linha e pela ordem alfabética crescente, em um arquivo denominado “words”. A posição da palavra no arquivo, dada pelo número da linha, será utilizada como identificação quando da representação vetorial.

A classe *Dictionary*, ilustrada pela Figura 19, foi projetada para atender tais requisitos. Através de instâncias de *TextTokenizer*, todas as palavras válidas constantes das amostras de treinamento são obtidas, sendo consideradas aquelas que, ao final, tenham aparecido em pelo menos três exemplos. O método *hasWord* retorna a informação sobre a presença, ou não, de uma palavra no dicionário.

Figura 19 – Classe *Dictionary*

Atribuição de pesos às palavras:

Esta é uma das atividades mais complexas, e não menos importante, de todo o processo de classificação. Como descrito no Capítulo 5, existem formas distintas para a atribuição de pesos às palavras que compõem os exemplos. Decidiu-se pela utilização da unidade de medida TFIDF neste processo, devido às recomendações constantes na literatura pesquisada, e conforme já mencionado.

O objetivo aqui perseguido constituiu-se da obtenção do modelo final de representação das amostras. O formato desejado, como já descrito anteriormente e ilustrado pela Figura 12, consistirá de um vetor onde:

- O primeiro elemento (1 ou -1) indicará o tipo de exemplo em questão (positivo ou negativo);

- Os elementos subsequentes constituem-se de pares que descrevem as palavras válidas (e constantes do dicionário, sendo indicadas pelo valor numérico correspondente no índice) encontradas no exemplo, com seus respectivos valores expressos em TFIDF.

O conteúdo integral de cada vetor é armazenado em uma linha do arquivo que servirá de entrada para os processos de treinamento (“train.dat”) e teste (“test.dat”) do algoritmo *SVM^{light}*.

Para realizar tais procedimentos a classe *FeaturesVectors* (Figura 20) foi concebida. Os métodos *generateTrainVectors* e *generateTestVectors* são responsáveis pela geração dos arquivos de entrada do algoritmo descritos. Essa classe também é responsável pelo cálculo do valor TFIDF de cada palavra, obtido pela multiplicação do valor de TF (*term frequency*), fornecido pela instância da classe *FeaturesFrequencies*, pelo valor IDF (*inverse document frequency*), provido pelo uso da classe *FeaturesIDF*.

Figura 20 – Classe *FeaturesVectors*

A classe *FeaturesFrequencies*, mostrada na Figura 21, realiza a leitura de cada exemplo através de um *TextTokenizer*, e compõe um conjunto (*Set*) de duplas “*feature:frequency*” que é retornado pelo método *getAllFeatures*. Cada *feature* é representada pelo seu número no índice do dicionário, fornecido pela classe *Dictionary*, e a respectiva *frequency* que indica o número de vezes que foi notada no exemplo.

Figura 21 – Classe *FeaturesFrequencies* Figura 22 – Classe *FeaturesIDF*

O conjunto (*Set*) obtido pela classe *FeaturesVectors* para cada exemplo é fornecido à classe *FeaturesIDF*, através do método *insertFeatures*. A classe *FeaturesIDF*, mostrada na Figura 22, calcula o valor IDF para cada palavra do dicionário, armazenando tal associação em um arquivo denominado “words.idf”. O processo do cálculo do IDF segue os passos:

- Primeiramente, a classe *FeaturesIDF* providencia o cálculo do valor DF (*document frequency*) para todas as palavras do dicionário, através de uma instância de *FeaturesDF* (Figura 23). Todas as *features* contidas em cada *Set* são contadas pela chamada do método *InsertFeature* da classe *FeaturesDF*;

- Em seguida, o valor IDF de cada palavra é calculado, podendo ser obtido pelo método *getFeatureIDF*. O cálculo envolve a aplicação da Equação 9, apresentada no Capítulo 5, para os valores DF de cada palavra, obtidos pela chamada ao método *getFeatureDF* da classe *FeaturesDF*.

Figura 23 – Classe *FeaturesDF*

6.1.4 Amostragem do conjunto problema

Esta etapa envolveu a busca e a coleta de amostras do conjunto problema, constituído pelo domínio das páginas Web dos sites brasileiros. Tais amostras, conforme descritas anteriormente, constituirão os conjuntos de exemplos positivos e negativos para as tarefas de treinamento e de testes do algoritmo SVM^{light}.

Para que se garantisse a consistência dos conjuntos de exemplos, cada amostra coletada foi previamente analisada com a utilização dos recursos de visualização presentes nos browsers Web. A tarefa de observação possibilitou a predição de cada amostra em constituir-se de um exemplo positivo ou negativo, em relação à presença de produto(s) à venda.

