Pontifícia Universidade Católica
do Rio de Janeiro

Fábio de Azevedo Soares

Categorização Automática de Textos Baseada em Mineração de Textos

Tese de Doutorado

Tese apresentada como requisito parcial para obtenção do grau de Doutor pelo Programa de Pós-graduação em Engenharia Elétrica do Departamento de Engenharia Elétrica da PUC-Rio.

Orientadora: Prof. Marley M. B. R. Vellasco

Co-Orientador: Prof. Emmanuel P. L. Passos

Rio de Janeiro

Junho de 2013

PONTIFÍCIA UNIVERSIDADE CATÓLICA
DO RIO DE JANEIRO

Fábio de Azevedo Soares

Categorização Automática de Textos Baseada em Mineração de Textos

Tese apresentada como requisito parcial para obtenção do grau Doutor pelo Programa de Pós-Graduação em Engenharia Elétrica do Departamento de Engenharia Elétrica do Centro Técnico Científico da PUC-Rio. Aprovada pela Comissão Examinadora abaixo assinada.

Profa. Marley Maria Bernardes Rebuzzi Vellasco
Orientadora
Departamento de Engenharia Elétrica – PUC-Rio

Prof. Emmanuel Piseces Lopes Passos
Co-Orientador
Aposentado do IME

Profa. Karla Tereza Figueiredo Leite
UEZO

Prof. Rubens Nascimento Melo
Departamento de Informática -PUC-Rio

Prof. Ronaldo Ribeiro Goldschmidt
UFRRJ

Prof. Douglas Mota Dias
Departamento de Engenharia Elétrica - PUC-Rio

Prof. Cláudio Márcio do Nascimento Abreu Pereira
Comissão Nacional de Energia Nuclear

Prof. José Eugenio Leal
Coordenador Setorial do Centro
Técnico Científico - PUC-Rio

Rio de Janeiro, 10 de Junho de 2013

Fábio de Azevedo Soares

Graduou-se Bacharel em Ciência da Computação em 2006. Mestre pela PUC-Rio em 2008. Atua como Desenvolvedor de Softwares, principalmente no desenvolvimento de Sistemas de Apoio à Decisão. Leciona para o ensino universitário. Tem interesse na pesquisa de novos algoritmos, principalmente, na área de Mineração de Textos e Aprendizado de Máquina.

Ficha Catalográfica

Soares, Fábio de Azevedo

Categorização automática de textos baseada em mineração de textos / Fábio de Azevedo Soares ; orientadores: Marley M. B. R. Vellasco, Emmanuel P. L. Passos. – 2013.

158 f. ; 30 cm

Tese (doutorado)–Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Elétrica, 2013.

Inclui bibliografia

1. Engenharia elétrica – Teses. 2. Mineração de textos. 3. Categorização. 4. Framework. 5. Português brasileiro. 6. Automática. I. Vellasco, Marley M. B. R. II. Passos, Emmanuel P. L. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Elétrica. IV. Título.

CDD: 621.3

Agradecimentos

A Deus por ser e pela oportunidade de todo dia começar de novo.

A Gabriela Soares por dar um novo sentido a minha vida, ser a minha fonte de amor e esperança, ensinar mais do que posso aprender e pelo sorriso de cada dia.

Ao professor Emmanuel Passos pelo apoio, dedicação, paciência, amizade e inspiração de vida.

À professora Marley Vellasco pela confiança depositada, oportunidade de realizar este trabalho e exemplo de mestre.

Aos meus pais pelo carinho, preocupação e expectativa de conclusão deste trabalho.

Ao meu amigo Thiago Mendonça por incentivar e compreender mais do que precisava.

Ao CNPq e à CAPES pelo apoio financeiro.

À PUC-Rio e à Vice Reitoria Acadêmica (VRAc) pela bolsa de isenção que me foi concedida.

Resumo

Soares, Fábio de Azevedo; Vellasco, Marley Maria Bernardes Rebuzzi (Orientadora); Passos, Emmanuel Piseces Lopes Passos (Co-Orientador). **Categorização Automática de Textos Baseada em Mineração de Textos.** Rio de Janeiro, 2013. 158p. Tese de Doutorado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

A Categorização de Documentos, uma das tarefas desempenhadas em Mineração de Textos, pode ser descrita como a obtenção de uma função que seja capaz de atribuir a um documento uma categoria a que ele pertença. O principal objetivo de se construir uma taxonomia de documentos é tornar mais fácil a obtenção de informação relevante. Porém, a implementação e a execução de um processo de Categorização de Documentos não é uma tarefa trivial: as ferramentas de Mineração de Textos estão em processo de amadurecimento e ainda, demandam elevado conhecimento técnico para a sua utilização. Além disso, exercendo grande importância em um processo de Mineração de Textos, a linguagem em que os documentos se encontram escritas deve ser tratada com as particularidades do idioma. Contudo há grande carência de ferramentas que forneçam tratamento adequado ao Português do Brasil. Dessa forma, os objetivos principais deste trabalho são pesquisar, propor, implementar e avaliar um *framework* de Mineração de Textos para a Categorização Automática de Documentos, capaz de auxiliar a execução do processo de descoberta de conhecimento e que ofereça processamento linguístico para o Português do Brasil.

Palavras-chave

Mineração de Textos; Categorização; Framework; Português brasileiro; Automática.

Abstract

Soares, Fábio de Azevedo; Vellasco, Marley Maria Bernardes Rebuzzi (Advisor); Passos, Emmanuel Piseces Lopes Passos (Co-Advisor). **Automatic Text Categorization Based on Text Mining.** Rio de Janeiro, 2013. 158p. Ph.D. Thesis - Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

Text Categorization, one of the tasks performed in Text Mining, can be described as the achievement of a function that is able to assign a document to the category, previously defined, to which it belongs. The main goal of building a taxonomy of documents is to make easier obtaining relevant information. However, the implementation and execution of Text Categorization is not a trivial task: Text Mining tools are under development and still require high technical expertise to be handled, also having great significance in a Text Mining process, the language of the documents should be treated with the peculiarities of each idiom. Yet there is great need for tools that provide proper handling to Portuguese of Brazil. Thus, the main aims of this work are to research, propose, implement and evaluate a Text Mining Framework for Automatic Text Categorization, capable of assisting the execution of knowledge discovery process and provides language processing for Brazilian Portuguese.

Keywords

Text Mining; Text Categorization; Framework; Brazilian Portuguese; Automatic.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Equações

Lista de Siglas

CT	Categorização de Textos
ETL	Extract Transform Load
IA	Inteligência Artificial
KDD	Knowledge Discovery in Databases
KDT	Knowledge Discovery in Texts
k-NN	k-Nearest Neighbors
NILC	Núcleo Interinstitucional de Linguística Computacional
PLN	Processamento de Linguagem Natural
RN	Redes Neurais
SGBD	Sistema Gerenciador de Bancos de Dados
SVM	Support Vector Machine
VISL	Visual Interactive Syntax Learning

1
Introdução

1.1.
Motivação

Informações podem ser armazenadas das mais variadas maneiras. O modo mais clássico de armazenamento de informação é através da palavra escrita, impressa. O acesso à informação estocada desta forma é lento, difícil, e de baixo rendimento. Nestes casos, para todas as etapas da manipulação da informação é necessária a presença do ser humano, que com suas limitações na capacidade de aquisição de conhecimento e processamento de grande volume de informação, constitui o principal gargalo do processo (MANDEL, SIMON & DELYRA, 1997).

A modernização dos últimos anos tornou as tecnologias de informação uma realidade inerente às vidas de todos nós. Das grandes multinacionais às pequenas empresas, das instituições públicas ao ensino e na nossa própria casa, termos como informática, computador, Internet e multimídia, entre tantos outros, passaram a fazer parte das tarefas do dia-a-dia, transformando-se em instrumentos fundamentais do trabalho. Todo este avanço tecnológico proporcionou meios muito mais eficientes para o armazenamento e disseminação de informação, esta, no formato digital, uma condição necessária para o amplo uso dos computadores no seu processamento.

Contudo, o maciço crescimento da oferta de dados traz consigo grandes desafios em termos de obtenção de informação. Embora usados muitas vezes como sinônimos, os termos dados e informação possuem significados distintos:

- Dado: um fenômeno qualquer, desprovido de significado e contexto (LAUDON & LAUDON, 2002).
- Informação: resultado do processamento, manipulação e organização de dados que passam a ter significados e, portanto, podem ser contextualizados, interpretados e compreendidos (GOLDSCHMIDT & PASSOS, 2005).

Os dados estão por toda parte. A maioria das organizações não sofre falta de dados, mas, sim, de uma abundância de dados redundantes e inconsistentes (SINGH, 2001). A informação desejada encontra-se entre os bits e bytes armazenados, por exemplo, em um disco rígido. Esses dados, após uma série de processamentos que envolvem, por exemplo, operações lógicas, serão transformados em informação. Finalmente, quando essa informação é recuperada, interpretada e analisada, chega-se ao conhecimento. É o que está resumido no diagrama da Figura 1.

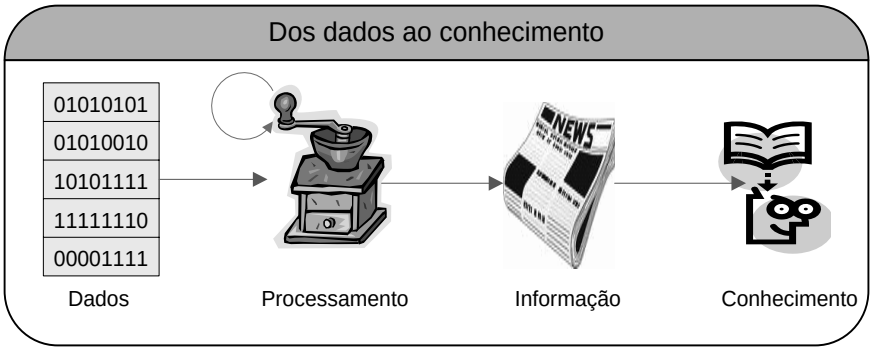


Figura 1 - Processo de obtenção de conhecimento

Métodos tradicionais de análise de dados, baseados principalmente no manuseio direto dos dados pelo homem, simplesmente não permitem a manipulação de conjuntos volumosos de dados (SINGH, 2001).

Técnicas de Mineração de Dados (MD) há algum tempo são utilizadas para lidar com o processo de obtenção de conhecimento em grandes bases de dados, essas em formato rigidamente estruturado. Porém, recentemente, comprovou-se que oitenta e cinco por cento de toda a informação do mundo está armazenada sob a forma de documentos textuais (GDS PUBLISHING, 2008) (IBM, 2008), ou seja, texto em formato livre, desprovidos de estruturas de dados, inviabilizando a utilização de técnicas de Mineração de Dados.

Técnicas de Recuperação de Informação sempre foram utilizadas para o armazenamento de documentos textuais e a recuperação de informação associada a eles (BAEZA-YATES & BERTIER, 1999) (MANNING, RAGHAVAN, & SCHÜTZE, 2007). Antes desta explosão de informação, as tarefas de recuperação de informação eram restritas a bibliotecas, nas quais com a ajuda de um bibliotecário, qualquer assunto poderia ser encontrado. Entretanto, a utilização

dessas técnicas na enorme massa de dados disponíveis, atualmente, não garante que a relevância das informações retornadas atenda às necessidades. Além disso, como todo processo de Recuperação de Informação precisa ser formalizado pelo usuário, por exemplo, através de uma consulta, estas técnicas oferecem suporte somente à obtenção da informação requisitada pelo mesmo, ignorando a possibilidade da existência de conhecimento, até então, desconhecido, nestes dados.

Em virtude do crescimento contínuo do volume de dados eletrônicos disponíveis, em formato textual, técnicas de extração de conhecimento automáticas tornam-se cada vez mais necessárias para manipular essa gigantesca massa de dados. **Mineração de Textos**[1] ou **Descoberta de Conhecimento em Textos**[2] surge, neste contexto, como uma abordagem à obtenção de informação útil a partir de bases de dados em formato textual. O principal objetivo das técnicas de Mineração de Textos é a manipulação de documentos em formato textual com o objetivo da obtenção do conhecimento implícito presente nestes (ARANHA & PASSOS, 2006).

Dentre as muitas tarefas desempenhadas em Mineração de Textos, a Categorização de Textos (CT) é a que tem recebido maior interesse da comunidade científica (LINDEN, 2008). Ao ser categorizado, um documento passa a pertencer a um grupo previamente definido que contém outros documentos que são semelhantes entre si. Uma vez identificados os documentos afins, torna-se mais fácil distinguir a informação relevante solicitada por um usuário, em meio a um conjunto de documentos irrelevantes, e repassá-la às pessoas que podem utilizá-la e a transformar em ação.

Separar a informação em categorias de conhecimentos que facilitam a sua manipulação e recuperação é o objetivo principal de Categorização de Textos. Além disso, classificar não é somente uma tendência natural do conhecimento; é igualmente uma necessidade da inteligência humana (PEIXOTO, BATISTA & CAPELO, 2003).

Contudo, por ser uma área ainda em fase de amadurecimento, as ferramentas de Mineração de Textos estão em processo de desenvolvimento.

[1] Do termo em inglês, *Text Mining*.

[2] Do termo em inglês, *Knowledge Discovery in Texts – KDT*.

Segundo (GOLDSCHMIDT & PASSOS, 2005), as ferramentas de Mineração de Dados podem ser classificadas quanto à usabilidade em:

- Primeira geração: ferramentas de análise dedicadas à realização de uma única tarefa de Mineração de Dados, como a construção de classificadores ou a descoberta de *clusters*, sem suporte às demais etapas do processo.
- Segunda geração: suítes de aplicativos capazes de realizar diversas operações de pré-processamento, como análises e transformação dos dados, além de tarefas de descoberta, como classificação, clusterização e visualização.
- Terceira geração: soluções de Mineração de Dados embarcadas. Constituem aplicativos para a solução de um problema específico de uma organização, em que as técnicas de Mineração de Dados são empregadas sem a necessidade de intervenção ou até mesmo conhecimento do usuário final. Alguns exemplos dessa geração são os *softwares* para análise de concessão de empréstimos.
- Quarta geração: aplicações que auxiliam o homem na condução do processo de descoberta do conhecimento.

Adotando esse mesmo critério para as ferramentas de Mineração de Textos, a maior parte das ferramentas disponíveis está classificada entre a segunda e a terceira geração.

Outra peculiaridade relacionada ao processo de Mineração de Textos é que diferente de tabelas e relações estatísticas, documentos textuais possuem uma linguagem associada a eles. Apesar do processamento total da linguagem natural ainda estar fora de alcance com a tecnologia atual (NEVES, 2012), existem técnicas que são capazes de extrair valiosas informações da linguagem natural presente nos documentos.

Embora possa ser abordado de forma puramente estatística, quando abordado de forma semântica, isto é, empregando a linguagem natural presente nos documentos textuais para obter melhor representação da informação, o processo de Mineração de Textos tende a melhores resultados (ARANHA C. N., 2007). Segundo (MONTEIRO, GOMES & OLIVEIRA, 2006), a Mineração de

Textos ainda é uma área pouco explorada e precisa do desenvolvimento de projetos que tratem as particularidades relacionadas ao Português do Brasil.

1.2. Objetivos do Trabalho

Os objetivos principais deste trabalho são pesquisar, propor, implementar e avaliar uma metodologia para realizar a Categorização Automática de Textos em Português do Brasil. O processo de Mineração de Textos proposto por (ARANHA C. N., 2007) com algumas adaptações é empregado para apoiar a Categorização de Textos, permitindo a utilização de processamento linguístico para o Português do Brasil e maior automação das tarefas.

Para isso, os seguintes objetivos intermediários são desejados:

- Prover um *framework* para a execução do processo de Categorização de Textos em Português do Brasil.
- Empregar técnicas de Mineração de Textos para o fornecimento de ferramentas que auxiliem a execução da tarefa de Categorização de Textos.
- Fornecer tratamento linguístico específico à Língua Portuguesa do Brasil, isto é, que faça proveito das ricas informações semânticas presentes em qualquer linguagem natural.
- Automatizar as etapas necessárias para a realização de Categorização de Textos, otimizando as técnicas de pré-processamento textual e a escolha dos algoritmos de Aprendizado de Máquina e seus respectivos dos parâmetros.

Como forma de avaliar os conceitos apresentados, foram realizados diversos experimentos de Categorização de Textos em Português do Brasil. Espera-se que esses estudos possam ser motivadores e um bom ponto de partida para outros pesquisadores.

1.3. Trabalhos relacionados

Esta seção tem por objetivo resumir as características dos principais trabalhos relacionados à área de Categorização de Textos com o emprego de Mineração de Textos e Processamento de Linguagem Natural.

O sistema Aíuri, desenvolvido por (SILVA A., 2007), é, segundo seu autor, um ambiente acadêmico cooperativo de alto desempenho, integrado a ambientes de *grids* computacionais, para a execução de algoritmos de Mineração de Textos. Faz uso dos *grids* do Intragrid NACAD, administrado pelo Núcleo de Computação de Alto Desempenho (NACAD) da COPPE, que agrupa máquinas heterogêneas do laboratório em um *grid* com finalidade didática, e do E-Infraestructure Shared Between Europe and Latin America (EELA), que é uma infraestrutura para o desenvolvimento e implantação de grids para uso científico, conectando a Europa e a América Latina.

Desenvolvido na linguagem de programação Java, o sistema Aíuri é provido de interface *web* e de toda uma infraestrutura para capacitá-lo a executar algoritmos em modo local ou em ambientes de *grids* computacionais. É composto por três módulos:

- O primeiro módulo realiza as atividades básicas para o funcionamento do sistema, como autenticação e carregamento dos arquivos de usuário.
- O segundo módulo é o responsável pelas tarefas de pré-processamento de dados textuais.
- O terceiro módulo contém as implementações dos algoritmos que são utilizados na etapa de Mineração do processo de descoberta de conhecimento: Classificadores Bayesiano e de Ranqueamento Linear. Além disso, contempla a fase de Pós-Processamento, com a disponibilização de diversas métricas de avaliação de resultados.

A etapa de pré-processamento empregada neste estudo é capaz de realizar abordagens (ver item 2.5) baseadas nas análises Estatística e Semântica. A primeira fase desta etapa é Tokenização (ver item 3.2.1). Em seguida, é realizada a eliminação de termos considerados irrelevantes ou *stopwords*. Por fim, utiliza-se

processamento linguístico para realizar a normalização morfológica dos termos (ver item 3.2.3.4) ou *stemming*.

Utiliza o Modelo de Espaço Vetorial para criar uma representação estruturada dos documentos (ver item 4.3.1). Os métodos utilizados para o cálculo de relevância (ver item 4.3.1.3) de cada termo são baseados na frequência desses no documento ou em toda a coleção: IDF e TF-IDF.

A base de documentos utilizada no estudo é um extrato do *corpus* CETENFolha (ver item 6.6). Experimentos nesta base foram realizados, ora com documentos distribuídos de forma balanceada entre as categorias, ora sem qualquer controle sobre a distribuição dos documentos e suas respectivas categorias.

A métrica utilizada para avaliar o desempenho dos classificadores foi a Medida F (ver item 3.5.3). A estratégia de treinamento empregada foi a de *hold-out*. Os estudos concluem que entre os dois classificadores utilizados, o classificador bayesiano, em média, foi mais rápido, e obteve modelos de classificação superiores aos obtidos pelo classificador de ranqueamento linear, sendo, desta maneira, uma boa opção de classificador para os textos em questão. Ambos os algoritmos tiveram desempenho superior nos conjuntos balanceados. Além disso, observou-se que para a maioria das classes quando houve a eliminação de *stopwords* e a utilização de *stemming*, ocorreram melhorias nos resultados obtidos, porém a utilização de *stemming* gerou grande aumento no tempo de processamento.

Em (MELO, 2007) é proposta uma abordagem ao reconhecimento de padrões textuais aplicada ao processo de classificação de documentos.

O modelo de representação de documentos utilizado é o Modelo de Espaço Vetorial (ver item 4.3.1.2). A seleção de atributos que compõem a representação vetorial dos documentos é baseada na escolha dos termos que possuem maior pontuação segundo as duas métricas de relevância utilizadas: escore de relevância e coeficiente de correlação.

Duas bases de documentos são utilizadas nos experimentos:

- A primeira é constituída por títulos e resumos de dissertações de mestrado e teses de doutorado do Departamento de Engenharia

Elétrica da COPPE/UFRJ. Possui cerca de quinhentos documentos organizados em cinco categorias.

- A segunda base é oriunda do corpus CETENFolha (ver item 6.6) e contém uma seleção de novecentos documentos também organizados em cinco categorias.