Os conjuntos de exemplos também foram agrupados para cada site pesquisado, a fim de se poder avaliar a qualidade de seus conteúdos na composição do modelo de treinamento. As amostras foram coletadas de 13 dos principais sites de venda de produtos brasileiros e 1 site de notícias econômicas. A descrição dos sites, bem como os totais de exemplos positivos e negativos coletados são apresentados na Tabela 2.

Tabela 2 – Conjuntos de exemplos amostrados

Nome do conjunto	URL do site	Positivos	Negativos
Americanas	www.americanas.com.br	50	47
Bondfaro	www.bondfaro.com.br	40	50
Buscape	www.buscape.com.br	20	19
EstadaoEconomia	www.estadao.com.br/economia	0	25
Extra	www.extra.com.br	33	25
GloboShopping	globoshopping.globo.com	21	2
Kalunga	www.kalunga.com.br	54	17
LivCultura	www.livcultura.com.br	33	33
LivSaraiva	www.livrariasaraiva.com.br	51	46
Luiza	www.magazineluiza.com.br	66	91
MercadoLivre	www.mercadolivre.com.br	50	71
ShoppingUOL	shopping.uol.com.br	27	10
Shoptime	www.shoptime.com.br	36	26
Submarino	www.submarino.com.br	45	42
Totais de exemplos:		526	504

Outros dois conjuntos de exemplos foram obtidos da análise das amostras coletadas e classificadas pelo *Crawler Agent* em dois sites de venda de produtos: Americanas e Luiza. Cada amostra foi checada, a fim de se confirmar a predição efetuada pelo agente, e o resultado de tal verificação é exibido na Tabela 3. Pode-se observar que tal amostragem contribuiu com dois novos conjuntos de exemplos, descritos pela Tabela 4.

Tabela 3 – Amostras obtidas pelo *Crawler Agent*

Conjunto	Preditos positivos	Preditos negativos	Falsos positivos	Falsos negativos	Positivos verdadeiros	Negativos verdadeiros
Americanas	177	6	17	2	160	4
Luiza	30	9	1	8	29	1

Tabela 4 – Exemplos obtidos das amostras do *Crawler Agent*

Nome do conjunto	Positivos	Negativos
Crawler_Americanas	162	21
Crawler_Luiza	37	2
Total de exemplos:	199	23

6.1.5 Aplicação do método e geração dos resultados parciais

Nesta etapa foram realizadas as atividades relacionadas à criação dos modelos para treinamento (classificador) e testes, através da aplicação das tarefas de pré-processamento descritas na seção 6.1.3, e à execução do algoritmo SVM^{light}, com a subsequente coleta dos resultados gerados.

Uma estratégia foi adotada para a seleção das amostras que comporiam os conjuntos de treinamento e teste, visando-se avaliar o desempenho do algoritmo para os seguintes critérios:

- Capacidade de classificação dos exemplos em partes do conjunto de treinamento;
- Capacidade de reconhecer os padrões nos exemplos obtidos da amostragem do *Crawler Agent*;
- Capacidade de classificação dos exemplos preditos em outros conjuntos.

Capacidade de classificação dos exemplos preditos em partes do conjunto de treinamento

Para se atender este critério, foram avaliados os resultados obtidos da aplicação do algoritmo para conjuntos de testes compostos por subconjuntos (individualizados por sites) do conjunto de treinamento. Os testes foram realizados em três situações distintas, diferenciadas pela composição do conjunto de treinamento:

- Integraram o conjunto de treinamento as amostras obtidas dos sites Americanas, Bondfaro, Buscape, Extra, Kalunga, LivCultura, LivSaraiva, Luiza, MercadoLivre, Shoptime e Submarino. Os exemplos obtidos das amostras do *Crawler Agent* não foram incluídos nesta etapa. Os tempos gastos para a composição do dicionário e dos vetores, e para a criação do modelo de treinamento (execução do *svm_train*), bem como o tamanho do dicionário e os resultados para cada teste são apresentados na Tabela 5;
- Nos próximos testes realizados sob o mesmo critério, incluíram-se as amostras do site ShoppingUOL ao conjunto de treinamento utilizado nos testes anteriores. Da mesma forma, os resultados então obtidos foram inseridos na Tabela 6;
- Os últimos testes realizados sob o mesmo critério diferenciaram-se dos demais por agruparem no conjunto as amostras obtidas de todos os sites e os exemplos provenientes da amostragem do *Crawler Agent*. Portanto, foram incluídos no conjunto de treinamento do teste anterior os exemplos contidos nas amostras EstadãoEconomia, GloboShopping, Crawler_Americanas e Crawler_Luiza. Os resultados desses testes são exibidos na Tabela 7.