O processamento linguístico é empregado para identificar as classes gramaticais (ver item 3.2.3.2) dos termos que serão selecionados para o processo de classificação. Diversas combinações de classes gramaticais de termos são utilizadas, como por exemplo, somente o uso de termos substantivos, a combinação de substantivos e adjetivos, o conjunto de substantivos, nomes próprios e verbos, dentre outros.

Duas técnicas de Aprendizado de Máquina são utilizadas: Redes Neurais e Classificadores Bayesianos. Os melhores resultados foram obtidos com Classificadores Bayesianos, nas duas bases de documentos, utilizando cento e cinquenta termos selecionados entre as classes gramaticais substantivos, nomes próprios e adjetivos.

Em (CAMARGO, 2007) é apresentada uma abordagem linguística na classificação de textos em Português.

O modelo de representação de documentos utilizado também é o Modelo de Espaço Vetorial (ver item 4.3.1.2). Para a seleção de atributos representativos dos documentos são utilizadas duas métricas de relevância: escore de relevância e ganho de informação.

Semelhante ao trabalho de (MELO, 2007), duas bases de textos são utilizadas, sendo a primeira formada por um subconjunto CETENFolha de 855 textos classificados em cinco categorias: esporte, imóveis, informática, política e turismo, sendo 171 arquivos por classe. A outra base de textos é formada por um conjunto de textos gerados pela junção do título e resumo das teses de pós-graduação (mestrado e doutorado) da área de Engenharia Elétrica da Universidade Federal do Rio de Janeiro - COPPE-UFRJ. São 475 textos classificados nas categorias controle, microeletrônica, processamento de sinais, redes de computadores e sistemas de potência, sendo 95 arquivos por classe.

O processamento linguístico é utilizado para selecionar os termos pelas suas funções sintáticas e desta forma, tornar possível selecionar dentre as classes

gramaticais os termos que se pretende utilizar para representação dos textos. Para isso, o analisador sintático utilizado neste trabalho para extração de informações linguísticas dos documentos é o PALAVRAS, desenvolvido por (BICK, 2000) para a língua portuguesa. Ele realiza tarefas de processamento léxico-morfológico, análise sintática e faz parte de um grupo de analisadores sintáticos do projeto Visual Interactive Syntax Learning[3] (VISL), do Institute of Language and Communication da University of Southern Denmark.

Os classificadores utilizados são o Bayesiano, a Máquina de Vetor Suporte e um classificador baseado em regras de decisão. Dos resultados obtidos pode-se concluir que o classificador bayesiano é o que apresenta melhores resultados tanto para textos de jornal, com uma taxa de erro de 7,49%, quanto para textos científicos, com uma taxa de erro de 15,98%. Uma ressalva deve ser feita em relação ao classificador *Support Vector Machines* (SVM): embora apresente bons resultados em trabalhos anteriores (ver capítulo 5), não repetiu o desempenho para os experimentos realizados neste estudo.

O autor (LINDEN, 2008) utiliza a combinação de classificadores *K-Nearest Neighbors* (*k-NN*) e *Support Vector Machines* na categorização hierárquica de textos. O motivo pela categorização hierárquica de documentos, segundo o autor, é de que a distribuição de uma coleção de documentos em categorias auxilia na organização, busca e recuperação de informações, mas à medida que o número de documentos e o número de categorias aumentam, essa tarefa torna-se mais complexa para o ser humano; uma forma de ajudar a organizar informações no auxílio à compreensão humana é utilizar a organização hierárquica.

O modelo de representação de documentos utilizado é o Modelo de Espaço Vetorial (ver item 4.3.1.2). A seleção de atributos é realizada com a remoção de *stopwords* (ver item 3.2.2) e de termos que possuem frequência menor do que três em cada documento. Também é aplicada e a execução de um processo de lematização (ver item 3.2.3.4).

Nesse trabalho, a coleção de textos utilizada para o desenvolvimento dos experimentos é a **Folha-RIcol**[4], derivada do corpus em língua portuguesa

[3] Projeto VISL disponível em http://beta.visl.sdu.dk/visl/pt/

[4] Disponível em http://www.linguateca.pt/Repositorio/Folha-RIcol/

CETENFolha (Corpus do NILC/Folha de São Paulo). Essa coleção compreende um conjunto de documentos organizados em 28 categorias hierárquicas.

A avaliação do processo de categorização é realizada pelo método *hold-out* e com a utilização do software WEKA. Os experimentos utilizam dois métodos combinatórios de classificadores: a votação e a heurística *k*-NN+SVM. A heurística citada propõe que nós da árvore de categorias com mais de dois filhos sejam classificados pelo *k*-NN, e nós com um filho utilizam o classificador SVM. Os resultados obtidos por meio da votação de classificadores foram superiores (valor da medida F1 = 94,4%) aos resultados obtidos pela heurística proposta (valor da medida F1 = 84,9%).

Em (REIS, 2011) utiliza-se Mineração de Textos para a categorização de cadeias de caracteres formadas por sequências de aminoácidos. Segunda a autora, uma palavra pode ser entendida como uma cadeia específica de caracteres que possuem um valor semântico. Abordar as proteínas como sequências de aminoácidos que possuem valor biológico permitiu o emprego das técnicas de Mineração de Textos.

Baseando-se nisso, foi desenvolvida uma metodologia capaz de treinar um classificador de textos sobre sequências de proteínas. A tarefa do classificador é identificar a categoria a que determinada proteína pertence baseada na sequencia de aminoácidos que possui em sua estrutura.

A ferramenta de mineração de textos utilizada nos estudos de casos é o sistema Aíuri (SILVA A., 2007), comentado anteriormente. Desta forma, o texto (ou a sequência de aminoácidos) é representado matematicamente por meio do Modelo de Espaço Vetorial. A métrica de atribuição de pesos (ver item 4.3.1.3) utilizada para definir a frequência dos termos é TF-IDF.

A eliminação de *stopwords* (ver item 3.2.2) ou utilização de Lematização (ver item 3.2.3.4) não faz sentido no contexto dessa aplicação. A seleção das características mais representativas dos documentos (ou proteínas) é baseada simplesmente no valor de frequência de cada termo (aminoácido).

Os resultados obtidos com os dois algoritmos de classificação disponíveis na ferramenta, Bayesiano e Ranqueamento Linear, são semelhantes, bons, de fácil interpretação e possuem tempos de execução mínimos quando comparados às

ferramentas desenvolvidas para esse propósito e que não utilizam técnicas de Mineração de Textos.

(NEVES, 2012) propõe uma metodologia baseada em Mineração de Textos para a extração de informação de textos não estruturados em Português a fim de popular um *Data Mart* para a gestão das informações adquiridas.

A etapa de pré-processamento inicia com o processo de Tokenização (ver item 3.2.1) e conta com o auxílio de um léxico que é construído manualmente durante a execução desta fase e que valida os termos obtidos no processo de tokenização, além de identificar as Entidades Nomeadas (ver item 3.2.3.5) definidas pelo usuário. Utiliza o Modelo de Espaço Vetorial como estrutura de representação dos documentos.

Em seguida, inicia-se o processamento linguístico:

- O primeiro passo visa condensar os termos validados pelo usuário na etapa anterior em uma versão resumida por meio de um processo de Sumarização dos documentos. Após a realização da Sumarização, é iniciada a análise morfológica (ver item 3.2.3.2), realizada por meio de consultas ao léxico que contém para cada termo a sua classificação morfológica.
- Logo em seguida, começa o processo de Análise Sintática (ver item 3.2.3.6) de cada termo do documento sumarizado. Os dados resultantes do analisador sintático são agregados aos dados morfológicos dos termos, identificando para cada termo, além de sua classe gramatical, a sua função sintática no período analisado. Durante o processo de Análise Sintática também ocorre a identificação dos períodos do documento sumarizado.

Após o processamento linguístico, são realizadas tarefas de remoção de s*topwords* (ver item 3.2.2) e *stemming* (ver item 3.2.3.4). O passo seguinte é a classificação dos termos que consiste em identificar, nos períodos, os termos que serão candidatos a popular as dimensões do *Data Mart*. O processo de classificação é baseado em heurísticas que utilizam as informações obtidas na durante o processamento linguístico.

Por fim, é executado um processo de **ETL**[5] que irá popular o *Data Mart* gestão das informações adquiridas.

A Tabela 1 apresentada abaixo resume para cada um dos trabalhos relacionados os corpus, o modelo de representação dos documentos e a métrica de atribuição de pesos utilizados.

Tabela 1 - Resumo comparativo dos trabalhos relacionados quanto ao corpus, modelo de representação dos documentos e atribuição de pesos utilizados

Autor	**Corpus**	**Representação dos documentos**	**Atribuição de pesos**
(SILVA A., 2007)	CETENFolha	Modelo de Espaço Vetorial	IDF
			TF-IDF
(MELO, 2007)	CETENFolha	Modelo de Espaço Vetorial	Escore de relevância
	Teses		Coeficiente de correlação
(CAMARGO, 2007)	CETENFolha	Modelo de Espaço Vetorial	Escore de relevância
	Teses		Ganho de Informação
(LINDEN, 2008)	Folha-RICol	Modelo de Espaço Vetorial	Ganho de Informação
			Coeficiente de correlação
(REIS, 2011)	Aminoácidos	Modelo de Espaço Vetorial	TF-IDF
(NEVES, 2012)	Websites	Modelo de Espaço Vetorial	TF-IDF

A Tabela 2 resume para cada um dos trabalhos relacionados as técnicas de Processamento de Linguagem Natural empregadas, as tarefas de MT realizadas e os modelos de classificadores utilizados.

[5] Do termo em inglês, Extract Transform Load (Extração Transformação Carga).

Tabela 2 - Resumo comparativo dos trabalhos relacionados quanto a técnica de PLN, tarefa de MT e modelos de classificadores utilizados

Autor	**PLN**	**Tarefa**	**Classificadores**
(SILVA A. A., 2007)	Stemming	Classificação	Ranqueamento Linear
			Classificador Bayesiano
(MELO, 2007)	Stemming	Classificação	Redes Neurais
	POS Tagging		Classificador Bayesiano
(CAMARGO, 2007)	Stemming	Classificação	Classificador Bayesiano
	POS Tagging		SVM
			Regras
(LINDEN, 2008)	Stemming	Classificação	KNN
	POS Tagging		SVM
			Comitê
(REIS, 2011)	N/D	Classificação	Ranqueamento Linear
			Classificador Bayesiano
(NEVES, 2012)	Stemming	Sumarização	N/D
	POS Tagging	Extração de Informação	

Os trabalhos aqui apresentados serviram de inspiração para a elaboração da abordagem utilizada nesse trabalho que visa lidar com o problema de automatizar a tarefa Categorização de Textos apoiada por técnicas de Mineração de Textos.

1.4. Organização da Tese

Este tese possui mais sete capítulos conforme organização descrita abaixo.

O capítulo 2 apresenta os principais fundamentos da área de Mineração de Textos. Conceitos como os principais elementos de um processo de Mineração de Textos, a estrutura de um documento em linguagem natural, as características representativas de um documento, as abordagens ao processo de Mineração de Textos e as áreas correlatas são explicados.

No capítulo 3 são descritas as etapas da metodologia de Mineração de Textos estudada, composta por: coleta, pré-processamento, indexação, mineração e análise de resultados. As principais métricas utilizadas na análise de resultados também são comentadas.

O Capítulo 4 apresenta os conceitos da área de Recuperação de Informação que são utilizados para o entendimento da representação de documentos textuais,

assim como os modelos de representação de documentos e as métricas de atribuição de pesos.

Um estudo sobre a área de Categorização de Textos é apresentado no capítulo 5. Também é apresentada uma pesquisa de softwares dedicados a essa tarefa.

No capítulo 6 é relatado o *framework* proposto nesta tese e a descrição dos elementos que o compõem.

No capítulo 7 é apresentado o estudo de caso realizado e os resultados obtidos.

O capítulo 7 descreve as conclusões e as principais contribuições proporcionadas pelo presente trabalho. Alternativas de trabalhos futuros também são sugeridas.

2
Mineração de Textos: Fundamentos

O principal objetivo deste capítulo é fornecer uma visão do surgimento, estruturação e evolução dos procedimentos utilizados no processo de Descoberta de Conhecimento em Textos.

2.1.
Definição

Mineração de Textos (MT), Mineração de Dados Textuais ou Descoberta de Conhecimento em Textos surge, neste contexto, como uma abordagem ao processamento de grandes bases de dados textuais com o objetivo de extrair informação relevante e obter conhecimento implícito e útil a partir destas. Conhecimento útil é aquele que pode ser aplicado de forma a apoiar um processo de tomada de decisão, ou seja, é aquele que pode ser aplicado de forma a proporcionar benefícios.

De acordo com (ARANHA C. N., 2007), muitas são as definições encontradas na literatura:

- Pode-se então definir Descoberta de Conhecimento em Textos ou Mineração de Textos como sendo o processo de extrair padrões interessantes e não triviais, a partir de documentos textuais (TAN A.-H. , 1999).
- Mineração de Textos é a descoberta, através de meios computacionais, de informações desconhecidas ou novas, através da utilização de ferramentas de extração automática de informação, a partir de documentos de textos não estruturados (HEARST, 1999).
- Mineração de Textos é o estudo sobre a extração de informação de textos usando os princípios da linguística computacional (SULLIVAN, 2000).

Apesar de muitas definições, é fácil concluir que Mineração de Textos, de maneira análoga a Mineração de Dados, busca extrair informação útil de bases de dados através da identificação e exploração de padrões interessantes. Porém, é importante ressaltar a principal diferença entre a Mineração de Dados e a de Textos. Mineração de Textos é um processo de obtenção de conhecimento oriundo a partir de bases de dados textuais, ou seja, documentos em linguagem natural, e que, portanto, possuem pouca ou nenhuma estrutura de dados. Em Mineração de Dados, a obtenção de conhecimento ocorre em bases de dados fortemente estruturadas, geralmente armazenadas em Sistemas Gerenciadores de Bancos de Dados (SGBD).

Dados mantidos em Sistemas Gerenciadores de Bancos de Dados apresentam uma estrutura de representação ou esquema previamente definido. Um SGBD é uma coleção de programas que permitem ao usuário definir, construir e manipular bases de dados para as mais diversas finalidades (DATE, 2005). O principal objetivo de um SGBD é retirar da aplicação cliente a responsabilidade de gerenciar o acesso, manipulação e organização dos dados. Para isto, todo SGBD disponibiliza uma interface para que os seus clientes possam incluir, alterar ou consultar dados. O acesso e a manipulação destes esquemas são tarefas específicas do SGBD. Usuários ou aplicações realizam operações sobre estes dados com base neste esquema.

Muitas são as funções de um Sistema Gerenciador de Bancos de Dados, porém, uma delas provê integridade às bases de dados gerenciadas por este: integridade semântica. Integridade semântica é função de um SGBD responsável por garantir a armazenagem correta de dados em relação ao domínio (DATE, 2005). Por exemplo, na coluna de uma tabela destinada a armazenar informações sobre o saldo bancário de um cliente, somente valores numéricos podem ser inseridos. A Figura 2 demonstra o conceito de integridade semântica em um SGBD.

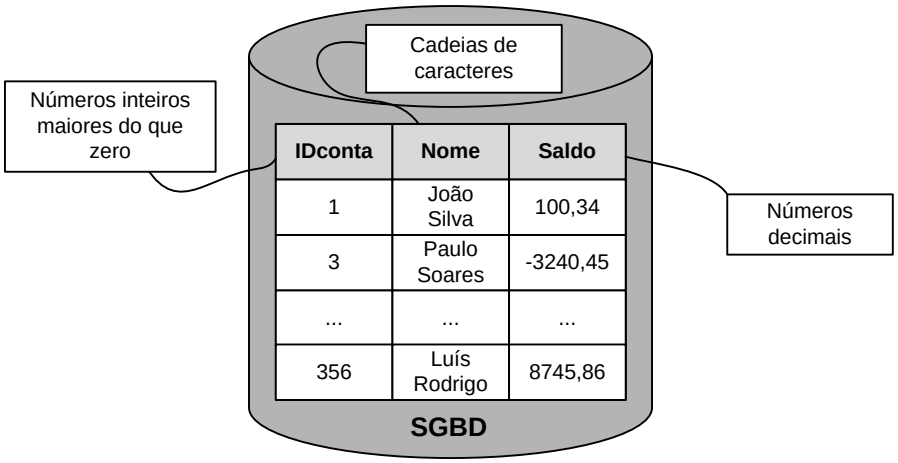


Figura 2 – Integridade semântica de um SGBD

Desta forma, os tipos de dados a serem retornados por um SGBD são conhecidos. Na coluna idade de uma pessoa física, sabe-se que somente valores inteiros e maiores do que zero serão retornados. Na coluna data de nascimento, sabe-se que somente dados no formato de data serão retornados. Portanto, não há o aspecto da imprevisibilidade de informação presente em dados textuais.

Já documentos em linguagem natural não possuem garantia alguma de integridade. Além disso, documentos deste tipo, ainda que dentro de um mesmo contexto, podem apresentar enormes diferenças estruturais entre si. Considerando dados referentes a um *curriculum vitae*, por exemplo, pode-se ter um pequeno texto informal descrevendo dados pessoais e experiência profissional ou, no extremo oposto, um documento organizado em seções e subseções, com *links* para dados das empresas e instituições onde a pessoa trabalhou.

A justificativa para tal fato decorre da própria natureza destes dados. Dados *Web*, por exemplo, apresentam uma organização bastante heterogênea (MARKOV & LAROSE, 2007), que pode variar de um texto sem nenhuma formatação até um conjunto de registros bem formatados. Em geral, dados textuais referem-se a massas de informação, quase sempre digitalizadas, e que não possuem rígida estrutura de dados. Exemplos deste tipo de dados são áudio, vídeo e texto livre, como por exemplo, o corpo de um e-mail.

Como Mineração de Dados lida com dados que já estão em formato estruturado, a maior parte das suas rotinas de pré-processamento concentra-se em duas tarefas: limpeza e integração de dados (ZHU & DAVIDSON, 2007). Sistemas de Mineração de Textos não submetem aos seus algoritmos de

descoberta de conhecimento coleções de textos despreparadas (GOMES, 2008). Em Mineração de Textos, operações de pré-processamento possuem como objetivo a identificação e a extração de características representativas de documentos para que estes possam ser representados adequadamente de maneira estruturada.

Contudo, embora a principal diferença entre Mineração de Dados e Mineração de Textos ocorra na apresentação da informação de trabalho, não significa que estes dois processos sejam completamente distintos. Ambos os processos são baseados em exemplos coletados em uma imensa base de dados e utilizam técnicas de **Aprendizado de Máquina**[6] semelhantes. Além disso, em grande parte dos casos o processo de Mineração de Textos ocorre por uma simples transformação de textos em dados estruturados, nos quais as técnicas já conhecidas de Mineração de Dados podem ser aplicadas sem qualquer restrição, não sendo necessário o entendimento de características específicas da língua em que se aplica o processo de Mineração de Textos.

Entretanto, em virtude da grande centralidade da linguagem natural nos processos de Mineração de Textos, a utilização da rica informação semântica presente em qualquer linguagem pode ser utilizada em proveito do processo de obtenção de conhecimento a partir de dados textuais. Para isto, fez-se necessário buscar avanços em áreas da Ciência que se relacionam com tratamento da linguagem como Ciência Cognitiva, Processamento de Linguagem Natural e Recuperação de Informação, dentre outras. Atualmente, muitas abordagens aos processos de Mineração de Textos tiram proveito da vasta informação linguística de suas coleções de documentos.

Independente da utilização, ou não, de dados linguísticos, da mesma forma que em Mineração de Dados, antes de ser iniciado um processo de Mineração de Textos, faz-se imprescindível aquisição de conhecimento básico sobre o domínio da aplicação no qual será realizada a mineração. Entender o domínio dos dados é naturalmente um pré-requisito para a obtenção de conhecimento útil (REZENDE, 2005).

Além disso, todo processo de Mineração de Textos visa um objetivo, ou seja, a realização de uma tarefa, e que precisa ser definido antes do início da

[6] Do termo em inglês, *Machine Learning –ML*.

mineração, pois, de acordo com este, todo o processo de Mineração de Textos será orientado. Em (REZENDE, 2005), algumas questões importantes sobre um processo de Mineração de Dados são sugeridas. Estas mesmas questões, citadas abaixo, também são aplicáveis ao processo de Mineração de Textos:

- *"Quais são os objetivos do processo?";*
- *"Quais critérios de desempenho são importantes?";*
- *"O conhecimento extraído deve ser compreensível a seres humanos ou um modelo tipo caixa-preta é apropriado?"*

Em (CARRILHO, 2007), as principais tarefas de Mineração de Textos são abordadas minuciosamente.