Tabela 5 – Resultados de testes realizados em partes do conjunto de treinamento (1)

Conjunto	Classificações		<i>Accuracy</i>	<i>Precision</i>	<i>Recall</i>
	Corretas	Incorretas			
Americanas	93	4	95,88%	100,00%	92,00%
Bondfaro	90	0	100,00%	100,00%	100,00%
Buscape	34	5	87,18%	100,00%	75,00%
Extra	58	0	100,00%	100,00%	100,00%
Kalunga	58	13	81,69%	80,60%	100,00%
LivCultura	66	0	100,00%	100,00%	100,00%
LivSaraiva	96	1	98,97%	100,00%	98,04%
Luiza	156	1	99,36%	100,00%	98,48%
MercadoLivre	115	6	95,04%	100,00%	88,00%
Shoptime	60	2	96,77%	100,00%	94,44%
Submarino	87	0	100,00%	100,00%	100,00%
Total de exemplos positivos:			478		
Total de exemplos negativos:			467		
Tamanho do dicionário:			9829		
Tempo gasto na geração do dicionário:			16s		
Tempo gasto na composição dos vetores:			2min5s		
Tempo gasto na criação do modelo:			5s		

Tabela 6 – Resultados de testes realizados em partes do conjunto de treinamento (2)

Conjunto	Classificações		<i>Accuracy</i>	<i>Precision</i>	<i>Recall</i>
	Corretas	Incorretas			
Americanas	94	3	96,91%	100,00%	94,00%
Bondfaro	90	0	100,00%	100,00%	100,00%
Buscape	34	5	87,18%	100,00%	75,00%
Extra	58	0	100,00%	100,00%	100,00%
Kalunga	58	13	81,69%	80,60%	100,00%
LivCultura	66	0	100,00%	100,00%	100,00%
LivSaraiva	96	1	98,97%	100,00%	98,04%
Luiza	156	1	99,36%	100,00%	98,48%
MercadoLivre	114	7	94,21%	97,78%	88,00%
Shoptime	60	2	96,77%	100,00%	94,44%
Submarino	87	0	100,00%	100,00%	100,00%
ShoppingUOL	34	3	91,89%	96,15%	92,59%
Total de exemplos positivos:			505		
Total de exemplos negativos:			477		
Tamanho do dicionário:			9947		
Tempo gasto na geração do dicionário:			16s		
Tempo gasto na composição dos vetores:			2min5s		
Tempo gasto na criação do modelo:			4s		

Tabela 7 – Resultados de testes realizados em partes do conjunto de treinamento (3)

Conjunto	Classificações		Accuracy	Precision	Recall
	Corretas	Incorretas			
Americanas	94	3	96,91%	100,00%	94,00%
Bondfaro	90	0	100,00%	100,00%	100,00%
Buscape	36	3	92,31%	100,00%	85,00%
Extra	58	0	100,00%	100,00%	100,00%
Kalunga	58	13	81,69%	80,60%	100,00%
LivCultura	66	0	100,00%	100,00%	100,00%
LivSaraiva	96	1	98,97%	100,00%	98,04%
Luiza	157	0	100,00%	100,00%	100,00%
MercadoLivre	115	6	95,04%	97,83%	90,00%
Shoptime	60	2	96,77%	100,00%	94,44%
Submarino	87	0	100,00%	100,00%	100,00%
ShoppingUOL	34	3	91,89%	96,15%	92,59%
EstadaoEconomia	24	1	96,00%	NA	NA
GloboShopping	23	0	100,00%	100,00%	100,00%
Crawler_Americanas	178	5	97,27%	98,16%	98,77%
Crawler_Luiza	38	1	97,44%	100,00%	97,30%
Total de exemplos positivos:			725		
Total de exemplos negativos:			527		
Tamanho do dicionário:			11205		
Tempo gasto na geração do dicionário:			20s		
Tempo gasto na composição dos vetores:			2min50s		
Tempo gasto na criação do modelo:			5s		

Capacidade de reconhecer os padrões nos exemplos obtidos da amostragem do *Crawler Agent*

Primeiramente, e para que possam ser realizadas as comparações de desempenho entre a classificação realizada pelo *Crawler Agent* com a realizada pelo algoritmo SVM^{light}, é apresentada na Tabela 8 a avaliação do resultado da classificação do primeiro, cujas amostras foram descritas anteriormente pela Tabela 3. Os conjuntos de amostras foram obtidos pela execução do *Crawler Agent*, limitando-se a profundidade da busca no site em 4 níveis, e classificados por ele pelo uso do algoritmo C4.5.

Tabela 8 – Avaliação dos resultados apresentados pelo *Crawler Agent*

Conjunto	Classificações		Accuracy	Precision	Recall
	Corretas	Incorretas			
Crawler_Americanas	164	19	89,62%	90,40%	98,76%
Crawler_Luiza	30	9	76,92%	96,67%	78,38%

Os testes realizados com o algoritmo SVM^{light} foram realizados utilizando-se o conjunto de treinamentos composto pelas amostras descritas pela Tabela 6, e a avaliação dos resultados obtidos é apresentada pela Tabela 9.

Tabela 9 – Resultados dos testes realizados sobre as amostras do *Crawler Agent*

Conjunto	Classificações		<i>Accuracy</i>	<i>Precision</i>	<i>Recall</i>
	Corretas	Incorretas			
Crawler_Americanas	165	18	90,16%	98,65%	90,12%
Crawler_Luiza	30	9	76,92%	100,00%	75,68%

Capacidade de classificação dos exemplos preditos em outros conjuntos

A avaliação de tal capacidade baseou-se nos resultados de testes realizados com um conjunto de treinamento composto pelas mesmas amostras descritas na Tabela 6. Foram realizados testes com os exemplos obtidos dos sites GloboShopping e EstadaoEconomia, cujas amostras não faziam parte do conjunto de treinamento. Os resultados obtidos são apresentados na Tabela 10.

Tabela 10 – Resultados de testes realizados em outros conjuntos de exemplos (1)

Conjunto	Classificações		<i>Accuracy</i>	<i>Precision</i>	<i>Recall</i>
	Corretas	Incorretas			
GloboShopping	22	1	95,65%	100,00%	95,24%
EstadaoEconomia	20	5	80,00%	NA	NA

Uma segunda avaliação sob tal critério envolveu a aplicação do modelo gerado a partir do conjunto de treinamento composto por todos os exemplos das amostras anteriores, descrito na Tabela 7, para a classificação de novos conjuntos, obtidos de sites comerciais e não comerciais. A descrição de tais conjuntos e os resultados da classificação são apresentados na Tabela 11.

Tabela 11 – Resultados de testes realizados em outros conjuntos de exemplos (2)

Conjunto	Exemplos			Classificação				Avaliação		
	Pos	Neg	Tot	FP	FN	TP	TN	<i>Accuracy</i>	<i>Precision</i>	<i>Recall</i>
Banespa S/A	0	60	60	1	0	0	59	98,33%	-	-
ICMC (USP)	0	90	90	0	0	0	90	100,00%	-	-
Receita Federal	0	40	40	0	0	0	40	100,00%	-	-
Ponto Frio	85	0	85	0	0	85	0	100,00%	100,00%	100,00%
Ri Happy	33	3	36	0	0	33	3	100,00%	100,00%	100,00%
Som Livre	54	19	73	1	7	47	18	89,04%	97,92%	87,04%

6.2 Tecnologia utilizada na implementação do experimento

Para a codificação das classes, que constituem a solução descrita na seção anterior, decidiu-se pela utilização da linguagem Java. Os principais motivos que apoiaram tal decisão foram:

- Compatibilidade da linguagem com o paradigma de programação orientada a objetos;
- Disponibilidade de ferramentas de edição, compilação e execução gratuitas;
- Existência de padrões adequados para documentação do código (JavaDoc);
- Portabilidade e reutilização do código;
- Disponibilidade de recursos para otimização de operações e para chamada de programas executáveis externos, fornecidos com a API Jsdk.

As classes foram implementadas e organizadas em um pacote, denominado “tcweb”, utilizando a API, o compilador e o ambiente de execução presentes no pacote j2sdk1.3.1 (*Java 2 SDK Standard Edition*). Diversos recursos do pacote Java foram utilizados para a otimização de:

- Acesso a disco: através do uso de estruturas de dados apropriadas ao armazenamento em memória, como *Set* e *Array*;
- Busca e recuperação: pelo uso de classes apropriadas, como *HashSet* e *HashTable*;
- Ordenação: notada pela aplicação da classe *SortedMap*.

Apesar da disponibilidade de ferramentas gratuitas para a edição e depuração do código Java, decidiu-se pela utilização de um editor de textos simples, mas poderoso, chamado “vim”.

O ambiente operacional utilizado nas atividades de codificação, compilação, execução e testes do experimento foi composto por um microcomputador (processador AMD-Athlon 1 GHz, com 512 MB de RAM) executando o sistema operacional Linux (distribuição Debian 3.0, versão 2.4.18 do kernel). A opção por tal ambiente firmou-se principalmente pelos seguintes aspectos:

- Sistema operacional e utilitários de código aberto gratuitos, minimizando o custo com a composição do ambiente;
- Domínio da tecnologia;
- Performance e confiabilidade notadas na operação do sistema.