2.2. Principais Elementos

O principal elemento de um processo de Mineração de Textos é a coleção de documentos, pois constitui o conjunto de dados sob o qual este processo é realizado. Um documento, elemento básico de uma coleção, é definido como uma unidade discreta de dados em formato textual, como por exemplo, uma página *web* ou um *e-mail* (KONCHADY, 2006). Em geral, Mineração de Textos lida com enormes coleções de documentos, e é esta característica que, principalmente, torna impossível, em tempo hábil, a análise desta imensa base de dados por humanos (FELDMAN & SANGER, 2007).

Em alguns cenários, essa coleção de documentos pode ser estática, ou seja, o conjunto de documentos selecionado permanece inalterado tanto em elementos, como em conteúdo. Entretanto, em grande parte dos casos, essa coleção de documentos é dinâmica, podendo ter elementos incluídos, excluídos e até mesmo alterados, o que demanda desafios ainda maiores para um sistema de Mineração de Textos. Um bom exemplo deste segundo caso é a própria *Web* brasileira. A Figura 3 ilustra os dados obtidos em um estudo recente sobre a Internet no Brasil em que se constata que quase metade dos *sites* hospedados possui conteúdo dinâmico (MODESTO, PEREIRA, ZIVIANI, CASTILLHO, & BAEZA-YATES, 2005):

Figura 3 – Sites brasileiros quanto à frequência de modificação do conteúdo

É importante também ressaltar aspectos sobre a organização de uma coleção. Usualmente, os itens de uma coleção são documentos do mundo real, e são arranjados de acordo com as características que melhor os representem. Muitos são os critérios que podem ser levados em consideração na organização de uma coleção, como, por exemplo, a separação por tipo de documento (*e-mail*, memorando, currículo), mas a predominância do assunto abordado em cada documento como critério de organização é notória. Ao preparar uma coleção de documentos, até mesmo a simultaneidade de um mesmo documento pertencer a diversas coleções é possível. Na Figura 4, em um sistema fictício de Mineração de Textos para área médica, há duas coleções de documentos organizadas pelo assunto. Um documento que retrate o diagnóstico de um paciente com suspeitas de uma doença X e que foi tratada com um medicamento Y pode pertencer, simultaneamente, à coleção de documentos referentes à doença X, bem como à coleção de textos referentes ao medicamento Y.

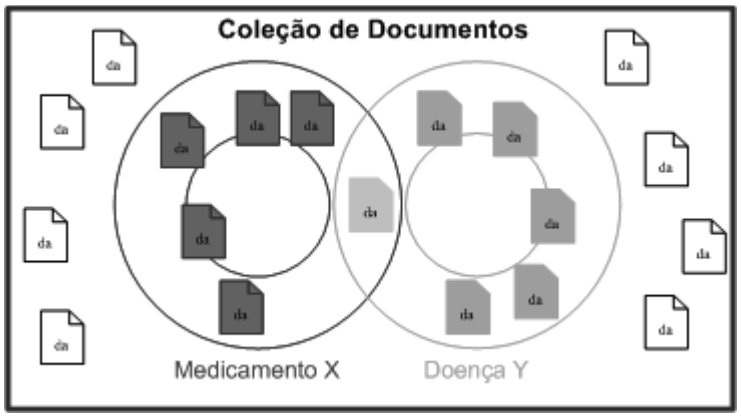


Figura 4 - Coleções de documentos com elementos em comum

2.3.
Documentos textuais são estruturados

Apesar de muitas vezes denominado não estruturado, um documento texto pode ser visto, sob muitas perspectivas, como um objeto estruturado. Especialmente, sob o enfoque da Linguística, mesmo um simples documento apresenta abundantes estruturas semânticas e sintáticas, ainda que estas estejam implícitas no texto. Elementos tipográficos, como pontuações, letras maiúsculas, números e outros caracteres especiais, ajudam a definir subcomponentes de um documento: parágrafos, títulos, datas, autores e outras informações. Até mesmo a sequencia das palavras pode definir características importantes de um documento. No exemplo da Figura 5, podemos visualizar importantes estruturas sintáticas presentes em um simples trecho de texto.

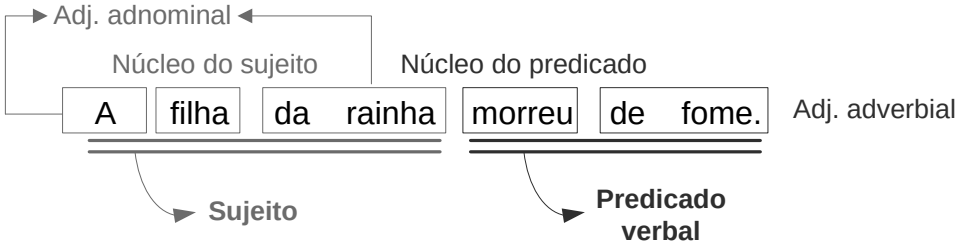


Figura 5 – Algumas estruturas sintáticas de um trecho de texto

Ainda que documentos textuais não possuam estruturas de dados definidas, é incorreto denominá-los não estruturados, pois, como visto, estes documentos possuem ricas estruturas sintáticas e semânticas. Atualmente, documentos textuais que apresentem poucos detalhes tipográficos são denominados fracamente estruturados.

Existem também documentos que fornecem mais informações sobre a sua estrutura do que aquela provida pelo texto. Documentos com extensivos e consistentes elementos de formatação (*tags*), estrutural ou visual, em que metadados podem ser facilmente inferidos, são denominados semiestruturados (SHOLOM, INDURKHYA, ZHANG, & DAMERAU, 2005) (FELDMAN & SANGER, 2007). Por exemplo, em uma mensagem de *e-mail*, informações sobre

remetente, destinatários e assuntos podem ser facilmente extraídas pela forma estrutural deste tipo de documento. Documentos em linguagem XML também podem ser considerados como semiestruturados (POWEL, 2007), quando apresentam diagramação estrutural que releva características adicionais sobre a sua estrutura.

Entretanto, não é o tipo de documento ou a linguagem de formatação do mesmo que o classifica em fracamente estruturado ou semiestruturado. Até mesmo documentos XML podem ser considerados como fracamente estruturados, pois, muitas vezes, não apresentam elementos que possam ajudar a inferir qualquer informação adicional sobre o texto que apresentam. Na Figura 6, há a representação de dois documentos na linguagem XML. Porém, somente um deles pode ser considerado semiestruturado, pois apresenta *tags* em suas definições que permitem a interpretação de informações adicionais sobre o conteúdo, o que não seria possível se o mesmo estivesse em formato livre.

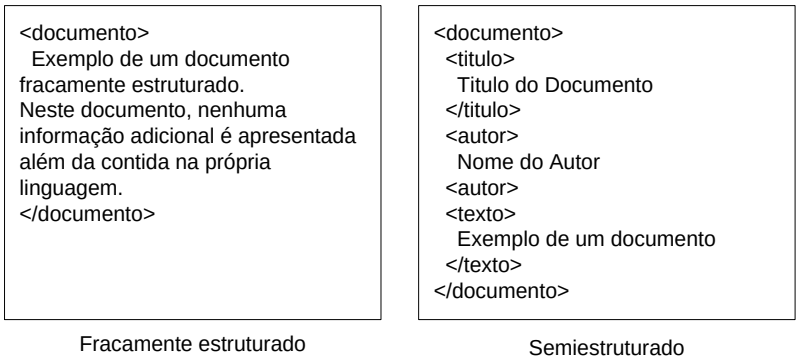


Figura 6 – Documentos em formatos fracamente estruturado e semiestruturado (respectivamente)

2.4. Características representativas de um documento

As operações de pré-processamento envolvidas em Mineração de Textos possuem como objetivo realçar os diferentes elementos presentes em um documento em linguagem natural para transformá-lo de uma representação estrutural irregular e implícita a uma representação explicitamente estruturada (FELDMAN & SANGER, 2007). Entretanto, dado o número potencialmente enorme de palavras, frases, sentenças, elementos tipográficos e *tags* de layout que,

até mesmo um simples e pequeno documento pode apresentar, uma tarefa essencial para qualquer processo de Mineração de Textos é a identificação do conjunto mais simples de características de um documento que pode representá-lo como um todo. Este conjunto de características recebe a denominação de modelo de representação, e cada documento em uma coleção é representado pelo conjunto de características que o seu modelo de representação possui. Os modelos de representação utilizados em Mineração de Textos serão explicados, com maiores detalhes, no capítulo 4.

Apesar de grandes esforços para desenvolver um modelo de representação eficiente, cada documento de uma coleção é, em geral, composto por um enorme número de características. Este enorme número de características afeta todo o processo de Mineração de Textos e influi principalmente na desempenho e design de um sistema de Mineração de Textos.

Problemas relacionados à alta dimensionalidade estão mais presentes em sistemas de Mineração de Textos do que em sistemas de Mineração de Dados. Isto está relacionado ao fato das inúmeras possibilidades existentes para a seleção de características de um documento com informações textuais (KONCHADY, 2006).

Outro fator relevante no aspecto de representação de um documento é a ausência de muitas características comuns a todos os documentos. Esta representação esparsa muitas vezes dificulta o descobrimento de padrões que são facilmente encontrados em tarefas de Mineração de Dados.

Ao selecionar quais características de cada documento serão utilizadas para construir o seu modelo de representação, há dois objetivos essenciais:

- O primeiro objetivo é determinar uma quantidade suficiente de características que permita representar os documentos sem que haja perda significativa de suas informações semânticas, o que, quase sempre, resulta em um grande número de características selecionadas.
- O segundo objetivo, por outro lado, é determinar o menor número possível de características para que os modelos de representação criados sejam computacionalmente eficientes.

Embora muitos critérios possam ser utilizados na seleção dos tipos de características que irão representar um documento, as três seguintes são as mais utilizadas:

- Caracteres: componentes individuais que são responsáveis pela formação de blocos com um nível semântico maior, como palavras e termos. Em geral, são utilizados junto com a posição em que ocorrem no texto. Abordagens que utilizam a combinação de um número predefinido de caracteres, como por exemplo, bigrama ou trigrama, são mais comuns do que a utilização de um único caractere. Embora a construção de um modelo de representação que utilize estas características seja considerada o mais próximo da realidade, a alta dimensionalidade, decorrente da escolha desta característica para representá-lo, torna impeditiva a utilização de diversas técnicas computacionais.
- Palavras: palavras retiradas de um documento constituem a menor unidade capaz de representar algum valor semântico. Por esta razão, é a característica mais utilizada para a construção de um modelo de representação de um documento. Entretanto, termos multipalavras como, por exemplo "casa da moeda", podem perder seus valores semânticos quando separados. Geralmente, para construir um modelo de representação de um documento baseado em palavras, seleciona-se somente aquelas que são mais representativas, eliminando-se *stopwords* (item "3.2.2"), caracteres simbólicos, dentre outros.
- Termos: termos podem ser compostos por uma única palavra ou por um conjunto de palavras que exprimem, por completo, a semântica que era desejada no texto. A extração de termos, na maioria das vezes, é auxiliada por um dicionário de palavras, o que permite identificar os termos que são compostos por mais de uma palavra.

No exemplo da Figura 7, as diferenças na utilização destes dois últimos critérios de seleção do tipo de características representativas de um documento podem ser visualizadas. No modelo baseado em palavras, houve perda semântica e um número maior de elementos. Porém, o modelo baseado em termos exigiu que o sistema tivesse conhecimento prévio dos termos que são compostos por mais de uma palavra.

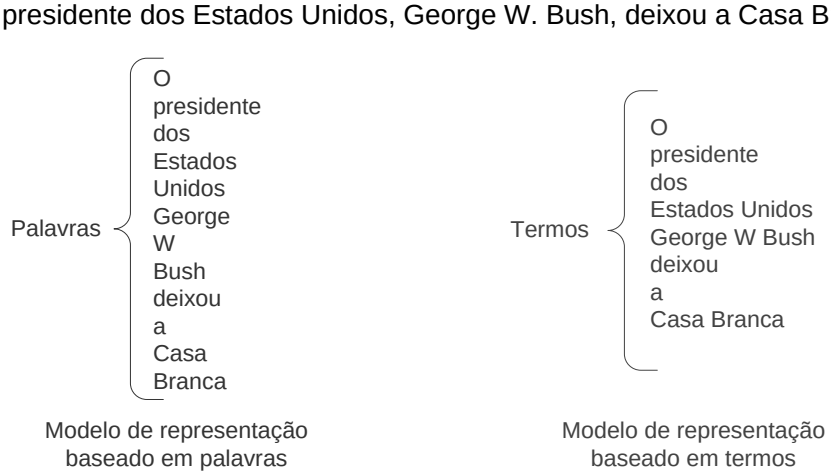


Figura 7 – Modelos de representação baseados em palavras e termos

2.5. Abordagens ao processo de Mineração de Textos

Bases de dados textuais fornecem dados semânticos e estatísticos em seu conteúdo. Diversas formas de abordagem aos dados textuais podem ser empregadas. As duas abordagens mais utilizadas são: a análise estatística, que é baseada na frequência de ocorrência dos termos nos textos, e a análise semântica, que é baseada na funcionalidade de cada termo no texto. Ambas as abordagens podem ser utilizadas sozinhas ou em conjunto.

2.5.1. Análise Estatística

Na Análise Estatística, a importância dos termos está diretamente ligada à frequência de ocorrência destes nos textos (SILVA A. A., 2007). Informações sobre contextualização, precedência ou sucessão de outros termos não são consideradas. Baseada no aprendizado estatístico, a principal vantagem desta abordagem é permitir a sua utilização em qualquer idioma.

2.5.2.
Análise Semântica

Na Análise Semântica, há a utilização da rica informação semântica, presente em qualquer linguagem, em proveito do processo de obtenção de conhecimento a partir de dados textuais. Mais do que considerar apenas aspectos estatísticos no tratamento de textos, a abordagem por Análise Semântica considera com grande centralidade a linguagem natural nos processos de Mineração de Textos.

Com o emprego de técnicas, estas baseadas no Processamento de Linguagem Natural, capazes de avaliar e identificar a funcionalidade correta de um determinado termo em uma sentença, é possível obter a verdadeira importância do mesmo em seu contexto, possibilitando aumento da qualidade dos resultados produzidos. Conforme (SILVA A. A., 2007), o emprego desse tipo de análise justifica-se pela melhoria em qualidade da Mineração de Textos quando incrementado de um processamento linguístico mais complexo.

A Tabela 3 resume as áreas de conhecimento mais envolvidas com os dois tipos de análise (CARRILHO, 2007). A Tabela 4 resume as principais características das duas abordagens. Na próxima seção, as áreas de conhecimento são explicadas de forma sucinta.

Tabela 3 - As duas abordagens para a Análise de Textos e suas principais Áreas de Conhecimento

Análise Estatística	**Análise Semântica**
Aprendizado de Máquina	Aprendizado de Máquina
Estatística	Ciência Cognitiva
Inteligência Computacional	Inteligência Computacional
Mineração de Dados	Mineração de Dados
Recuperação de Informação	Processamento de Linguagem Natural
Web Mining	*Web Mining*

Tabela 4 - As principais características de cada uma das abordagens para a Análise de Textos

Análise Estatística	**Análise Semântica**
Utilizável em qualquer idioma.	Necessita conhecimento específico do
Modelos com simples implementação e	idioma que será objeto de análise.

conhecidos na literatura. Descarta qualquer valor semântico presente nos textos.	Utiliza a informação semântica dos textos, tal como humanos.

2.6. Áreas correlatas a Mineração de Textos

Mineração de Textos é um campo multidisciplinar. Para o tratamento de textos e obtenção de conhecimento presente neles, fez-se necessário buscar e empregar avanços, técnicas e conceitos de diversas áreas como Ciência Cognitiva, Processamento de Linguagem Natural, Aprendizado de Máquina, Estatística, Recuperação de Informação e, principalmente, Mineração de Dados, da qual teve seu ponto de partida. A Figura 8 ilustra a demanda de ferramentas em outras áreas da Mineração de Textos.

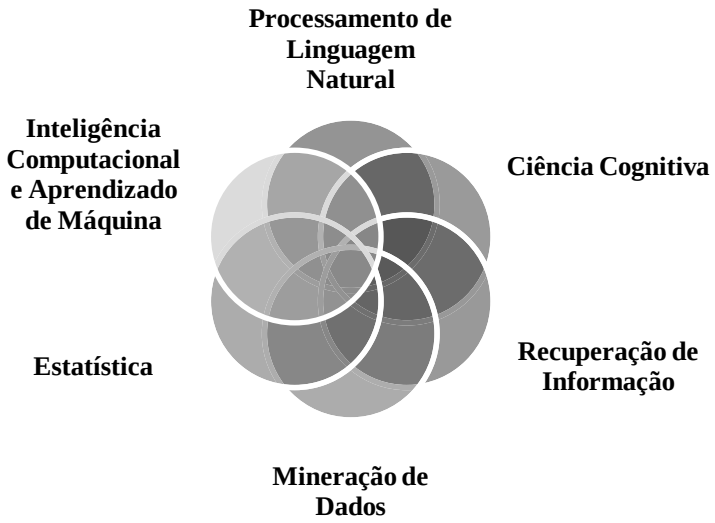


Figura 8 – Multidisciplinaridade da Mineração de Textos

2.6.1. Ciência Cognitiva

Ciência Cognitiva consiste em um conjunto de esforços interdisciplinares visando compreender, cientificamente, a mente e sua relação com o cérebro

humano. Entender a mente humana requer numerosos métodos e teorias, por isso, desta área fazem parte a Psicologia, a Filosofia, a Inteligência Artificial, a Neurociência e a Linguística (PINKER, 1998). As Neurociências colaboram na parte referente às estruturas cerebrais, a Psicologia, com as teorias de funcionamento da mente, a Filosofia, através da Lógica e da Epistemologia, a Linguística, com o exame da linguagem e a Inteligência Artificial, com os modelos de máquinas reais ou teóricas que poderiam simular o funcionamento do cérebro ou de suas partes.

O legado de contribuições da Ciência Cognitiva é enorme. Redes Neurais (RN) são exemplos de inteligência artificial conexionista, isto é, baseada na estrutura física e/ou biológica do cérebro; e que adquirem conhecimento através da experiência. Constituem um dos grandes exemplos dos avanços desta área.

Mineração de Textos busca, nesta área, principalmente, entender processo de formação da fala e da escrita.

2.6.2. Processamento de Linguagem Natural

O Processamento de Linguagem Natural (PLN) é o conjunto de métodos formais utilizados para analisar textos e gerar frases escritas em um idioma humano (ARANHA C. N., 2007). Normalmente computadores estão aptos a compreender instruções escritas em linguagens de computação, que seguem uma forte estrutura sintática, mas possuem muita dificuldade em entender comandos escritos em uma linguagem humana. Isso se deve ao fato das linguagens de computação serem extremamente precisas, contendo regras fixas e estruturas lógicas bem definidas que permitem ao computador saber exatamente como proceder a cada comando. Já em um idioma humano uma simples frase normalmente contém ambiguidades, nuances e interpretações que dependem do contexto, do conhecimento do mundo, de regras gramaticais, culturais e de conceitos abstratos (INSITE, 2001).

O objetivo final do Processamento de Linguagem Natural é fornecer aos computadores a capacidade de entender e compor textos. Entender um texto significa reconhecer o contexto, fazer análise sintática, semântica, léxica e

morfológica, criar resumos, extrair informação, interpretar os sentidos e até aprender conceitos com os textos processados.

Em Mineração de Textos, técnicas de PLN são utilizadas principalmente na fase de pré-processamento. Tarefas como identificação de classes gramaticais de termos, reconhecimento de entidades e até mesmo redução da dimensionalidade de representação de documentos são auxiliadas por PLN.

2.6.3. Aprendizado de Máquina

Aprendizado de Máquina é uma área de Inteligência Artificial (IA) cujo objetivo é o desenvolvimento de técnicas computacionais sobre o aprendizado, bem como a construção de sistemas capazes de adquirir conhecimento de forma automática (BISHOP, 2007).

Extrair conhecimento de bases de dados pode envolver, entre outras coisas, a utilização de algoritmos de Aprendizado de Máquina capazes de generalizar os exemplos encontrados em uma grande massa de dados na forma de regras de alto nível, isto é, compreensíveis ao ser humano.

Para que seja possível o aprendizado, um sistema de IA deve ser capaz de realizar três tarefas (RUSSELL & NORVIG, 2004):

- Armazenar conhecimento;
- Aplicar o conhecimento armazenado para resolver problemas;
- Adquirir novo conhecimento.

No modelo simples de aprendizagem de máquina representado pela Figura 9, o ambiente fornece alguma informação para um elemento de aprendizagem. O elemento de aprendizagem utiliza, então, esta informação para aperfeiçoar a base de conhecimento, e finalmente, o elemento de desempenho utiliza a base de conhecimento para executar a sua tarefa.

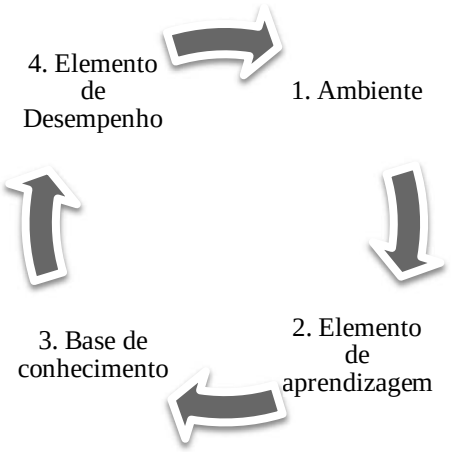


Figura 9 – Modelo simples de aprendizagem de máquina

Na área de Mineração de Textos, o Aprendizado de Máquina é utilizado para a realização de diversas tarefas como a classificação automática de documentos, na qual se destacam as **Máquinas de Vetor de Suporte**[7] (BURGES, 1998) e Classificadores Bayesianos (MCCALLUM & NIGAM, 1998) (KIM, HAN, RIM, & MYAENG, 2006), bem como, para a identificação de classes gramaticais, na qual **Cadeias de Markov Escondidas**[8] apresentam ótimos resultados (HARPER & THEDE, 1999) (SEYMORE, MCCALLUM, & ROSENFELD, 1999).

2.6.4. Estatística

Estatística é a ciência que, por meio de teorias probabilísticas, tem por objetivo a coleção, análise e interpretação de dados numéricos a respeito de fenômenos coletivos ou de massa, bem como a indução das leis a que tais fenômenos cabalmente obedecem e, ainda, a representação numérica e comparativa, em tabelas ou gráficos, dos resultados da análise desses fenômenos (SPIEGEL, 2003). Desta forma, a Estatística busca modelar a aleatoriedade e a incerteza de forma a estimar ou possibilitar a previsão de fenômenos futuros, conforme o caso.

[7] Do termo em inglês, *Support Vector Machines – SVM.*

[8] Do termo em inglês, *Hidden Markov Models – HMM.*

Há alguns anos a Estatística vem sendo utilizada no ramo da Computação. Muitos programas de e-mail modernos realizam a filtragem de *spams*[9] por meio do emprego de classificadores probabilísticos, como o classificador de Bayes. O classificador de Bayes é baseado na aplicação do Teorema de Bayes: a ideia principal é que a probabilidade de um evento A dado um evento B depende não apenas do relacionamento entre os eventos A e B, mas também da probabilidade simples da ocorrência de cada evento envolvido. Embora de simples formulação, como pode ser visto na Equação 1 , classificadores de Bayes apresentam ótimos resultados na categorização de documentos quanto ao assunto (MCCALLUM & NIGAM, 1998). Outras tarefas de Mineração de Textos são baseadas na Estatística, como por exemplo, a seleção de amostras de dados, o cálculo de aproximações, taxas de erro, médias e desvios, bem como as validações de hipóteses e conhecimentos adquiridos ao final do processo de mineração.

Teorema de Bayes

$$\mathrm{P}(A/B) = P(B/A) \times \frac{P(A)}{P(B)}$$

Equação 1 - Teorema de Bayes

2.6.5. Recuperação de Informação

Recuperação de Informação (RI) é a área da computação que lida com o armazenamento de documentos, geralmente textuais, e a recuperação automática de informação associada a eles (BAEZA-YATES & BERTIER, 1999) (MANNING, RAGHAVAN, & SCHÜTZE, 2007). De uma forma simplificada, Recuperação de Informação lida com documentos, termos de indexação e as expressões de buscas dos usuários.

[9] Mensagem eletrônica não-solicitada, geralmente, com fins publicitários. Abreviação em inglês de "*spiced ham*".

Sistemas de RI foram originalmente usados para gerenciar a explosão da informação na literatura científica na segunda metade do século XX. Muitas universidades e bibliotecas públicas usam estes sistemas para prover acesso a livros, jornais, periódicos e outros documentos.

Com a explosão demográfica da *Web*, técnicas de Recuperação de Informação passaram a ser utilizadas em máquinas de buscas**.** As máquinas de buscas surgiram logo após o surgimento da internet, com a intenção de prestar um serviço extremamente importante: a localização de qualquer informação na *Web*, apresentando os resultados de uma forma organizada**,** como um meio de prover a localização do conteúdo desejado. Atualmente, Google, Yahoo e MSN são as máquinas de buscas globais mais acessados e realizam milhões de consultas diárias em seus servidores.

Por tratar de coleções de documentos enormes, Mineração de Textos recorre à área de Recuperação de Informação para obter documentos relevantes ao tópico que será trabalhado, de forma rápida e eficiente. Este assunto será abordado em detalhes no capítulo 4.

2.6.6. Mineração de Dados

O processo de Descoberta de Conhecimento em Bases de Dados (do inglês, *Knowledge Discovery in Databases,* KDD) consiste na exploração de grandes quantidades de dados à procura de padrões consistentes e potencialmente úteis, como regras de associação ou sequencias temporais, para a obtenção do conhecimento implícito presente nestes dados (GOLDSCHMIDT & PASSOS, 2005) (TAN, STEINBACH, & KUMAR, 2005). O termo processo implica que existem vários passos envolvendo preparação de dados, procura por modelos, avaliação de conhecimento e refinamento, todos estes repetidos em múltiplas iterações (FERRO & LEE, 2001).

Mineração de Dados (do inglês, *Data Mining*) é um dos principais passos no processo de *KDD* e utiliza muitas técnicas de análises estatísticas sofisticadas e algoritmos de Aprendizagem de Máquina, para descobrir padrões escondidos e relações em bases de dados.

É importante ressaltar a grande diferença existente entre processamento de dados e Mineração de Dados. O primeiro lida com operações comuns em uma base de dados: recuperação, exclusão, inserção e atualização de dados. O segundo encontra informação desconhecida nas bases de dados (KONCHADY, 2006).

Mineração de Textos buscou na área de Mineração de Dados os principais algoritmos e técnicas para a descoberta de conhecimento relevante em dados. O processo de *KDD* serviu como referência para a criação de uma metodologia de Mineração de Textos baseada em etapas bem definidas. Para uma referência completa e prática acerca do processo de *KDD* vide (GOLDSCHMIDT & PASSOS, 2005).

3
Metodologia de Mineração de Textos

Neste capítulo são analisadas e discutidas as etapas de uma metodologia para Mineração de Textos. Embora Mineração de Textos possa ser empregada para a realização de diversas tarefas, como por exemplo, para a classificação automática de textos (KUDO & MATSUMOTO, 2004), segundo (ARANHA C. N., 2007), todo processo de Mineração de Textos consiste de cinco etapas, encadeadas nesta ordem: coleta de documentos, pré-processamento, indexação, mineração e análise. Na Figura 10 são exibidos o encadeamento destas etapas e principais atividades realizadas em cada uma delas.

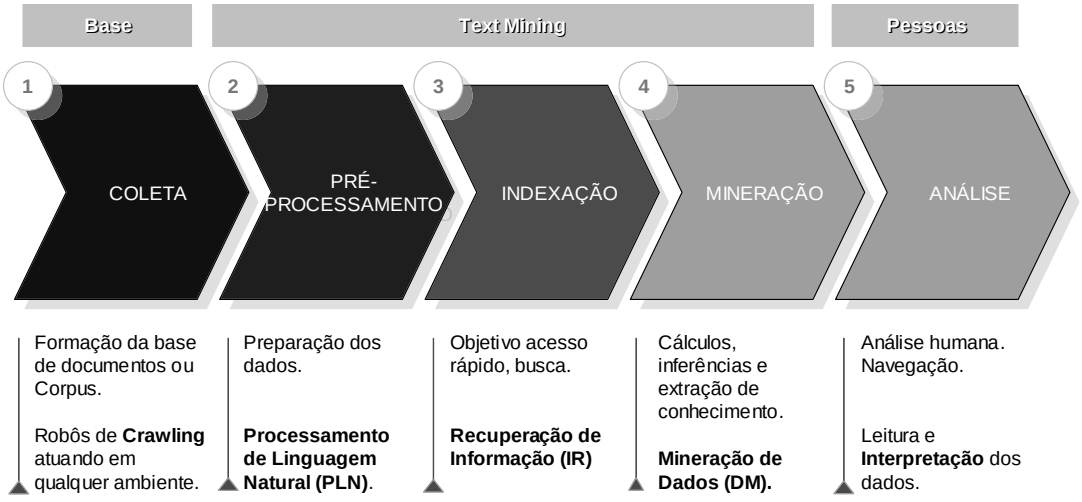


Figura 10 – Linhas cronológica das etapas de um processo de Mineração de Textos (por Aranha)

A primeira etapa a ser realizada é a de Coleta, cujo objetivo é a formação da coleção de documentos, elemento básico de qualquer processo de Mineração de Textos.

Em seguida, inicia-se a etapa de Pré-processamento. É neste momento que os documentos, obtidos na fase anterior, são submetidos a inúmeras operações capazes de obter uma forma de representá-los estruturadamente.

Após o Pré-Processamento, inicia-se a fase de Indexação. Indexação é o processo responsável pela criação de estruturas auxiliares denominadas índices e que garantem rapidez e agilidade na recuperação dos documentos e seus termos.

Uma vez indexados, documentos e termos são submetidos a algoritmos de Aprendizado de Máquina e de Estatística para que seja realizada a extração de conhecimento dos mesmos. A extração de conhecimento tem a finalidade de descobrir padrões úteis e desconhecidos presentes nos documentos.

Finalizando o processo de Mineração de Textos, há a etapa de Análise. Na etapa de Análise é realizada a avaliação e interpretação de todo o conhecimento obtido pelo processo.

3.1. Coleta de Dados

A primeira etapa de um processo de Descoberta de Conhecimento em Textos é a Coleta de Dados (SHOLOM, INDURKHYA, ZHANG, & DAMERAU, 2005) (KONCHADY, 2006) (ARANHA C. N., 2007) (FELDMAN & SANGER, 2007). Esta etapa envolve a seleção dos textos que irão compor a Coleção de Documentos, elemento básico de qualquer processo de Mineração de Textos. É interessante ressaltar que documentos devem ser relevantes ao domínio da aplicação do conhecimento a ser extraído, pois a seleção de documentos irrelevantes para fazer parte da Coleção de Documentos pode prejudicar o processo de Mineração de Textos, além de aumentar a dimensionalidade dos dados desnecessariamente.

Quanto à origem, documentos podem ser obtidos das mais diversas fontes, mas, em geral, são três os principais ambientes de localização dos mesmos: pastas de arquivos encontradas no disco rígido de usuários, tabelas de diversos bancos de dados e a *Web*.

Na *Web*, a coleta de dados pode ser realizada de forma automatizada através de *crawlers* (HEATON, 2002). Um *crawler* é um robô que visita todo e qualquer documento *Web* disponível e repassa as informações coletadas para outro componente responsável pela indexação desses documentos. Atualmente, visando paralelismo e escalabilidade, a arquitetura mais moderna de varredura na *Web*

utiliza vários desses robôs de forma distribuída trabalhando de maneira cooperativa (WEN, 2006).

(SOARES, 2008) propõe uma metodologia de coleta inteligente de dados na *Web* baseada em técnicas de Mineração Textos para a construção de um *crawler* focado. Um *crawler* focado é altamente efetivo na construção de coleções de documentos de qualidade sobre tópicos específicos e oriundos da *web*, usando modestos computadores "caseiros" (DOM, CHAKRABARTI, & BERG, 1999).

3.2. Pré-Processamento

Sistemas de Mineração de Textos não submetem aos seus algoritmos de descoberta de conhecimento coleções de textos despreparadas (GOMES, 2008). Uma vez realizada a Coleta de Dados, o próximo passo é a preparação dos textos para que os mesmos possam ser manipulados pelos algoritmos de Mineração de Textos. Esta segunda etapa denomina-se Pré-Processamento e é responsável por criar uma representação do texto mais estruturada, capaz de alimentar algoritmos de Máquinas de Aprendizado (GONÇALVES, SILVA, QUARESMA, & VIEIRA, 2006), muitos destes também utilizados em Mineração de Dados.

É na etapa de Pré-processamento que é criado o modelo de representação dos documentos, ou seja, a transformação de textos em dados estruturados. Existem diversos modelos para representação estruturada de documentos textuais na literatura de RI, entretanto, o mais utilizado em Mineração de Textos é o **Modelo de Espaço Vetorial**[10] (SILVA A. A., 2007), e será visto em detalhes no item 4.3.1.2.

Uma vez criado o modelo de representação dos textos, é necessário que este seja computacionalmente tratável, e para isto são realizadas algumas operações de Análise de Dados que visam selecionar somente as características que melhor expressam o conteúdo dos textos. Este processo é ilustrado na Figura 11.

[10] Do termo inglês, *Vector Space Model*.

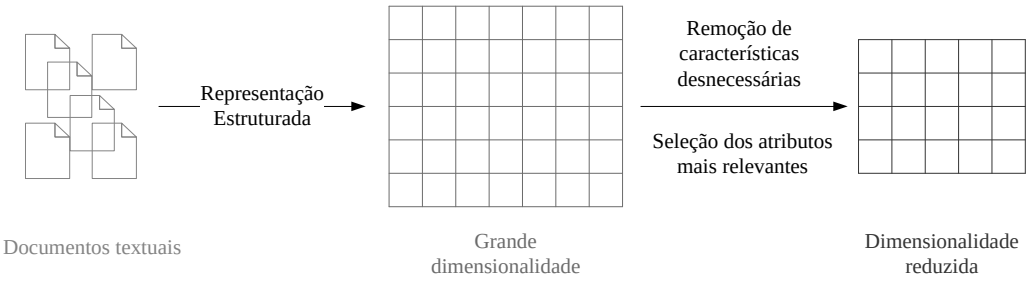


Figura 11 – Processo de representação estruturada de um texto

Pré-processar textos é, por muitas vezes, o processo mais oneroso da metodologia de Mineração de Textos, uma vez que não existe uma única técnica que possa ser aplicada para a obtenção de uma representação satisfatória em todos os domínios, sendo necessária a realização de muitos experimentos empíricos para se chegar à representação adequada (CARRILHO, 2007).

3.2.1. Tokenização

O primeiro passo de uma operação de Pré-processamento é a **tokenização**[11] ou **atomização**[12] e sua execução tem como finalidade seccionar um documento textual em unidades mínimas, mas que exprimam a mesma semântica original do texto. O termo *token* é utilizado para designar estas unidades, que geralmente correspondem a somente uma palavra do texto, porém há casos em que estas unidades textuais não podem ser consideradas palavras ou apresentam mais de uma palavra: “21/10/2007”, “PM”, “R$100,00” e “couve-flor”.

O processo de tokenização é auxiliado pelo fato das palavras serem separadas por caracteres de controle de arquivo ou de formatação, tais como espaços ou sinais de pontuação, que em alguns casos podem ser considerados *tokens* delimitadores (FELDMAN & SANGER, 2007). A criação de *tokens* de um texto baseada em seus delimitadores é uma estratégia simples e que apresenta

[11] Do inglês, *tokenization*.

[12] Alguns autores de língua portuguesa utilizam o termo atomização para fazer referência à tarefa de tokenização (FINATTO, 2005) (LINGUATECA, 2007).

bons resultados. Entretanto, a tarefa de identificação de *tokens*, que é relativamente simples para o ser humano, pode ser bastante complexa de ser executada por um computador. Este fato, segundo (CARRILHO, 2007), é atribuído ao grande número de papéis que os delimitadores podem assumir. Por exemplo, o "ponto" pode ser usado para marcar o fim de uma sentença, mas também é usado em abreviações e números ("A Av. Brasil possui 58 km de extensão.").

Em processos de Mineração de Textos que são assistidos por um dicionário de dados, este pode ser utilizado a fim de verificar as sequencias de caracteres que compõem um termo e validar sua existência, bem como corrigir possíveis erros ortográficos.

Um algoritmo muito utilizado para verificar a corretude de um termo é o de Distância de Edição (FONSECA & REIS, 2002), pois informa quantas operações (deleção, substituição ou inserção de caracteres) são necessárias para que um termo seja transformado em outro. O exemplo abaixo exibe os passos necessários para transformar o termo "casas" em "massa", definindo a distância de edição em três:

1. casas ^ masas — substituição de 'c' por 'm'
2. masas ^ mass — eliminação de 'a'
3. mass ^ massa — inserção de 'a'

A Figura 12 ilustra a metodologia proposta em (KONCHADY, 2006) para a identificação de *tokens*, que, com o uso de dicionários de dados e regras de formação de palavras, procura manter o mesmo nível semântico apresentado pelos *tokens* de um texto antes do processo de tokenização.

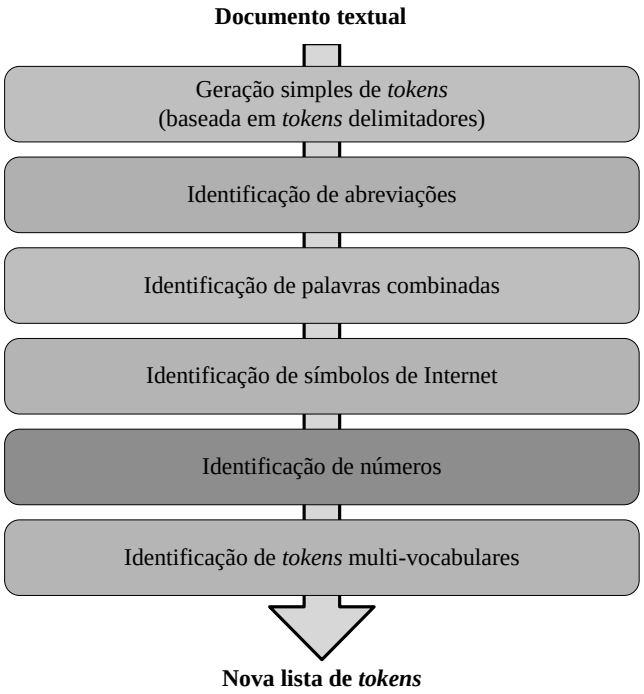


Figura 12 - Metodologia de identificação de tokens proposta por KONCHADY

3.2.2. Remoção de *stopwords*

Buscando sempre tornar possível o processamento computacional de textos, uma vez realizado o processo de tokenização, o passo seguinte é a identificação do que pode ser desconsiderado nos passos posteriores do processamento dos dados. É a tentativa de retirar tudo que não constitui conhecimento nos textos.

Em um documento, existem muitos *tokens* que possuem pouco valor semântico, sendo úteis apenas para o entendimento e compreensão geral do texto. Estes *tokens* são palavras classificadas como ***stopwords***[13]e fazem parte do que é chamado de *stoplist* de um sistema de Mineração de Textos (BASTOS, 2006).

Geralmente, fazem parte de uma *stoplist* termos como conjunções, preposições, pronomes e artigos, pois são considerados termos de menor relevância, ou seja, sua presença pouco contribuiu para a determinação do valor semântico de um documento. Uma *stoplist* bem elaborada permite a eliminação de

[13] Em "http://linguateca.di.uminho.pt/Paulo/stopwords/" há uma lista de trezentas *stopwords* da Língua Portuguesa.

muitos termos irrelevantes, tornando mais eficiente o resultado obtido pelo processo de Mineração de Textos. Normalmente, 40 a 50% do total de palavras de um texto são removidas com uma *stoplist* (SILVA A. A., 2007). A Figura 13 ilustra o exemplo de um processo de tokenização seguido por outro de remoção de *stopwords*.