6.3 Considerações finais

Neste capítulo foi descrita a solução adotada pelo presente trabalho para o problema da classificação de páginas Web de sites brasileiros, através da aplicação do algoritmo SVM^{light} no processo de decisão realizado pelo *Crawler Agent* do sistema DEEPSIA.

Encontram-se também descritas a modelagem e a aplicação do experimento, de acordo com a metodologia concebida para a solução. Os dados sobre o processo de amostragem e os resultados dos testes realizados sob três critérios distintos são apresentados e avaliados.

Ao final do capítulo encontram-se as informações sobre a tecnologia e os recursos aplicados na implementação e na operação do experimento.

7 Resultados Obtidos

Buscou-se alcançar, durante a realização do presente trabalho, um resultado satisfatório para o problema de classificação de páginas Web enfrentado pelo *Crawler Agent* do sistema DEEPSIA, sobre o domínio de sites brasileiros. O fato inovador foi a aplicação do algoritmo SVM, utilizando-se a implementação SVM^{light}, que não havia sido testado anteriormente pelos desenvolvedores do *Crawler Agent*.

Outros resultados também foram alcançados, como o modelo para a implementação de um experimento, incluindo as atividades de pré-processamento de conteúdos textuais de páginas Web, na aplicação do método SVM.

A comparação com os resultados apresentados pelo *Crawler Agent* pode ser obtida pelo confronto dos dados apresentados na Tabela 8 e na Tabela 9.

A Tabela 12 sintetiza a avaliação dos resultados dos testes efetuados sobre as amostras do conjunto problema sob os três critérios distintos, apontados no Capítulo 6.

Tabela 12 – Sumário da avaliação dos resultados obtidos na realização dos testes

Capacidade de classificar os exemplos	<i>Accuracy</i>		<i>Precision</i>		<i>Recall</i>	
	Pior	Melhor	Pior	Melhor	Pior	Melhor
Preditos em partes do conjunto de treinamento	81,69%	100,00%	80,60%	100,00%	75,00%	100,00%
Amostrados pelo <i>Crawler Agent</i>	76,92%	90,16%	98,65%	100,00%	75,68%	90,12%
Preditos em outros conjuntos	80,00%	100,00%	97,92%	100,00%	87,04%	100,00%

8 Conclusões e Proposta para Trabalhos Futuros

Conclui-se finalmente que a aplicação de um método estatístico de classificação de textos para o problema que envolve a decisão sobre uma página Web conter (ou não) a descrição de um produto à venda pode apresentar resultados satisfatórios, sendo estes dependentes da qualidade dos exemplos escolhidos para a composição do conjunto de treinamento.

Conclui-se também que o algoritmo SVM, através da implementação SVM^{light}, pode ser aplicado à solução de problemas com classificação de conteúdos textuais e que, no caso de páginas Web, um processo adequado de preparação de tal conteúdo deve ser elaborado, como o apresentado pelo presente trabalho.

A solução ora apresentada restringe-se ao problema de classificação de páginas Web e avalia o algoritmo SVM para tanto. Sugere-se como um futuro trabalho a aplicação do mesmo algoritmo, e a metodologia aqui demonstrada, para a solução de outros problemas relacionados à classificação de conteúdos textuais em categorias distintas.

A experiência agregada por este trabalho ao grupo de pesquisas pode ser útil quando da aplicação de solução parecida em processos de decisão enfrentados por sistemas detectores de intrusão. O reconhecimento de padrões de ataque através de fatos descritos em conteúdos textuais de arquivos de *log* do sistema operacional pode se tornar um objeto da aplicação de uma solução similar à explorada pelo presente trabalho.

Outra sugestão para a elaboração de futuros trabalhos consiste da aplicação de outros métodos de classificação de textos para a solução do mesmo problema que se buscou solucionar.

Os impactos da integração da solução ao *Crawler Agent* do sistema DEEPSIA, como a avaliação de sua aplicação em um domínio mais amplo composto por páginas Web européias, e a sua conseqüente adaptação para outras línguas presentes nas descrições textuais de tal páginas, consiste um objeto de estudo importante, ora também sugerido.