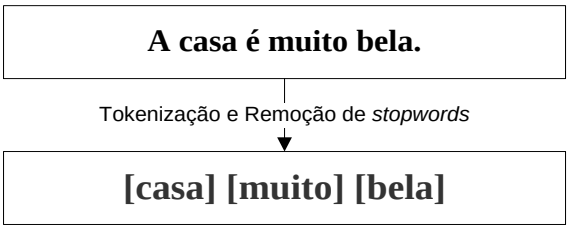


Figura 13 - Processo de tokenização seguido por remoção de stopwords

3.2.3. Processamento de Linguagem Natural

O uso de técnicas de Processamento de Linguagem Natural em Mineração de Textos tem o objetivo de identificar a real importância de cada termo em determinados contextos, possibilitando um ganho na qualidade dos resultados produzidos. O PLN é utilizado para agregar valores semânticos que poderão beneficiar o processo de descoberta de conhecimento em etapas posteriores. Embora muitas abordagens ao processo de Mineração de Textos não façam uso de PLN, a sua utilização tem incrementado os resultados obtidos e justificado o esforço computacional adicional, como em (ARANHA C. N., 2007).

3.2.3.1. Identificação de Colações

A ordem ou disposição dos vocábulos em uma sentença pode incrementar ou, até mesmo, alterar totalmente o significado de alguns termos. Palavras que expressam essa relação são conhecidas como colocações. Certas colocações têm a mesma forma que termos complexos, como por exemplo, “proteção ambiental” e “proteção do meio ambiente”. Em outras, a ordem dos elementos componentes de sua formação pode

variar, como por exemplo, “grande amigo” e “amigo grande”. Muitos são os tipos e possibilidades de uma colação e é interessante que ao criar um *token*, este seja composto por todo este conjunto de palavras que traduz uma ideia diferente.

3.2.3.2. Identificação de Classes Gramaticais

Entende-se por classe gramatical como a forma de classificação de um termo segundo seu significado e função (CEGALLA, 2005). Na língua Portuguesa há dez classes gramaticais, sendo que seis são variáveis (substantivo, artigo, adjetivo, numeral, pronome e verbo) e quatro, invariáveis (advérbio, preposição, conjunção e interjeição). A identificação de classes de palavras ou **etiquetagem de classes gramaticais**[14] presentes em uma sentença facilita o entendimento desta e muitas vezes soluciona alguns problemas simples de ambiguidade.

Cadeias de Markov Escondidas (HARPER & THEDE, 1999) e **TBL**[15] (BRILL, 1995) têm sido utilizados com sucesso em tarefas de identificação de classes gramaticais. Outro método simples para a execução desta tarefa é a simples consulta a dicionário de dados, porém, este método não é dotado de nenhuma heurística que garanta a identificação correta de uma classe gramatical quando uma mesma palavra pode assumir mais de uma classe gramatical.

3.2.3.3. Análise de Discurso

Análise de Discurso, também conhecida por Resolução de Referências (RUSSELL & NORVIG, 2004), é a interpretação de um pronome ou de um sintagma nominal que se refere a um objeto presente no texto. Em linguística, a referência a algo que já foi apresentado é chamado de anáfora. Descobrir anáforas em um texto é o principal objetivo da tarefa de Análise de Discurso e exige conhecimentos sobre o contexto e partes anteriores do texto.

[14] Do termo inglês, *part of speech tagging.*

[15] Acrônimo do termo inglês, *Transformation Based Learner*.

Contexto é a situação histórico-social de um texto, envolvendo não somente as instituições humanas, como ainda outros textos que sejam produzidos em volta e que se relacionem. Todo contexto envolve elementos tanto da realidade do autor quanto do receptor e a análise destes elementos ajuda a determinar o seu sentido (FOUCAULT, 2002). No exemplo da Figura 14, a anáfora só pode ser bem definida quando levado em consideração o contexto.

Figura 14 - Reconhecimento de anáfora com informações do contexto

Para entender que "ele" na segunda sentença faz referência a Luís, é necessário identificar que a primeira sentença menciona duas pessoas e que Luís é quem representa o papel de cliente, logo, é provável que ele faça um pedido em vez do frentista.

3.2.3.4. Lematização

Qualquer documento textual apresenta muitas palavras flexionadas nas mais diversas formas. Na língua Portuguesa, um substantivo pode ser flexionado em gênero, número e grau, e apresentar o mesmo valor semântico. O processo de formação de palavras é, na maior parte das vezes, realizado pela derivação de radicais, resultando na criação de palavras que também exprimem o mesmo significado (CEGALLA, 2005).

Lematização ou *stemming* é o processo de reduzir ao radical original palavras derivadas ou flexionadas deste. O principal objetivo da utilização de um processo de *stemming* é reduzir a grande dimensionalidade das aplicações de Mineração de Textos, pois, com a remoção de prefixos e sufixos de palavras derivadas de um mesmo radical, e que, antes, seriam consideradas como *tokens* distintos, obtém-se um único *token* para a representação de todas elas.

Embora utilizem técnicas de linguística, o que os torna dependentes do idioma, algoritmos de *stemming* não buscam chegar às regras básicas da

linguística do idioma ao radicalizar uma palavra, mas sim, melhorar o desempenho das aplicações, o que pode resultar em tipos de erros que devem ser observados e controlados durante a execução do *stemming*:

- *Overstemming*: ocorre quando o conjunto de caracteres removidos de uma palavra não faz parte de uma derivação ou flexão desta, mas, sim, de seu radical. A ocorrência deste erro pode fazer com que se obtenha um mesmo radical para palavras distintas.
- *Understemming*: ocorre quando os caracteres resultantes do processo de *stemming* ainda fazem parte de uma derivação ou flexão da palavra original. A ocorrência deste erro pode fazer com que sejam obtidos radicais distintos para palavras de mesma origem.

No exemplo da Figura 15, temos o exemplo destes dois tipos de erros comuns no processo de *stemming*. O algoritmo de *stemming* utilizado, neste exemplo, foi o de Porter (PORTER, 1980).

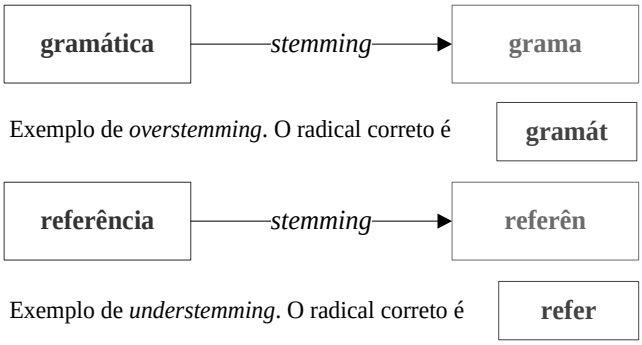


Figura 15 – Erros de um processo de stemming: overstemming e understemming

O algoritmo de *stemming* mais utilizado na Língua Portuguesa é o de Porter (PORTER, 1980). Além deste, há outros algoritmos de *stemming* disponíveis na literatura e que são apresentados a seguir. Embora a maior parte destes algoritmos tenha sido desenvolvida para o idioma inglês, é frequente encontrar adaptações de alguns deles para os mais diversos idiomas:

- Método do Stemmer S: Constitui um dos mais simples métodos de *stemming*. Apenas uns poucos finais de palavras (“ies”, “es" e “s”)

são removidos, com exceções. Embora, simples, este algoritmo é utilizado em razão do seu elevado nível de conservadorismo.

- Método de Porter: O funcionamento do algoritmo de *stemming* de Porter (PORTER, 1980) consiste na identificação e substituição das diversas inflexões e derivações de uma mesma palavra por um mesmo radical. Como, em geral, termos que derivam de um mesmo radical possuem significados semelhantes, consegue-se reunir em um único *token* a importância de todas as suas derivações, como no exemplo da Figura 16. Este algoritmo remove cerca de sessenta sufixos diferentes.

Figura 16 – Derivações de um mesmo radical identificadas pelo algoritmo de Porter

- Método de Lovins: O algoritmo de *stemming* desenvolvido por Lovins (LOVINS, 1968) é capaz de remover cerca de duzentos e cinquenta sufixos diferentes em um único passo. Sensível ao contexto, este algoritmo remove no máximo um sufixo por palavra, geralmente, o mais longo. Embora vários sufixos não sejam removidos por este método, este é o mais agressivo dos três algoritmos de *stemming* apresentados.

3.2.3.5. Reconhecimento de Entidades Nomeadas

Segundo (SUTTON & MCCALLUM, 2006)**,** o **Reconhecimento de Entidades Nomeadas**[16] é o problema de identificar e classificar nomes próprios em textos, incluindo localizações, tais como Brasil e Rio de Janeiro; pessoas, tais com Dilma e Miriam; organizações, tais como Ministério da Educação e Ministério da Cultura; tempo, tais como uma data ou um período de duração; além de outras entidades.

No exemplo da Tabela 5, a sentença contém quatro entidades diferentes: Mário como pessoa, trezentos como número (quantidade), Petrobras como organização e 2006 como tempo (data). A saída marcada com *tags* foi realizada pelo software ENAMEX que foi desenvolvido para a "Message Understanding Conference" na década de noventa.

Tabela 5 - Marcação de *tags* para Reconhecimento de Entidades Nomeadas

Entrada	Mário comprou 300 ações da Petrobras em 2006.
Saída	<ENAMEX TYPE="PERSON">Mário</ENAMEX> comprou <NUMEX TYPE="QUANTITY">300</NUMEX> ações da <ENAMEX TYPE="ORGANIZATION">Petrobras</ENAMEX> em <TIMEX TYPE="DATE">2006</TIMEX>.

O grande desafio deste problema é que muitas entidades nomeadas, mesmo em grandes conjuntos de treinamento, possuem pouca frequência; portanto, o sistema deve identificar tais entidades utilizando apenas o contexto em que esta é empregada. Outra peculiaridade é que a mesma entidade pode apresentar classificações diferentes dependendo do contexto em que se encontra.

A Tabela 6 exibe um exemplo em que, na primeira sentença a entidade Brasil representa um local, enquanto na segunda sentença, essa entidade representa uma organização (Governo Brasileiro).

[16] Do termo inglês, Named Entity Recognition (NER)

Tabela 6 - Exemplo de classificações distintas de uma mesma entidade

Sentença 1	Visitarei o *Brasil* no próximo ano.
Sentença 2	A proposta apresentada pelo *Brasil* na ONU...

Inicialmente os algoritmos de NER eram baseados em regras escritas manualmente que indicavam a existência de uma entidade em determinado contexto. Atualmente, aprendizado supervisionado tem sido a técnica predominante na tarefa de reconhecimento de entidades nomeadas (NADEAU; SEKINE, 2007).

3.2.3.6. Análise Sintática

A análise sintática tem como objetivo examinar a estrutura de um período e das orações que compõem esse período (NEVES, 2012). O fato a ser considerado é a impossibilidade, em certas sentenças, de se obter sentido sem o emprego de funções gramaticais.

Na análise sintática, cada palavra ou grupo de palavras da oração é chamado de termo da oração. Um termo é classificado de acordo com a função sintática que exerce na oração.

De acordo com a Nomenclatura Gramatical Brasileira, os termos da oração podem ser:

1. Essenciais: Também chamados de fundamentais: Sujeito e Predicado.
2. Integrantes: Completam o sentido dos verbos e dos nomes:
   a. Complemento Verbal - Objeto Direto e Objeto Indireto
   b. Complemento Nominal
   c. Agente da Passiva
3. Acessórios: Desempenham função secundária (especificam o substantivo ou expressam circunstância):
   a. Adjunto Adnominal
   b. Adjunto Adverbial

c. Aposto

No exemplo abaixo, por meio da análise sintática pode-se compreender que a oração possui um predicado nominal (o verbo estar denota estado, logo é um verbo de ligação) sobre cujo sujeito simples (a manhã) é revelada uma característica (ensolarada) por meio do predicativo do sujeito (revela uma característica sobre o mesmo), pois se tem um predicado nominal.

Sentença	A manhã está ensolarada.	
Análise	Sujeito simples	a **manhã (núcleo)**
	Predicado nominal	está ensolarada
	Predicativo do sujeito	ensolarada

De maneira geral, a análise sintática, também conhecida como *parsing* consiste da utilização de dois componentes principais. Primeiramente, uma gramática contendo os fatos sintáticos da linguagem utilizada é exigida. Esta gramática servirá como base de atuação do segundo componente do processamento sintático: o analisador. Também conhecido como *parser*, o analisador compara as formalizações descritas na gramática com a sentença de entrada. O resultado é a geração da estrutura hierárquica contendo as unidades de significado da sentença.

3.3. Indexação

Após a etapa de Pré-Processamento, independente da utilização de Processamento de Linguagem Natural, documentos textuais, antes fracamente estruturados, agora possuem representação estruturada, esta, baseada em um dos diversos modelos de representação disponíveis (ver item 4.3.1). Entretanto, para que uma simples consulta seja realizada é necessário percorrer toda a coleção de documentos, analisando documento a documento, o que demanda tempo e esforço computacional.

Indexação é fase responsável por criar estruturas de dados denominadas índices, capazes de permitir que uma consulta seja realizada sem que seja

necessário analisar toda uma base de dados (MANNING, RAGHAVAN, & SCHÜTZE, 2007). Técnicas de indexação de documentos foram bastante difundidas pela demanda e crescimento da área de Recuperação de Informação desde a década de sessenta. Contudo, muitas pessoas acreditam que esta é uma área nova. Esta ideia talvez tenha surgido com a grande popularização das máquinas de buscas que tornaram possível a pesquisa do conteúdo de páginas *web*, ou seja, documentos textuais. No entanto, segundo (BAEZA-YATES & BERTIER, 1999), há aproximadamente quatro mil anos já são praticadas técnicas de catalogação manual por índices.

Semelhantes ao sumário de um livro que é uma lista detalhada, com a indicação de localização no texto, dos principais tópicos abordados no interior deste, índices são utilizados para otimizar a velocidade e o desempenho da busca por um documento relevante em relação a um termo buscado. O custo pelo ganho de tempo durante a recuperação de informação é o espaço de armazenamento computacional adicional necessário para armazenar o índice.

É importante ressaltar que a etapa de Indexação é diretamente influenciada pela etapa de Pré-Processamento, pois, todo o conteúdo que será indexado, ou não, foi determinado por esta etapa. Desta forma, quando a etapa de Pré-Processamento faz uso de PLN e, com isso, fornece características linguísticas do texto processado, a etapa de Indexação utiliza estes dados ricos em semântica na construção do índice. Além disso, a etapa de Indexação também pode fazer uso de PLN, tornando possíveis duas abordagens distintas ao processo de criação de índices: Indexação Textual ou Indexação Temática.

3.3.1. Indexação Textual

O processo de Indexação Textual é realizado pela indexação dos termos presentes em um documento. É um procedimento automático e não utiliza informações externas, como por exemplo, um dicionário de palavras. Dependendo do algoritmo utilizado na construção do índice, é possível a realização de consultas com a utilização de operadores de proximidades e operadores booleanos.

Atualmente, a técnica mais utilizada para a indexação textual é a de **índices invertidos**[17] (KONCHADY, 2006) (FELDMAN & SANGER, 2007). Em sua apresentação básica, um índice invertido é uma estrutura de dados composta de uma lista ordenada, geralmente denominada vocábulo ou vocabulário, que armazena todas as palavras distintas encontradas nos textos e os documentos em que elas ocorrem. Informações adicionais como a frequência e posição de ocorrência da palavra no texto também podem ser armazenadas. Em (FONSECA & FIDALGO, 2002), há o exemplo de um índice invertido construído com auxílio de técnicas de Processamento de Linguagem Natural, reproduzido na Figura 17. Nota-se que, neste exemplo, todas as palavras foram mantidas em sua forma singular e em caixa baixa. Além disso, foram removidas do processo de indexação todas as palavras com pouco poder discriminatório (*stopwords*).

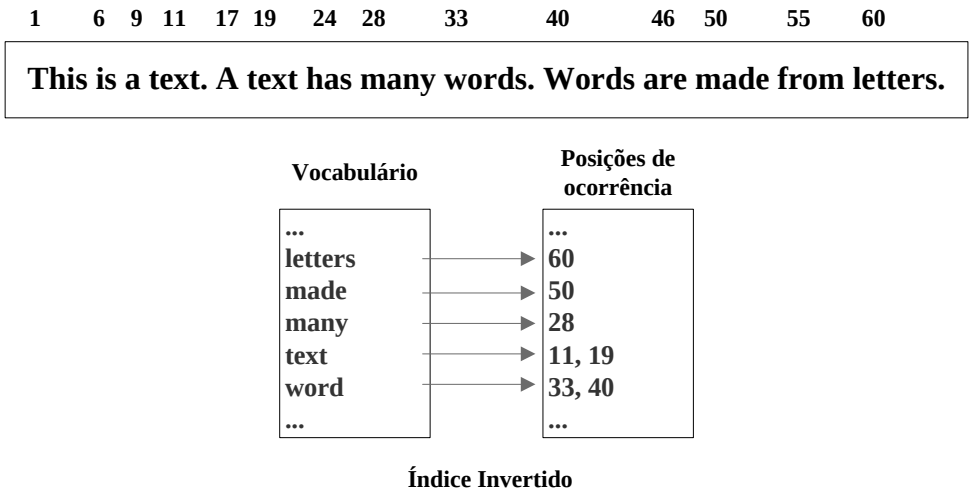


Figura 17 - Representação de um índice invertido

3.3.2. Indexação Temática

O procedimento de Indexação Temática é caracterizado pela constante consulta a um dicionário de termos. Este dicionário é conhecido pelo nome de *thesaurus* e sua função é simples: após receber uma palavra encontrada no texto, realiza uma consulta em sua base de dados e indica ao indexador o termo correto a

[17] Recebem essa denominação por inverter a hierarquia da informação. No lugar de uma lista de documentos contendo termos, tem-se uma lista de termos referenciando documentos.

ser utilizado na indexação da palavra recebida. Isto é possível devido a sua estrutura, ilustrada na Figura 18, que mapeia em único termo, este denominado termo preferido, todo um conjunto de termos sinônimos, estes denominados de termos não preferenciais. Como exemplo, podemos citar as palavras “carro”, “automóvel”, “veículo”, que poderiam ser associadas a uma única palavra que é “carro”.

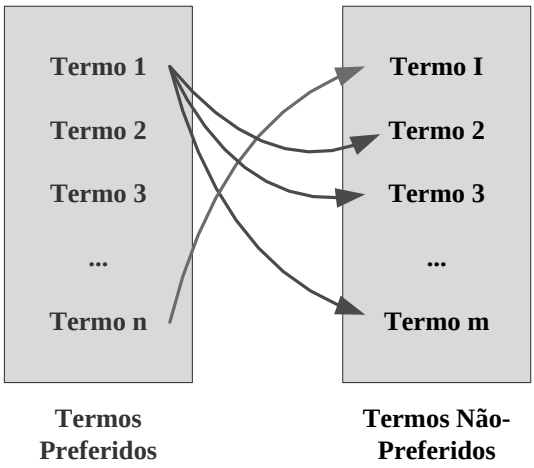


Figura 18 - Estrutura básica de um Dicionário Thesaurus

A utilização desta técnica, além de tornar o índice mais compacto, permite a localização de documentos grafados de forma diferente, mas que apresentam mesmo valor semântico. Porém, a maior dificuldade para a utilização deste mecanismo reside na criação do próprio dicionário.

3.4. Mineração

É na etapa de Mineração que ocorre a busca efetiva por conhecimentos novos e úteis a partir dos dados. Compreende a aplicação de algoritmos de Aprendizado de Máquina sobre os dados de forma a abstrair o conhecimento implícito presente nestes.

A escolha do algoritmo a ser utilizado está relacionada com o objetivo da tarefa de Mineração de Textos. Este objetivo, definido no início do processo, irá determinar quais as opções possíveis de Aprendizado de Máquina que se aplicam ao problema. Além disso, outros detalhes devem ser considerados, como por

exemplo, a necessidade ou não de que o conhecimento aprendido seja facilmente interpretável, o que pode descartar da lista de opções possíveis algoritmos de Aprendizado de Máquina do tipo "caixa preta", como Redes Neurais, pois a compreensão da rede neural resultante de um processo de aprendizado não é uma tarefa trivial e requer esforço adicional para a extração das regras aprendidas por esta técnica, como em (SETIONO & LEOW, 1998).

Outro fator restritivo é a necessidade de urgência do processo. Alternativas que, embora possam apresentar excelentes resultados, muitas vezes precisam ser desconsideradas em razão do elevado tempo de processamento computacional necessário para o treinamento destas.

Em (GOLDSCHMIDT & PASSOS, 2005) é ressaltado que a dificuldade de escolha de um algoritmo de aprendizado apropriado é intensificada na medida em que surjam novos algoritmos com o mesmo propósito, aumentando a diversidade de alternativas, mas, que, geralmente, a escolha dos algoritmos se restringe às opções conhecidas pelo analista do processo, que muitas vezes, deixa de considerar muitas alternativas promissoras.