9 Referências Bibliográficas

- [ABA00] ABASOLO, J.M.; GÓMEZ, M. *MELISA. An ontology-based agent for information retrieval in medicine.* In: ECDL 2000 Workshop on the Semantic Web, Portugal. 2000.
- [AGG01] AGGARWAL, C.C.; AL-GARAWI, F.; YU, P.S. *Intelligent Crawling on the World Wide Web with Arbitrary Predicates.* In: Proceedings of the 10th World Wide Web Conference (WWW10), 2001.
- [AMB01] AMBRÓSIO, D.R. *Alternativas de Implementação de Reconhecimento de Padrões para Agentes Móveis em Ambiente de Segurança Computacional.* São Carlos, 2001. Dissertação (Mestrado) - Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo.
- [BAD02] BADUE, C.; BAEZA-YATES, R.; MEIRA, W.; RIBEIRO-NETO, B.; ZIVIANI, N. *Distributed Architecture for Information Retrieval.* In: Proceedings of the First International Seminar on Advanced Research in Electronic Business. Rio de Janeiro, Brazil, 2002.
- [BEL89] BERNERS-LEE, T. *Information Management: A Proposal.* In: W3 Archive web page: <http://www.w3.org/History/1989/proposal.html>, 1989.
- [BEL01] BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. *The Semantic Web.* Scientific American. 2001.
- [BER99] BERNARDES, M.C. *Avaliação do uso de agentes móveis em segurança computacional.* São Carlos, 1999. Dissertação (Mestrado) - Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo.
- [BON98] BONIFÁCIO Jr., J.M. *Sistemas de Segurança Distribuído: Integração de Firewalls com Sistemas de Detecção de Intrusão.* São Carlos, 1998. Dissertação (Mestrado) - Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo.
- [BUR98] BURGESS, C.J.C. *A Tutorial on Support Vector Machines for Pattern Recognition.* In: Data Mining and Knowledge Discovery 2, 121-167, 1998.
- [CAN97] CANSIAN, A.M. *Desenvolvimento de um sistema adaptativo de detecção de intrusos em redes de computadores.* São Carlos, 1997. 153p. Tese (Doutorado) - Instituto de Física de São Carlos, Universidade de São Paulo.
- [CHA01] CHAVES, M.S.; VIEIRA, R.; RIGO, S. *Uso de ontologias para gerenciamento e acesso a documentos na web.* Universidade do Vale do Rio dos Sinos. 2001.
- [CIC94] CICILINI, R. *Desenvolvimento de um Agente SNMP para Plataformas Rodando DOS.* São Carlos, 1994. 107p. Dissertação (Mestrado) - Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo.

- [COR98] CORLEY, S.; TESSELAAR, M.; COOLEY, J.; MEINKHN, J.; MALABOCCHIA, F.; GARIJO, F. *The Application of Intelligent and Mobile Agents to Network and Service Management*. In: 5th International Conference On Intelligence in Services and Networks. Antuérpia, Bélgica, 1998.
- [COU02] COUTINHO, B.; TEODORO, G.; TAVARES, T.; PINTO, R.; NOGUEIRA, D.; MEIRA JR, W. *Assessing the impact of distribution on e-business services*. In: Proceedings of the First International Seminar on Advanced Research in Electronic Business. Rio de Janeiro, Brazil, 2002.
- [CRA99] CRAVEN, M. et al *Learning to Construct Knowledge bases from de World Wide Web*. Elsevier Science. 1999.
- [DOO97] DOORENBOS, R.B.; ETZIONI, O.; WELD, D.S. *A Scalable Comparision-Shopping Agent for the World-Wide Web*. In: Proceedings of Agents'97. 1997.
- [END98] ENDLER, M. *Novos Paradigmas de Interação usando Agentes Móveis*. In: SBRC98. Departamento de Ciência da Computação – IME, Universidade de São Paulo.
- [ENG99] ENGUIX, C.F.; DAVIS, J.G. *Filling the Gap: New Models for Systematic Page-based Web Application Development & Maintenance*. In Proceedings of the 8th World Wide Web Conference (WWW8), 1999.
- [FIL01] FILHO, S.F.P. *Avaliação para Ambientes Servidores para Agentes Móveis*. São Carlos, 2001. Dissertação (Mestrado) - Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo.
- [FRA96] FRANKLIN, S.; GRAESSER, A. *Is It An Agent, or Just a Program? A Taxonomy for Autonomous Agents*. In: Proceedings of the Third International Workshop on Agent Theories, Architectures and Languages. Springer-Verlag, 1996.
- [GAR01] GARÇÃO, A.S.; SOUSA, P.A.; PIMENTÃO, J.P.; SANTOS, B.R.; BLASQUÉZ, V.; OBRATANSKI, L. *Annex to DEEPSIA's Deliverable 4 – System Architecture*. Technical Report of IST PROJECT-1999-20483. January 2001.
- [GEN94] GENESERETH, M.R.; KETCHPEL, S.P. *Software Agents*. Communications of the ACM, 37(7): 48-53, 1994.
- [GÓM99] GÓMEZ-PÉREZ, A. *Ontological Engineering*. Laboratorio de Inteligencia Artificial, Universidad Politécnica de Madrid. 1999.
- [GOU98] GOULARTE, R. *Utilização de metadados no gerenciamento de acesso a servidores de vídeo*. São Carlos, 1998. Dissertação (Mestrado) - Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo.
- [GRU93] GRUBER, T.R. *A Translation Approach to Portable Ontology Specifications*. Technical Report KSL 92-71, Knowledge System Laboratory, Stanford. 1993.
- [GRÜ96] GRÜNINGER, M. *Designing and Evaluating Generic Ontologies*. Department of Industrial Engineering, University of Toronto. 1996.

- [HER02] HERRERA, J.F.; MARTINS JUNIOR, J.; MOREIRA, E. *A Model for Data Manipulation and Ontology Navigation in DEEPSIA*. In: Proceedings of the First International Seminar on Advanced Research in Electronic Business. Rio de Janeiro, Brazil, 2002.
- [HUH97] HUHNS, M.N.; SINGH, M.P. *Ontologies for Agents*. IEEE Internet Computing. November-December 1997.
- [IMA01] IMAMURA, C.Y. *Pré-processamento para Extração de Conhecimento de Bases Textuais*. São Carlos, 2001. Dissertação (Mestrado) - Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo.
- [ISA95] ISAKOVITZ, T.; STOHR, E.A.; BALASUBRAMANIAN, P. *RMM: A Methodology for Structured Hypermedia Design*. Communications of the ACM, 38(8):34-44, August 1995.
- [JEN95] JENNINGS, N.R.; WOOLDRIDGE, M. *Intelligent Agents: Theory and Practice*. In: Knowledge Engineering Review. v. 10, n. 2, 1995.
- [JOA97] JOACHIMS, T. *A Probabilistic Analysis of the Roccio Algorithm with TFIDF for Text Categorization*. Universitat Dortmund. 1997.
- [JOA98] JOACHIMS, T. *Text Categorization with Support Vector Machines: Learning with Many Relevant Features*. Universitat Dortmund. 1998.
- [JOA99a] JOACHIMS, T. *Transductive Inference for Text Classification using Support Vector Machines*. In: Proceedings of the International Conference on Machine Learning (ICML). 1999.
- [JOA99b] JOACHIMS, T. *Making large-Scale SVM Learning Practical*. Advances in Kernel Methods – Support Vector Learning, B. Schölkopf and C. Burges and ^a Smola (ed.), MIT-Press, 1999.
- [KEP99] KEPHART, J.O.; GREENWALD, A. *Shopbot Economics*. In: Autonomous Agents '99, Seattle, 1999.
- [KLA01] KLAPSING, R.; NEUMANN, G.; CONEN, W. *Semantics in Web Engineering Applying the Resource Description Framework*. In: IEEE Multimedia. April-June 2001.
- [LAW98] LAWRENCE, S.; GILES, C.L. *Searching the World Wide Web*. NEC Research Institute, 1998.
- [LIE95] LIEIRA, J.F. *Utilização de Áudio e Vídeo em Sistemas Gerenciadores de Redes de Computadores*. São Carlos, 1995. 111p. Dissertação (Mestrado) - Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo.
- [LIN96] LINGNAU, A.; DROBNIK, O. *An Infrastructure for Mobile Agentes: Requeriments and Architecture*. Frankfurt am Main, Germany, 1996.
- [MAG02] MAGALHÃES, J.; LUCENA, C. *A Multi-Agent Product Line Architecture for Websearching*. In: Proceedings of the First International Seminar on Advanced Research in Electronic Business. Rio de Janeiro, Brazil, 2002.