3.5. Análise

A etapa de Análise, algumas vezes chamada de Pós-Processamento, abrange o tratamento do conhecimento obtido na etapa de Mineração, através da análise, visualização e interpretação deste. Tal tratamento tem como objetivo viabilizar a avaliação da utilidade do conhecimento descoberto (ZHU & DAVIDSON, 2007).

Para analisar o resultado de um processo de Mineração de Textos são utilizadas métricas de avaliação de desempenho. O objetivo de uma métrica de desempenho é graduar a execução de uma tarefa. As principais métricas de avaliação utilizadas em Mineração de Textos foram adotadas da área de Recuperação de Informação e são baseadas na noção de relevância. Um documento é considerado relevante quando possui importância para o tópico considerado. Precisão, Abrangência e Média-F são as métricas de desempenho mais utilizadas e serão abordadas nos itens "3.5.1", "3.5.2" e "3.5.3", respectivamente.

Porém, de acordo com o objetivo de cada processo de Descoberta de Conhecimento em Textos, métricas de avaliação de desempenho diferentes das citadas acima devem ser utilizadas. Por exemplo, uma tarefa de Sumarização não será bem avaliada por medidas como Abrangência, Precisão ou Medida-F.

Muitas vezes, de forma a facilitar análise do conhecimento obtido, podem ser utilizados métodos de transformação de dados que consistem, basicamente, na conversão de uma forma de visualização para outra (KANTARDZIC, 2002). Da mesma forma que as medidas de desempenho, diferentes estratégias de visualização podem ser empregadas, cada qual mais adequada ao objetivo do processo de Mineração de Textos. Por exemplo, para melhor visualizar as regras obtidas por um processo de Mineração de Textos é comum a conversão de árvores de decisão em regras ou vice-versa. No exemplo abaixo, as regras na Tabela 7 são visualizadas sob a forma de árvore de decisão na Figura 19.

Tabela 7 – Visualização das regras para concessão de empréstimos em uma tabela

Montante	Salário	Possui Conta?	Empréstimo
médio	baixo	indiferente	**não**
médio	alto	indiferente	**sim**
alto	indiferente	sim	**sim**
alto	indiferente	não	**não**

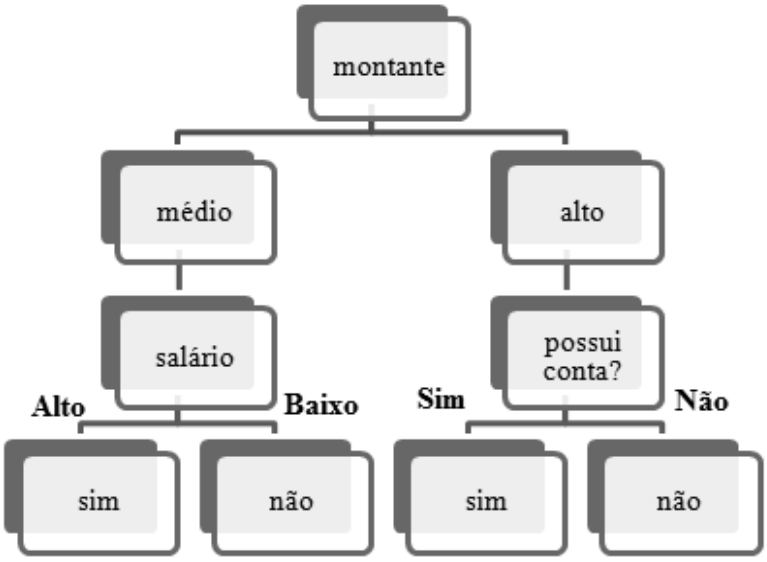


Figura 19- Visualização das regras para concessão de empréstimo em uma árvore de decisão

Usualmente, as técnicas de visualização de dados facilitam a compreensão do conhecimento obtido. Além de árvores de decisão e regras, outros recursos podem ser utilizados: gráficos bi ou tridimensionais, planilhas, tabelas e cubos de dados.

3.5.1. Precisão

Para um dado conjunto de itens recuperados, precisão é definida como a proporção entre o número de itens relevantes recuperados e o número total de itens recuperados (Equação 2).

$$\textbf{Precisão} = \frac{\mathbf{N_{recuperados} \cap N_{relevantes}}}{\mathbf{N_{recuperados}}}$$

Equação 2 - Fórmula da métrica de desempenho “Precisão”

3.5.2. Abrangência

Para um dado conjunto de itens recuperados, a abrangência é definida como a proporção entre o número de itens relevantes recuperados e o número total de itens relevantes no sistema em questão (Equação 3).

$$\textbf{Abrangência} = \frac{\mathbf{N_{recuperados} \cap N_{relevantes}}}{\mathbf{N_{relevantes}}}$$

Equação 3 - Fórmula da métrica de desempenho “Abrangência”

3.5.3.
Medida-F

A Medida-F, Média Harmônica ou *F-Mean* é a combinação de Abrangência e Precisão em uma única métrica (Equação 4) Esta função assume valores no intervalo [0, 1]. Quando o valor retornado é zero, não há documentos relevantes no conjunto de dados medido. Quanto mais próximo de um o resultado da fórmula, maior relevância possui o conjunto de dados testado.

Algumas vezes, a Medida-F é definida à priori; nesses casos, busca-se encontrar a relação ideal entre Abrangência e Precisão para que o resultado da métrica seja obtido. Também pode ser utilizada com pequenas variações nos termos visando atribuir pesos diferentes para Abrangência e Precisão.

$$\mathbf{Medida - F} = \frac{\mathbf{2}}{\frac{\mathbf{1}}{\mathbf{Abrangência}} + \frac{\mathbf{1}}{\mathbf{Precisão}}}$$

Equação 4 - Fórmula da métrica de desempenho "Medida-F"

3.5.4.
Precisão x Abrangência

Em resumo, Precisão é a porcentagem dos itens recuperados que são relevantes. Abrangência é a porcentagem dos itens relevantes que foi recuperada. Por exemplo, uma consulta com valor de Precisão igual a 0.70 significa que 70 por cento dos itens recuperados são relevantes, ao passo que uma consulta com valor de Abrangência igual a 0.70 possui apenas 70 por cento dos documentos que são ou poderiam ser relevantes.

Abrangência e Precisão são frequentemente objetivos contraditórios (BAEZA-YATES & BERTIER, 1999), pois, na medida em que se deseja obter mais itens relevantes (aumentando o nível de abrangência), mais itens irrelevantes também são recuperados (diminuindo o nível de precisão) (SILVA F. R., 2007). Estudos empíricos sobre o desempenho mostram uma tendência de declínio da precisão na medida em que a abrangência aumenta. Em (BUCKLAND & GEY, 1944) é comprovado que obter altos índices de Precisão ou Abrangência é

possível a qualquer sistema, porém não simultaneamente. Na Figura 20 é ilustrado o gráfico de equilíbrio entre Abrangência e Precisão. A linha contínua representa o mapeamento real da relação Abrangência-Precisão em uma coleção de documentos e a linha tracejada representa a relação ideal entre estas duas métricas.

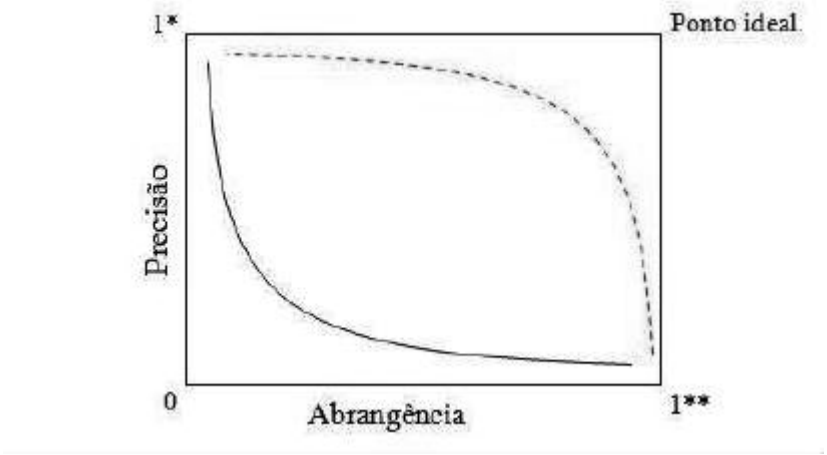


Figura 20 – Gráfico de compensação entre precisão e abrangência

Com base nesta relação, de acordo com (SILVA F. R., 2007), pode-se determinar que um Sistema A é melhor do que o Sistema B, segundo as métricas de Precisão e Abrangência destes sistemas, se, em todos os pontos de Abrangência, o valor da precisão do Sistema A for maior do que do Sistema B. Caso isso não aconteça, as médias dos valores de precisão para valores de abrangência selecionados são calculadas e comparadas.

4
Recuperação de Informação

No presente capítulo são apresentados os fundamentos da área de Recuperação de Informação utilizados em Mineração de Textos, como por exemplo, os modelos de representação de documentos e as principais operações envolvidas nestes processos.

4.1.
Introdução

Recuperação de Informação lida com a representação, armazenamento, organização e acesso a itens de informação (BAEZA-YATES & BERTIER, 1999) (MANNING, RAGHAVAN, & SCHÜTZE, 2007). O conceito de itens de informação, neste contexto, refere-se ao tratamento diferenciado que estes objetos, geralmente documentos textuais, recebem: todos possuem muita ou pouca relevância. Julga-se um documento relevante quando este supre a necessidade de informação do usuário. Relevância, a característica central de Sistemas de Recuperação de Informação, é o que distingue Sistemas de Recuperação de Informação de Sistemas de Recuperação de Dados.

Recuperação de Dados busca meios eficientes de recuperar objetos baseado em um critério simples: dado o conjunto de termos desejado, encontrar todos os documentos que atendam ao critério booleano determinado. E isto é suficiente para muitas aplicações, como por exemplo, Sistemas Gerenciadores de Bancos de Dados. Mas, para um usuário que deseja informações sobre um determinado tópico, a consulta baseada em termos nem sempre trará somente bons resultados, ou seja, nem sempre será relevante. A Tabela 8 apresenta algumas das diferenças entre Recuperação de Dados e Recuperação de Informação (RIJSBERGEN, 1979).

Tabela 8 - Comparação entre Recuperação de Dados x Recuperação de Informação

Características	**Recuperação de Dados**	**Recuperação de Informação**
Comparação	Exata	Aproximada
Dados	Fortemente estruturados	Fracamente estruturados
Inferência	Dedução	Indução
Modelo	Determinístico	Probabilístico
Ling. Consulta	Artificial	Natural
Esp. da Consulta	Completa	Incompleta

Usuários de Sistemas de RI estão mais interessados na recuperação de informação associada a documentos do que na recuperação dos termos presentes nestes. Com o crescimento do volume de publicações, ao longo dos anos, foram desenvolvidas técnicas específicas para a área de Recuperação de Informação com o intuito de atender às necessidades dos usuários.

A ferramenta mais importante para auxiliar o processo de recuperação de informação é denominada índice. Índices são estruturas de dados associadas à parte textual dos documentos, e, portanto, indicam o local onde a informação desejada pode ser localizada. Segundo (BAEZA-YATES & BERTIER, 1999), há aproximadamente quatro mil anos já são praticadas técnicas de catalogação manual por índices.

Recuperação de Informação, antes, interesse de poucos, agora, é uma das áreas que mais tem recebido atenção de cientistas e pesquisadores. Contribuiu principalmente para isto a explosão demográfica da *Web* que é de longe o maior acervo de dados do mundo (CHAKRABARTI, 2003). E na *Web*, prevalecem os documentos hipertextos que, em sua essência, constituem o objeto de estudo de RI: documentos textuais.

A crescente complexidade dos objetos armazenados e o grande volume de dados exigem processos de recuperação cada vez mais sofisticados. Diante deste quadro, recuperação de informação apresenta a cada dia, novos desafios e se configura como uma área de significância maior (CARDOSO, 2000).

Porém, além de muito sucesso, a *Web* também trouxe novos desafios para a área de RI. Por ser um ambiente onde impera a cultura liberal e informal de propagação de conteúdo, encontrar informação relevante na *Web* tem sido cada

vez mais difícil, motivando também uma grande pesquisa em torno da Recuperação de Informação na Internet, em especial, na *Web*.

4.2. Histórico da área de Recuperação de Informação

A área de Recuperação de Informação pode ser cronologicamente dividida em três fases. A primeira fase compreende as décadas de 50 e 60. A fase seguinte situou-se entre as décadas de 70 e 80. Por último, a terceira fase que compreende a década de 90 aos dias atuais.

4.2.1. 1ª Fase – Décadas de 50 e 60

Sistemas de Recuperação de Informação foram originalmente utilizados para gerenciar a explosão de conteúdo da literatura científica na segunda metade do século XX (RIJSBERGEN, 1979). Bibliotecas estão entre as primeiras instituições a adotarem Sistemas de RI.

Em suas primeiras versões, Sistemas de RI funcionavam como um simples catálogo eletrônico. O processo de indexação era basicamente manual e os documentos eram indexados somente pelos termos principais de um dicionário de sinônimos criado para este propósito: um dicionário *thesaurus* (ver item "3.3.2"). A ideia deste conceito é simples: permitir a indexação somente do termo principal sempre que o próprio termo ou termos sinônimos estiverem presentes em um texto, evitando assim, que a escolha de um sinônimo ou outro possa impedir a localização do documento. Já na década de 60, Sistemas de RI deram início ao processo de indexação automática, porém, somente título e abstract eram processados. Surgiram também os primeiros algoritmos de busca textual.

4.2.2. 2ª Fase – Décadas de 70 e 80

Neste período, houve grandes avanços na área tecnológica, o que resultou em aumento significativo do poder computacional da época, permitindo, também,

a evolução de diversos sistemas, inclusive dos Sistemas de RI. Avanços como a indexação automática de todo o conteúdo e o desenvolvimento de funcionalidades adicionais de pesquisas foram possíveis.

RI – unida a área de Linguística – iniciou os primeiros estudos de Processamento de Linguagem Natural possibilitando a criação de um sistema simples de perguntas-respostas (BAEZA-YATES & BERTIER, 1999). Foi também nesta fase que o modelo de representação de documentos mais utilizado foi criado: o Modelo de Espaço Vetorial.

4.2.3.
3ª Fase – Década de 90 em diante

Nesta fase, o grande crescimento da *Web* e a necessidade de informação relevante neste ambiente colocaram em foco novamente a área de RI. Inicialmente, técnicas tradicionais de Sistemas de RI foram utilizadas, porém, grandes foram os problemas encontrados na adaptação destas técnicas:

- Escalabilidade das soluções: escalabilidade, neste contexto, indica a capacidade de preparo para a manipulação de grandes quantidades dados, seja esta relacionada ao poder de processamento ou armazenamento.
- Velocidade de atualização das páginas-*web*: a incrível velocidade de modificação do conteúdo dos *web sites* torna difícil manter um índice operacional e coerente sem saber a frequência de atualização dos documentos indexados.
- Velocidade de acesso aos documentos: em razão da sua distribuição geográfica mundial, a Web contém documentos nas mais diversas localidades. O acesso e indexação destes documentos exigem a disponibilidade dos mesmos, além do tempo necessário para que toda a informação neles seja transferida de um local para outro.

Atualmente, novas tecnologias estão sendo desenvolvidas para explorar as peculiaridades de um documento hipertexto e toda a sua relação n*a Web.*

4.3.
Recuperação de Informação Clássica

Recuperar informação é o propósito básico de qualquer sistema Recuperação de Informação. Baseada em índices, a recuperação de informação nestes sistemas obedece à arquitetura ilustrada na Figura 21.

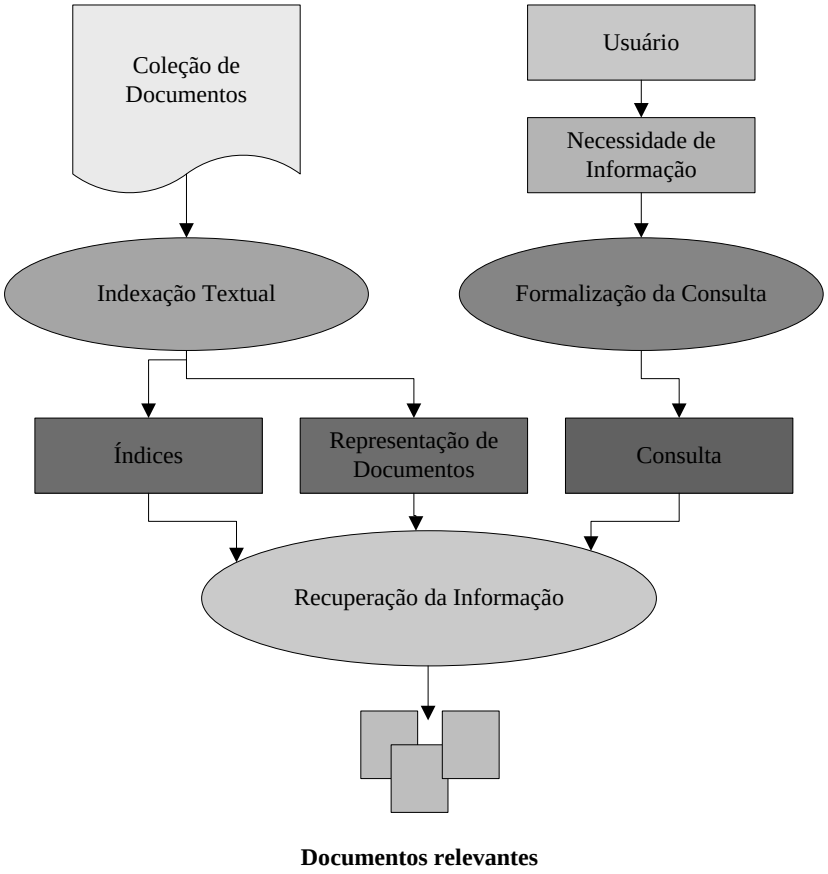


Figura 21 - Sistema Clássico de Recuperação de Informação

Neste modelo, duas entidades justificam a existência de um sistema de RI: a coleção de documentos, estes, geralmente textos, e o usuário com necessidade de informação. Os outros componentes decorrem destes.

A consulta é a representação formalizada da necessidade de informação do usuário em uma linguagem entendida pelo sistema. O processo de especificação da consulta geralmente é uma tarefa difícil. Há frequentemente uma distância semântica entre a real necessidade do usuário e o que ele expressa na consulta formulada (CARDOSO, 2000). Essa distância é gerada pelo limitado

conhecimento do usuário sobre o universo de pesquisa e pelo formalismo da linguagem de consulta.

Assim que formalizada, a consulta é processada junto aos documentos, que estão representados pelos seus respectivos modelos de representação textuais, e, em seguida, a resposta à necessidade de informação na coleção de documentos é exibida ao usuário. O processo de recuperação consiste na geração de uma lista de documentos recuperados para responder a consulta formulada pelo usuário. Os índices construídos para uma coleção de documentos são usados para acelerar esta tarefa. Além disso, a lista de documentos recuperados é classificada em ordem decrescente de um grau de similaridade entre o documento e a consulta (CARDOSO, 2000).

Os modelos de representação textuais utilizados em Sistemas de RI podem ser vistos como uma representação fortemente estruturada dos textos. E, como todo documento é considerado um conjunto de termos ou *tokens*, esta nova representação estruturada é baseada na presença ou ausência destes termos ou *tokens*.

Quando todo o conjunto de *tokens* de um documento é utilizado para representá-lo tem-se uma indexação textual completa ou *full text indexing*. Porém, embora a indexação textual completa seja aquela que forneça a visão lógica mais completa de um documento, nem sempre é possível utilizá-la, em razão do elevado custo computacional para o manuseio desta enorme quantia de dados, tornando necessário que um documento seja representado por um conjunto menor de *tokens*.

Como nem todas as palavras num texto não igualmente importantes para representá-lo semanticamente, para que seja bem representado por um conjunto menor de *tokens*, um documento pode ser submetido a sucessivos métodos de processamento textual, tais como remoção de *stopwords* e *stemming*, que visam eliminar conteúdo irrelevante do texto, permitindo que seja possível a representação lógica do mesmo. A Figura 22 ilustra algumas das possibilidades existentes em um processo de indexação (BAEZA-YATES & BERTIER, 1999).