- [MEI00] MEIRA Jr., W.; MURTA, C.; RESENDE R. *Comércio Eletrônico na WWW*. XII Escola de Computação, São Paulo, SP, Julho, 2000.
- [MIL02] MILAGRES, F.; MOREIRA, E.; PIMENTÃO, J.; SOUSA, P. *Security Analysis of a Multi-Agent System in EU's DEEPSIA Project*. In: Proceedings of the First International Seminar on Advanced Research in Electronic Business. Rio de Janeiro, Brazil, 2002.
- [MIN94] MINSK, M.; RIECKEN, D. *A conversation with Marvin Minsk about Agents*. Communications of the ACM, 37(7):23-29, 1994.
- [MLA98] MLADENIC, D. *Machine Learning on non-homogeneous, distributed text data*. PhD thesis, University of Ljubljana, Faculty of Computer and Information Science, Ljubljana. 1998.
- [MON97] MONARD, M.C. et al *Uma Introdução ao Aprendizado Simbólico de Máquina por Exemplos*. Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo. 1997.
- [MOR95] MORAES, S. *Voz em Sistemas Computacionais: Projeto e Implementação de Módulos de Processamento de Voz em Gerenciamento de Redes*. São Carlos, 1995. 103p. Dissertação (Mestrado) - Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo.
- [MOR97] MORISHITA, F.T. *Uma Avaliação Evolutiva dos Protocolos de Gerenciamento da Internet: SNMPv1, SNMPv2 e SNMPv3*. São Carlos, 1997. 68p. Dissertação (Mestrado) - Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo.
- [MOU97a] MOURO, R.B.; MORISHITA, F.T.; MOREIRA, E.S. *NetTracker :Uma Arquitetura Operacional Extensível Para Ferramentas de Gerenciamento de Redes*. In: 15º Simpósio Brasileiro de Redes de Computadores, São Carlos, 1997. p164-174.
- [MOU97b] MOURO, R.B. *Uma Arquitetura Operacional Extensível para Ferramentas de Gerenciamento de Redes*. São Carlos, 1997. 60p. Dissertação (Mestrado) - Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo.
- [MOW97] MOREIRA, D.A.; WALCZOWSKI, L.T. *Using Software Agents to Generate VLSI Layouts*. IEEE Expert, p26-32, November-December 1997.
- [NIC98] NICOLAS, P.R. *Agent-based Workflow Automation*. In: WWW: White Paper: Agent-based Workflow Automation, <http://www.ikonodyne.com/whitewkflow/agents.html>. Ikonodyne Inc, 1998.
- [NIG99] NIGAM, K. et al *Text Classification from Labeled and Unlabeled Documents Using EM*. Kluwer Academic Publishers. 1999.
- [NWA96] NWANA, H.S. *Software Agents: An Overview*. In: Knowledge Engineering Review, vol. 11, no. 3, p205-244, October-November 1996.

- [ODA94] ODA, C.S. *Desenvolvimento de um Sistema Monitor Gráfico Baseado em Protocolo de Gerenciamento SNMP*. São Carlos, 1994. 111p. Dissertação (Mestrado) - Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo.
- [PER01] PEREZ, H.A.M.; MOURA, A.M.C.; TANAKA, A.K. *Extração de Dados em Sistemas de Informações Ambientais: Arquitetura e Esquema de Metadados*. Instituto Militar de Engenharia. 2001.
- [RAP02] RAPPA, M. *Managing the Digital Enterprise: Business Models on the Web*. In: E-Commerce Learning Center of North Carolina State University web site: <http://ecommerce.ncsu.edu/research.html>. Último acesso em: 23/12/2002.
- [REA98] REAMI, E.R. *Especificação e Prototipagem de um Ambiente de Gerenciamento de Segurança Apoiado por Agentes Móveis*. São Carlos, 1998. 82p. Dissertação (Mestrado) - Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo.
- [SCH99] SCHWABE, D.; PONTES, R.A. *A Method-based Web Application Development Environment*. In Proceedings of the 8th World Wide Web Conference, 1999.
- [SEB99] SEBASTIANI, F. *Machine Learning in Automated Text Categorization*. Technical Report IEI B4-31-12-99, Instituto di Elaborazione della Informazione, Consiglio Nazionale delle Ricerche, Pisa, Italy. 1999.
- [SEL94] SELKER, T. *Coach: A Teaching Agent that Learns*. Communications of the ACM, 37(7):92-99. 1994.
- [SON02] SONG, M.; PEREIRA, A.; GORGULHO, G.; CAMPOS, S.; MEIRA JR, W. *Model Checking Patterns for e-Commerce Systems*. In: Proceedings of the First International Seminar on Advanced Research in Electronic Business. Rio de Janeiro, Brazil, 2002.
- [TAV99] TAVARES, D.M.; CASTEJON, E.F.; ROSSI, G.B.; CANSIAN, A.M.; MOREIRA, E.S. *ACME! (Advanced Counter-Measures Environment), Um Mecanismo de Captura e Análise de Pacotes para Aplicação em Detecção de Assinaturas de Ataque*. Anais do Primeiro Simpósio de Segurança, em Informática, p39-46, 1999.
- [TAV02] TAVARES, D.M. *Avaliação de Técnicas de Captura para Sistemas Detectores de Intrusão*. São Carlos, 2002. 110p. Dissertação (Mestrado) - Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo.
- [VAP95] VAPNIK, V. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.
- [WAN02] WANG, Y.; HU, J. *A Machine Learning Based Approach for Table Detection on The Web*. In: Proceedings of the WWW2002 Conference. Honolulu, Hawaii, USA, 2002.
- [WEI98] WEISS, S.M.; INDURKHYA, N. *Predictive Data Mining: A Pratical Guide*. Morgan Kaufmann Publisher, Inc. p75-79;p197-199. 1998.

[ZAK02] ZACON, R.H. *Hobbes' Internet Timeline* v5.6. In:
<http://www.zakon.org/robert/internet/timeline/>. Último acesso em: 23/12/2002.