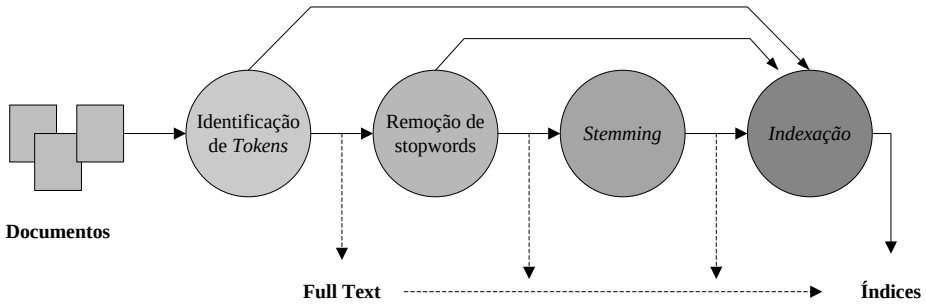


Figura 22 – Etapas possíveis no processo de Indexação de documentos textuais

Por utilizar seus próprios métodos de processamento textual, o interesse de Mineração de Textos na área de RI restringe-se às técnicas de representação e identificação de documentos. Muitos dos métodos de processamento textual utilizados em Mineração de Textos foram baseados naqueles utilizados em RI, e, portanto, foram abordados, sob o enfoque de Mineração de Textos, no item "3.2".

A seguir, serão apresentados dois modelos de representação de documentos utilizados, tanto em RI, como em Mineração de Textos: **Modelo de Recuperação Booleano**[18] e Modelo de Espaço Vetorial.

4.3.1.
Modelos de Representação de Documentos

4.3.1.1.
Modelo de Recuperação Booleano

Um dos primeiros modelos de pesquisa a ser adotado foi o Modelo de Recuperação Booleano ou, simplesmente, Modelo Booleano. Fundamentado na Álgebra Booleana e na Teoria dos Conjuntos, interpreta toda consulta como uma expressão lógica, permitindo até mesmo a utilização dos conectivos lógicos "e", "ou" e "não", e, portanto, possui critério de decisão simples para julgar a

[18] Do termo inglês, *Boolean retrieval model.*

relevância de um documento: documentos relevantes são aqueles que contêm, ou não, os termos que satisfazem a expressão lógica da consulta.

Em virtude do critério de decisão binário deste modelo não existem meios para a realização de igualdade parcial da consulta com os documentos. Portanto, também não existem critérios de graduação de relevância dos documentos encontrados, ou seja, não é possível ordenar documentos de acordo com a relevância individual de cada um.

Este modelo de representação é muito mais utilizado em Sistemas de Recuperação de Dados do que em Sistemas de Recuperação de Informação. É de fácil utilização para usuários que dominam lógica booleana, o que não ocorre na maioria dos casos.

Algumas das vantagens do modelo booleano são a excelente *performance* e a fácil implementação. Possui como principal desvantagem a dificuldade de se expressar a necessidade de informação por meio de uma expressão booleana. Outra característica ruim deste modelo é desconsiderar a frequência de ocorrência dos termos em um texto.

4.3.1.2. Modelo de Espaço Vetorial

O Modelo de Espaço Vetorial busca abordagem geométrica para resolver problemas de representação de documentos. Documentos são representados como vetores em um espaço Euclidiano *t-dimensional* em que cada dimensão corresponde a um *token* da coleção de documentos (REZENDE, 2005), ou seja, cada *token* é um eixo deste espaço Euclidiano.

Neste modelo, vetores são representados pela forma $D_i = (t_1; t_2; t_3; ...; t_n)$, em que D_i é o *i-ésimo* documento de uma coleção, e t_n o *n-ésimo token* da coleção de documentos, ou seja, para cada documento da coleção existem n *tokens*-índices que os representa (SILVA A. A., 2007), conforme ilustrado na Figura 23 . Cada *token* desta coleção de documentos está associado a sua frequência de ocorrência em cada documento, desta forma, para o documento D_i e para o token t_j, $w_{i,j} \geq 0$ representa essa associação e o tamanho do $eixo_j$ no vetor D_i. Quando o *token* j não ocorre no documento D_i, tem-se $w_{i,j} = 0$.

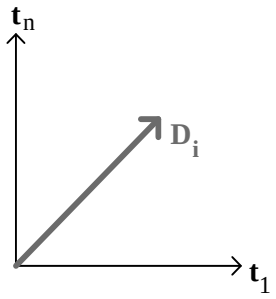


Figura 23 - Representação vetorial do documento Di no espaço n-dimensional (*n =2*)

No Modelo de Representação Vetorial, o processamento de uma consulta é realizado através de um cálculo de similaridade entre cada documento da coleção e a própria consulta, ou seja, toda consulta é também representada de forma vetorial, e através de um cálculo de similaridade entre cada documento da coleção e a consulta, obtém-se uma lista dos documentos relevantes para aquela necessidade de informação.

O Modelo do Espaço Vetorial é o modelo de representação mais utilizado em Mineração de Textos (REZENDE, 2005). Contribuem para isto a sua forma de representação, intuitiva e prática, que torna possível:

- A ponderação de termos na representação dos documentos e processamento das consultas;
- A recuperação de documentos que não possuem todos os termos definidos na consulta;
- Ordenação do resultado baseada na relevância dos documentos.

Desvantagens deste modelo de representação são a necessidade de novo processamento da coleção de documentos quando esta é alterada e a ausência de relação semântica entre os *tokens* de uma coleção.

4.3.1.3. Frequência dos termos

No Modelo de Espaço Vetorial, cada documento é representado por um vetor cujas dimensões são os termos presentes na coleção de documentos. Cada

coordenada do vetor é um termo da coleção de documentos e possui valor numérico que representa a frequência de ocorrência deste termo no documento (LOPES, 2004).

A associação de valores numéricos as coordenadas dos vetores é conhecida como **atribuição de pesos**[19] e visa atribuir maior importância aos termos que são mais relevantes. A seguir, são citadas e explicadas as medidas de atribuição de pesos mais comuns:

- Binária: Quando um termo está presente em determinado documento, é atribuído o valor *true* ou um para indicar esta ocorrência. Quando um termo está não presente em determinado documento, é atribuído o valor *false* ou zero para indicar esta ausência. Por ser muito simples, esta medida de atribuição de pesos é raramente utilizada.
- Frequência do Termo: **Frequência do Termo**[20] ou **TF**[21] é definida como o número de ocorrências de um determinado termo em um documento (SALTON & BUCKLEY, 1988). Em geral, termos presentes em muitos documentos com alta frequência não possuem caráter discriminatório para a diferenciação dos documentos de uma coleção e são considerados como uma *stopword*. É comum normalizar em um documento a frequência de seus termos, pois, sem este artifício, os documentos mais extensos de uma coleção seriam privilegiados no processo de recuperação de informação. Na Equação 5 é ilustrado o cálculo normalizado da frequência do Termo_i no Documento_j que possui *k* termos.

$$\mathrm{tf_{i,j}} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

Equação 5 - Cálculo da medida TF em um documento

[19] Do termo inglês, *weighting*.

[20] Do termo inglês, *Term Frequency*.

[21] Acrônimo de *Term Frequency*.

- TF-IDF: TF-IDF ou *Term Frequency – Inverse Document Frequency* é uma medida de atribuição de pesos que favorece termos que ocorrem em poucos documentos de uma coleção (SALTON & BUCKLEY, 1988). É utilizada para avaliar o quão importante é um termo para o documento em que ele ocorre, em relação a todos os documentos da coleção. A medida TF-IFD de um termo, ilustrada na Equação 6, é a combinação de sua medida local (TF) e global (IDF).

$$\boldsymbol{tf - idf_{i,j} = tf_{i,j} \times idf_i}$$

$$idf_i = \log \frac{|D|}{\left|\{d_j : t_i \in d_j\}\right|}$$

$$|D|, número\ total\ de\ documentos\ da\ coleção;$$
$$\left|\{d_j : t_i \in d_j\}\right|, número\ de\ documentos\ em\ que\ o\ termo\ t_i\ ocorre;$$

Equação 6 - Cálculo da medida TF-IDF em um documento

- Escore de relevância: Proposto por (WIENER, PEDERSEN, & WEIGEND, 1995) é baseado na importância que um termo possui em representar uma determinada categoria da coleção de documentos. Termos que aparecem em muitas categorias obtém valores baixos, por serem pouco discriminantes; termos que aparecem em poucas categorias recebem valores altos, podendo representar a categoria em que possui maior frequência. O escore de relevância de um termo, ilustrado na Equação 7, é dado por:

$$\boldsymbol{r_t} = \log \frac{\frac{w_{ct}}{d_c} + \frac{1}{6}}{\frac{w_{\bar{c}t}}{d_{\bar{c}}} + \frac{1}{6}}$$

Equação 7 - Cálculo do escore de relevância de um termo

Em que:

- w_{ct} é o número de documentos pertencentes a uma categoria *(c)* que contém o termo *t*;
- d_c é o número total de documentos da categoria considerada *(c)*;
- $w_{\overline{ct}}$ é o número de documentos de outras categorias que contém o termo *t*;
- $d_{\bar{c}}$ é o número total de categorias de outros documentos.
- A constante $\frac{1}{6}$ é utilizada para eliminar o problema de divisão por zero, possível de ocorrer quando um termo só está presente na categoria considerada *(c)*.

- Coeficiente de correlação: Desenvolvido por (NG, GOH, & LOW, 1997) para indicar o grau de correlação de uma palavra e um documento. Leva em conta a quantidade total de documentos de uma coleção, a quantidade de documentos em que o termo aparece e a quantidade de documentos em que o termo não aparece. A Equação 8 ilustra a definição do coeficiente de correlação entre o termo *t* e a classe *c*:

$$\boldsymbol{C_{(t,c)}} = \frac{(\boldsymbol{N_{r+}} \times \boldsymbol{N_{n-}} - \boldsymbol{N_{r-}} \times \boldsymbol{N_{n+}}) \times \sqrt{\boldsymbol{N}}}{\sqrt{(\boldsymbol{N_{r+}} + \boldsymbol{N_{r-}}) \times (\boldsymbol{N_{n+}} + \boldsymbol{N_{n-}}) \times (\boldsymbol{N_{r+}} + \boldsymbol{N_{n+}}) \times (\boldsymbol{N_{r-}} + \boldsymbol{N_{n-}})}}$$

Equação 8 - Cálculo do Coeficiente de Correlação

Em que:

- N_{r+} é o número de documentos relevantes para $C_j$que contém o termo *t*;

- N_{r-} é o número de documentos relevantes para C_j que não contém o termo *t*;
- N_{n+} é o número de documentos não relevantes para $C_j$que contêm o termo *t;*
- N_{n-} é o número de documentos não relevantes para C_j que não contêm o termo *t.*

- Ganho de informação: Métrica de atribuição de pesos proposta por (YANG & PEDERSEN, 1997), é um critério que define a qualidade de cada termo. Ele mede a quantidade de pequenos pedaços ou partições de informação obtidos para a predição da categoria através da presença ou ausência de um termo no documento. Este método é comumente utilizado no campo de aprendizagem de máquina e na construção de árvores e regras de decisão. Os autores afirmam que a categorização de textos, normalmente, possui um espaço dimensional muito grande, alcançando até dezenas de milhares de características, e é preciso calcular a qualidade do termo de maneira global. A partir de um conjunto de textos de treinamento, para cada termo único é calculado o ganho de informação. Os termos que não alcançarem um limiar predefinido serão excluídos. A ideia principal deste método é dividir o conjunto de exemplos em partições ou subconjuntos de exemplos, sendo estes subconjuntos compostos de exemplos de uma mesma classe ou similares. Ao grupo aplica-se o cálculo do ganho. O conjunto vai sendo subdividido repetidamente até que um subconjunto contenha apenas exemplos de uma única classe ou o número de exemplos seja inferior a um limite estabelecido. A conclusão é que o ganho de informação reduz os ruídos conforme o conjunto vai sendo subdividido, de forma que, no final, o último subconjunto será composto apenas por exemplos similares. A Equação 9 exibe a fórmula proposta por (MLAENIC & GROBELNIK, 1998) para o cálculo de ganho de informação:

$$G_f = P_w \times \sum_i P_{C_i|w} \times \log \frac{P_{C_i|w}}{P_{C_i}} + P_{\bar{w}} \times \sum_i P_{C_i|\bar{w}} \times \log \frac{P_{C_i|\bar{w}}}{P_{C_i}}$$

Equação 9 - Cálculo do Ganho de Informação

Cálculo do Ganho de Informação

Em que:

- G_f é o ganho de informação de característica *f*. O termo *w* é representado pela característica *f*;
- P_w é a probabilidade de ocorrer o termo *w*;
- $P_{C_i|w}$ é a probabilidade condicional de ocorrer o termo *w* na *i-ésima* classe;
- P_{C_i} é a probabilidade da *i-ésima* classe;
- $P_{\bar{w}}$ é a probabilidade de não ocorrer o termo *w*;
- $P_{C_i|\bar{w}}$ é a probabilidade condicional de não ocorrer o termo *w* na *i-ésima* classe.

4.3.1.4. Cálculo de Similaridade

No Modelo do Espaço Vetorial cada documento é representado por um vetor de *n* dimensões, em que cada dimensão é um termo distinto e presente em algum documento da coleção. A cada termo é atribuído um peso como forma de identificar a importância deste no documento e para isto são utilizadas as medidas de atribuição de pesos mencionadas acima.

Uma das técnicas mais utilizadas para obter o grau de similaridade entre documentos ou entre documentos e consultas decorre naturalmente deste modelo de representação: é através do cosseno do ângulo formado pelos vetores de representação destes objetos (BAEZA-YATES & BERTIER, 1999).

O cálculo do cosseno do ângulo entre dois vetores é ilustrado na Equação 10. Quanto mais perto de um o valor do cosseno, mais ortogonais são os vetores comparados, o que significa que existem poucos termos comuns entre os documentos. Quanto mais perto de zero o valor do cosseno, mais paralelos são os vetores comparados, o que significa que existem muitos termos comuns entre os documentos.

$$\mathbf{cs}\,\boldsymbol{\theta} = \frac{\overrightarrow{\boldsymbol{v_1}} \cdot \overrightarrow{\boldsymbol{v_2}}}{\|\overrightarrow{\boldsymbol{v_1}}\| \times \|\overrightarrow{\boldsymbol{v_2}}\|}$$

Equação 10 - Cálculo de similaridade entre documentos por meio do cosseno

5
Categorização de Textos

Neste capítulo são apresentados os fundamentos da área de Categorização de Textos, bem como um breve histórico da área. Além disso, uma pesquisa sobre os *softwares* de Mineração de Textos e suas possibilidades quanto à tarefa de Categorização.

5.1.
Introdução

Nos últimos anos, as tarefas de manipulação de documentos baseadas em conteúdo, conhecidas coletivamente como Recuperação de Informação, ganharam um status de destaque na área de Sistemas de Informação, devido ao aumento da disponibilidade de documentos em formato digital e a consequente necessidade de acessá-los e organizá-los de maneiras mais flexíveis.

Categorização de Textos (CT), a atividade de rotular textos em linguagem natural com categorias temáticas a partir de um conjunto pré-definido, é uma dessas tarefas (SEBASTIANI, 2002).

Na comunidade de pesquisa a abordagem dominante para este problema é baseada em técnicas de Aprendizado de Máquina: um processo indutivo cria um classificador que, a partir de um conjunto de documentos categorizados, aprende as características de cada categoria.

As vantagens desta abordagem sobre a heurística baseada em Engenharia do Conhecimento, que consiste na definição manual de um classificador - baseado em regras - por especialistas do domínio, são boa eficácia, economia considerável em termos de força de trabalho especializada e pouca dependência de domínio.

5.2.
Histórico da área de Categorização de Textos

A área de Categorização de Textos pode ser cronologicamente dividida em duas fases. Na primeira fase predominam as abordagens baseadas em Engenharia do Conhecimento e situou-se até o final da década 80. A fase seguinte foi marcada pelas abordagens ao problema baseadas em Aprendizado de Máquina e compreende a década de 90 aos dias atuais.

5.2.1.
1ª Fase - Até o final década de 80

CT remonta ao início dos anos 60 e até o final dos anos 80, a abordagem era baseada na Engenharia do Conhecimento, em que próprio especialista codifica o classificador através de regras que definem cada categoria. Nesse método é extremamente dependente de intervenção humana para a construção do classificador ou adaptação para outro domínio de conhecimento.

5.2.2.
2ª Fase - Década de 90 em diante

Nos anos 90, a abordagem mencionada acima começou a perder popularidade. O paradigma de Aprendizagem de Máquina, segundo o qual um processo indutivo cria um classificador que, a partir de um conjunto de documentos categorizados aprende as características de cada categoria, passou a centralizar os esforços acadêmicos.

As vantagens desta abordagem são: precisão comparável à obtida por especialistas humanos; e uma economia considerável em termos de força de trabalho especializada, uma vez que nenhuma intervenção de engenheiros do conhecimento ou especialista do domínio é necessária para a construção do classificador ou por sua portabilidade para um conjunto diferente de categorias ou domínio.

Atualmente, CT é, portanto, uma disciplina, interdisciplinar a Aprendizagem de Máquina e Recuperação de Informação, e, como tal, compartilha uma série de características com Mineração de Textos (Cavaleiro 1999; Pazienza 1997).

Ainda há um debate considerável sobre onde a fronteira exata entre essas disciplinas se encontra e a terminologia ainda está em processo de evolução. Mineração de Textos é cada vez mais utilizada para designar todas as tarefas que, através da análise de grandes quantidades de texto detecta padrões de uso e, extrai informações úteis. De acordo com este ponto de vista, CT é uma instância de Mineração de Textos.

Uma observação deve ser realizada sobre CT: muitas vezes, o termo **Categorização Automática de Textos**[22] (CAT) é utilizado para referenciar a abordagem de CT realizada por Aprendizagem de Máquina, mesmo que todas as etapas necessárias para a realização deste processo, como por exemplo, o ajuste dos parâmetros dos classificadores utilizados seja manual. Além disso, segundo (MANNING, RAGHAVAN, & SCHÜTZE, 2007), o termo também é utilizado para designar a descoberta de um conjunto de grupos que compartilham características comuns, como em (BORKO & BERNICK, 1963); uma tarefa normalmente chamada de Clusterização de Documentos.

5.3. Definição

De acordo com (LINDEN, 2008), Categorização de Textos é a atribuição de um valor booleano para cada par $(d_i, c_j) \rightarrow D \times C$, em que D é uma coleção de documentos e $C = \{c_j, \ldots, c_{|c|}\}$ um conjunto de categorias predefinidas. Partindo desse conceito inicial, a CT é formalmente descrita como a decisão de classificar $\phi d_i \notin c_j$, o valor-verdade $\mathbb{F}$ é associado ao par (d_i, c_j), caso contrário aplica-se o valor-verdade $\mathbb{V}$.

[22] Do termo inglês, Automated Text Categorization.

Para que o processo de decisão seja automatizado, é necessário que uma inferência seja realizada. Mais formalmente, a tarefa é aproximar a função de destino desconhecida $\breve{\phi} : D \times C \rightarrow \{\mathbb{V}, \mathbb{F}\}$ por meio de uma função $\phi : D \times C \rightarrow \{\mathbb{V}, \mathbb{F}\}$ denominada classificador, tal que $\breve{\phi}$ e ϕ coincidam tanto quanto possível.

Na realidade, a função $\breve{\phi}$ é uma aproximação da função ϕ. Essa função descreve o modo como os documentos da coleção D devem ser classificados em C. Os dados de entrada dessa função são os documentos ou suas características representativas e suas respectivas categorias. A saída é um conjunto de valores $\{\mathbb{V}, \mathbb{F}\}$.

(SEBASTIANI, 2002) também ressalta algumas premissas sobre o processo de Aprendizagem de Máquina em CT:

- As categorias são apenas rótulos simbólicos, e nenhum conhecimento adicional de seu significado está disponível.
- Nenhum tipo de conhecimento exógeno, ou seja, informações de qualquer tipo fornecidas para efeitos de classificação por uma fonte externa está disponível. A classificação deve ser baseada apenas no conhecimento endógeno, ou seja, o conhecimento extraído dos documentos. Metadados, tais como data de publicação, tipo de documento, fonte de publicação, etc., não estão disponíveis.

5.4. Tipos de Classificadores

Diferentes restrições podem ser aplicadas à tarefa de CT, dependendo do domínio do problema. Pode ser necessário que para um determinado número inteiro k, exatamente k (ou $\leq k$, ou $\geq k$) elementos de C sejam atribuídos a cada $d_j \in D$. O caso em que exatamente uma categoria deve ser atribuída a cada $d_j \in D$ é denominado monocategórico. O caso em que qualquer número de categorias de 0 à $|C|$ pode ser atribuído ao mesmo $d_j \in D$ é denominado multicategórico. Um caso especial do tipo monocategórico é o binário, em que cada $d_j \in D$ deve ser atribuído tanto à categoria c_i ou ao seu complemento $\overline{c_i}$.

Do um ponto de vista teórico, o caso binário é mais geral do que o multicategórico, uma vez que um algoritmo de classificação binária também pode ser usado para a classificação multicategórica: para isso, é necessário tratar o problema da classificação multicategórica de $\{c_1, \ldots, c_{|c|}\}$ em $|C|$ problemas independentes de classificação binária de $\{c_i, \overline{c_i}\}$, para $i = 1, \ldots, |C|$.

No entanto, essa abordagem requer que as categorias sejam estocasticamente independentes umas das outras, isto é, para qualquer c', c'', o valor de $\breve{\phi}(d_j, c')$ não depende do valor de $\breve{\phi}(d_j, c'')$, e vice versa. O inverso não é verdadeiro: um algoritmo de classificação multicategórico não pode ser usado para o caso binário. Na realidade, dado um documento d_j para ser classificado, o classificador multicategórico pode atribuir $k > 1$ categorias a d_j, e talvez não seja óbvio como escolher uma categoria mais adequada, ou o classificador multicategórico pode não atribuir a d_j categoria alguma, e talvez não seja óbvio como escolher a categoria menos inadequada de C.

5.5. Modelagem da categorização

Existem duas maneiras diferentes de se modelar um categorizador de textos. A primeira é denominada **categorização orientada a documento**[23]: dado $d_j \in D$, deseja-se encontrar todas as categorias $c_i \in C$ pertinentes a d_j. A alternativa é designada **categorização orientada a categoria**[24]: dada uma categoria $c_i \in C$, deseja-se encontrar todos os documentos d_j pertencentes a c_i.

Mais pragmática do que conceitual, esta distinção é importante quando os conjuntos C e D não estão inteiramente disponíveis desde o início. Também é relevante para a escolha do método de construção do classificador, pois alguns métodos permitem a construção de classificadores com uma inclinação definida para um ou outro modelo.

[23] Do termo inglês, *document-pivoted categorization.*

[24] Do termo inglês, *category-pivoted categorization.*

A categorização orientada a documento é, portanto, apropriada quando os documentos se tornam disponíveis em diferentes momentos no tempo, por exemplo, na filtragem de e-mail. Já a categorização orientada a categoria é mais adequada quando uma nova categoria $c_{|c|+1}$ pode ser adicionada a um conjunto existente $C = \{c_1, \ldots, c_{|c|}\}$ depois que uma série de documentos já tenham sido classificados em C, e esses documentos precisam ser reconsiderados para a classificação em $c_{|c|+1}$. A categorização orientada a documento é utilizada mais frequentemente do que PCC, pois a primeira situação é mais comum do que a última.

5.6. Tipos de categorização

Enquanto a automação completa da tarefa CT exige uma decisão $\mathbb{V}\, ou\, \mathbb{F}$ para cada par $(d_j,\ c_i)$, a automação parcial deste processo pode ter necessidades diferentes.

Por exemplo, dado $d_j \in D$ um sistema pode simplesmente classificar as categorias $C = \{c_1, \ldots, c_{|c|}\}$ de acordo com o grau de pertinência estimada à d_j, sem tomar qualquer decisão definitiva para qualquer uma das categorias. Tal lista ordenada seria de grande ajuda para um especialista humano encarregado de tomar a decisão final da categorização, uma vez que ele poderia, assim, restringir sua escolha às categorias do topo da lista, sem ter que examinar todo o conjunto.

Alternativamente, dado $c_i \in C$ um sistema pode simplesmente classificar os documentos em D de acordo o grau de pertinência estimada à c_i; simetricamente, para a classificação em c_i um especialista humano iria apenas examinar os documentos do topo da lista em vez de todo o conjunto de documentos.

Essas duas modalidades são denominadas categorização de textos ranqueada por categoria e categorização de textos ranqueada por documento (Yang, 1999), respectivamente, e são as contrapartidas evidentes de categorização orientada a documento e categorização orientada a categoria.

5.7.
Aplicações de Categorização de Textos

5.7.1.
Organização de documentos

Indexação com um vocabulário controlado é uma instância do problema geral da organização de base do documento. Em geral, muitas outras questões relativas à organização e armazenamento de documentos, seja para fins de organização pessoal ou estruturação de uma base corporativa de documentos, podem ser abordadas por técnicas de CT.

Por exemplo, nos escritórios de um jornal de classificados, os anúncios devem ser, antes de sua publicação, categorizados em categorias, tais como encontros, carros para venda, imóveis e outras. Jornais que lidam com um grande volume de anúncios classificados se beneficiariam de um sistema automático que define a categoria mais adequada para um determinado anúncio. Outras aplicações possíveis são a organização de patentes em categorias para tornar a sua busca mais fácil ou a identificação automática de artigos de jornal sob as seções apropriadas (Política, Esportes, Estilo de Vida, etc.).

5.7.2.
Filtragem de Documentos

Filtragem de documentos é a atividade de categorizar um fluxo de chegada de documentos enviados de forma assíncrona de um produtor de informação para um consumidor de informação (Belkin e Croft, 1992). Um caso típico é um *feed* de notícias, em que o produtor é uma agência de notícias e o consumidor é um jornal (Hayes et al., 1990). Neste caso, o sistema de filtragem deve bloquear o fornecimento de documentos em que provavelmente o consumidor não está interessado.

Pode ser vista como um caso de CT binária, isto é, a classificação dos documentos que entram em duas categorias disjuntas: relevantes e irrelevantes. Além disso, um sistema de filtragem também podem ainda classificar os

documentos considerados relevantes para o consumidor de informação em categorias temáticas. No exemplo acima, todos os artigos sobre esportes devem ser classificados de acordo com o esporte que lidam, de modo a permitir que os jornalistas especializados em esportes individuais acessem apenas os documentos de interesse potencial para eles.

5.7.3. Desambiguação Lexical de Sentido

A ambiguidade lexical é causada, fundamentalmente, pela existência de algumas relações semânticas interlexicais, principalmente a polissemia e a homonímia. De acordo com (Lyons, 1977), na polissemia uma mesma palavra tem dois ou mais significados diferentes, mas relacionados entre si, sendo que, normalmente, somente um dos significados se ajusta a um determinado contexto. Na homonímia duas ou mais palavras com significados totalmente distintos, sem traços comuns, são idênticas quanto ao som (homofonia) e/ou à grafia (homografia).

O problema da ambiguidade lexical pode, ainda, ser classificado como ambiguidade categorial ou ambiguidade de sentido (Ullmann, 1964).

A ambiguidade categorial ocorre quando as duas ou mais opções de significados de uma dada palavra são de diferentes categorias gramaticais. Na tradução, um exemplo de ambiguidade categorial causada pela relação de homonímia é a palavra do inglês *field*, que pode ser traduzida para as palavras "campo" (substantivo) ou "interceptar" (verbo), no português. Já um exemplo de ambiguidade categorial derivada da relação de polissemia é a palavra do inglês *eats*, que pode ser traduzida no português como "mantimentos, víveres, gêneros alimentícios" (substantivos) ou "come" (verbo "comer" conjugado na terceira pessoa singular, presente do indicativo).

A ambiguidade de sentido, por sua vez, ocorre quando as duas ou mais opções de sentido (ou tradução) de uma dada palavra têm a mesma categoria gramatical. Alguns exemplos são a palavra *know*, que pode ser traduzida como "saber" ou "conhecer", como um caso de polissemia, e a palavra light, que pode ser traduzida como "leve" ou "luz", como um caso de homonímia.

A ambiguidade categorial é, em geral, muito mais simples que a de sentido, uma vez que pode ser resolvida, na maioria das vezes, pela análise das características sintáticas das palavras, realizada por procedimentos de etiquetação gramatical (ver item 3.2.3.2) ou análise sintática (ver item 3.2.3.6), por exemplo. Procedimentos dessa natureza alcançam, atualmente, resultados bastante satisfatórios. A **resolução da ambiguidade de sentido**[25], por sua vez, exige a análise da semântica das palavras e, eventualmente, a análise do uso de tais palavras (realizadas por procedimentos de análise semântica e pragmática, por exemplo).

WSD é muito importante para muitas aplicações, como Processamento de Linguagem Natural e indexação de documentos pelo valor semântico dos termos. Pode ser vista como uma tarefa de CT, uma vez que contextos de ocorrência do termo sejam tratados como documentos e os sentidos do termo como categorias. Em geral, é abordado como um problema de categorização binária e modelo como categorização orientada a documento.

5.8. Aprendizagem de Máquina em CT

5.8.1. Aprendizagem Supervisionada

Nesta abordagem de Aprendizagem de Máquina, um processo indutivo cria automaticamente um classificador para uma categoria c_i observando as características representativas de um conjunto de documentos categorizados manualmente sob c_i ou $\bar{c}_i$ por um especialista no domínio. A partir destas características, o processo indutivo compila as características que um novo documento deve ter para ser classificado sob c_i.

Na terminologia de Aprendizagem de Máquina, o problema de classificação é uma atividade de aprendizagem supervisionada, uma vez que esse processo de

[25] Do ter inglês, word sense disambiguation (WSD)

aprendizagem é guiado pelo conhecimento das categorias e dos exemplos de treinamento que lhes são pertinentes.

5.8.2. Treinamento e Teste

Existem diversas estratégias para a execução do treinamento e, posteriormente, aplicação de testes. O treinamento tem como objetivo apresentar ao classificador exemplos que o farão conhecer e aprender sobre a massa textual. A aplicação de testes possibilita a avaliação do desempenho. A seguir, a descrição das principais estratégias relatadas na literatura:

- **Holdout**: Consiste em separar do conjunto de treinamento uma determinada porção, compondo o conjunto de teste. Usualmente, o teste utiliza 1/3 do conjunto total, mantendo o restante para treinamento. Apesar de simples e rápida de se aplicar, recebe críticas por não usar de forma otimizada o conjunto total de amostras – o classificador poderia ficar mais bem construído se utilizado todo o conjunto de treinamento – e pela própria aleatoriedade dos dados, isto é, o conjunto de teste pode acabar ficando "favorecido", levando a uma falsa conclusão da real adequação do treinamento.
- **K-Fold Cross Validation** (Validação Cruzada): Validação Cruzada é a metodologia de treinamento e teste que trabalha com o conceito de *folds*. Desta forma, o conjunto de amostras inicial é dividido em *k* subamostras. Destas *k* subamostras, uma subamostra é retida para ser utilizada na validação do modelo (conjunto de teste) e as *k-1* subamostras compõem o conjunto de treinamento. O processo é então repetido *k* vezes, de modo que cada uma das *k* subamostras seja utilizado ao menos uma vez como teste. O resultado final é a média do desempenho do classificador nas *k* iterações. O objetivo desta estratégia é aumentar a confiabilidade da avaliação, com o ônus de se despender mais tempo que a técnica anterior. Vale ressaltar que nada impede que as duas estratégias possam ser combinadas, com a aplicação da técnica de *holdout* como mais uma

forma de validar os resultados conseguidos com a Validação Cruzada, com o ônus de se despender muito mais tempo para a execução dos ciclos e de ser necessário mais dados para que os conjuntos (treinamento e teste) formados possam ter tamanho significante.

5.8.3. *k*-Nearest Neighbors

O *k-Nearest Neighbors* (KNN) é um algoritmo de aprendizagem supervisionado, pertencente a um grupo de técnicas denominado de *Instance-based Learning* ou *Lazy Learning* que tem sido usado em muitas aplicações no campo da Mineração de Dados, reconhecimento de padrões estatísticos, processamento de imagens e entre outros. É considerado um dos melhores métodos para a classificação de texto, é simples, efetivo e escalonável para grandes aplicações. Algumas aplicações de sucesso incluem reconhecimento de escrita à mão e imagens de satélite.

O algoritmo *k-Nearest Neighbors* classifica um dado elemento desconhecido (de teste) baseado nas categorias dos *k* elementos vizinhos mais próximos, ou seja, os elementos do corpus de treinamento que obtiveram os graus de similaridade mais altos com o elemento de teste, conforme ilustrado na Figura 24.

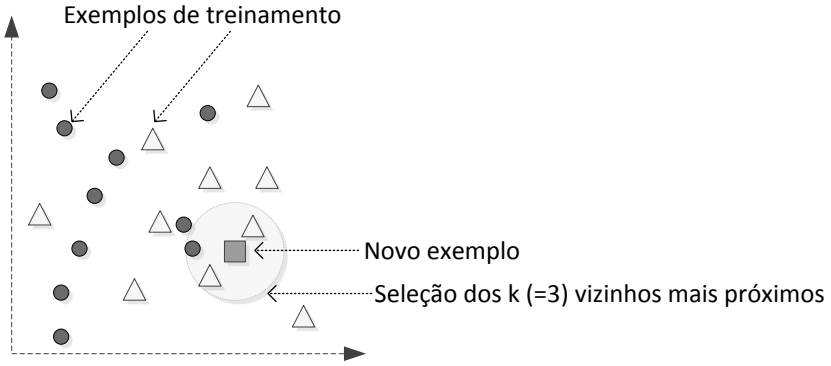


Figura 24 – Algoritmo KNN - Seleção baseada nos *k* (= *3)* vizinhos

Calcula-se a similaridade de cada um dos elementos do corpus de treino com o elemento que se quer classificar e então ordena os componentes da base de treino do mais similar ao menos. Dos elementos ordenados, selecionam-se os *k* primeiros, que irão servir como parâmetro para a regra de classificação.

Dois pontos importantes do kNN são: a regra de classificação e a função que calcula a similaridade. A regra de classificação diz como o algoritmo vai tratar a importância de cada uns dos *k* elementos mais próximos. A função de similaridade vai mensurar quão dois elementos são semelhantes de forma a poder identificar quais são os kNN.

Para $k = 1$ não existe regra de classificação, pois o elemento de classe desconhecida terá a mesma classificação do vizinho mais próximo. Para $k > 1$ é preciso uma regra para decidir a qual classe se atribuirá o elemento. Em (WANG, 2006) são revistas duas regras de classificação bem comuns: maioria na votação (*majority voting scheme*) e a soma do peso da similaridade (*weighted-sum voting scheme*). Na primeira, cada elemento tem uma influência igual, a classe escolhida será aquela que tiver mais representantes entre os *k* elementos. Na segunda, entre os *k* elementos, são somadas as similaridades dos elementos de mesma categoria, o elemento desconhecido será classificado na categoria que obtiver maior valor. As funções do cálculo de similaridade mais conhecidas na literatura são: a do cálculo do cosseno de dois vetores e a distância euclidiana.

5.8.4. SVM

Support Vector Machines ou Máquinas de Vetores de Suporte constituem uma técnica de Aprendizagem de Máquina supervisionada, utilizada para classificação e regressão. Em problemas de classificação, a ideia principal de uma máquina de vetores de suporte é construir um hiperplano como superfície de decisão de tal forma que a margem de separação entre exemplos positivos e negativos seja máxima, conforme ilustrado na Figura 25.

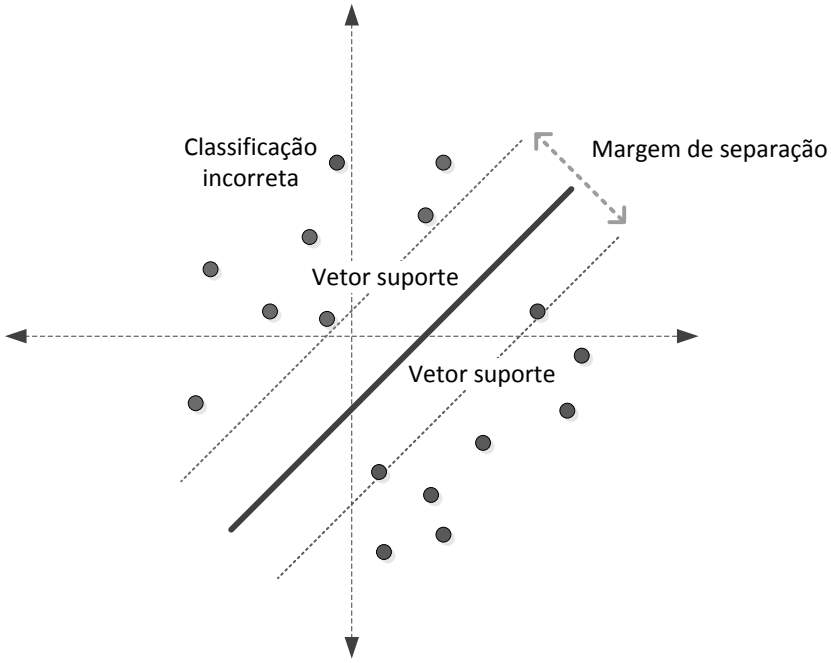


Figura 25 – Máquina de Vetores de Suporte

Aplicada com sucesso em diversas áreas, SVM tem sido muito utilizada em PLN, e com excelentes resultados em tarefas como Categorização de Textos (JOACHIMS, 1998) e Desambiguação Lexical de Sentido (LEE, NG, & CHIA, 2004).

SVM é um classificador binário. Para estender a utilização de SVMs em problemas de *k* classes (*k > 2*), há uma abordagem básica, conforme ilustrado na Figura 26, que é reduzir o problema de *k* classes a um conjunto de problemas binários. Para isto, duas abordagens são possíveis, e seguem os seguintes passos:

- Decomposição um por classe:
  1. Construção de *k* SVMs binárias para separar uma classe de todas as outras.
  2. Classificação nas *k* classes: o ponto x é classe da SVM com maior saída.
- Separação das classes duas a duas:
  1. Construção da SVM_{ij} para distinguir a classe *i* da classe *j*, $i \neq j$ e $i, j \in \{1,...,k\}$.
  2. Classificação na classe *i* ou *j* (voto para essa classe).
  3. Classificação final através de votação.

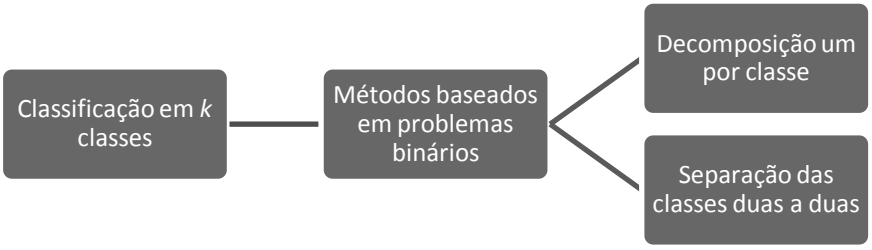


Figura 26 – Abordagens SVM para problemas não binários

Para ambas as abordagens vistas acima, podem ocorrer situações em que o resultado final não pode ser concluído sobre a classe a ser predita. Por exemplo, ao reduzir um problema de classificação com três classes distintas, é necessária a construção de três SVMs, sejam estas para a decomposição um por classe (SVM_1, SVM_2, SVM_3) ou separação das classes duas a duas (SVM_{12}, SVM_{13}, SVM_{23}). E nestes casos, pode ocorrer empate no resultado final, conforme ilustrado Tabela 9:

Tabela 9 - Resultados conflitantes de SVMs binárias

Um por classe	Saída SVM
SVM_1	Sim
SVM_2	Não
SVM_3	Sim

Duas a duas	Saída SVM
SVM_{12}	1 (sim)
SVM_{13}	3 (não)
SVM_{23}	2 (sim)

Outro detalhe importante sobre a utilização de SVMs em problemas de classificação é relativo ao valor do parâmetro *C*. que possui as seguintes características:

- Controla o compromisso entre a complexidade da maquina e o número de pontos não separáveis;
- Pode ser visto como uma forma de parâmetro de regularização e deve ser selecionado pelo usuário.

Em geral, são utilizadas heurísticas gulosas para o ajuste deste parâmetro, ou seja, inicia-se com valores pequenos e gradativamente aumenta-se o valor de *C*.

Importante também é definir o tipo de função *kernel* a ser utilizada pela SVM. A saber, são três funções disponíveis: Linear; Polinomial e RBF.

5.8.5. Combinação de Classificadores

A combinação de classificadores, uma alternativa para melhorar o processo de CT, é baseada na seguinte premissa: dada uma tarefa que exige o conhecimento de especialista, uma quantidade n de especialistas irá apresentar um resultado de melhor qualidade do que apenas um especialista apresentaria. Na Categorização Automática de Textos, esse esforço consiste em executar um conjunto de *k* classificadores $\{\phi_1, \ldots, \phi_k\}$ para a mesma tarefa de categorização e combiná-los de forma apropriada. Essa combinação de classificadores é caracterizada por uma escolha de *k* classificadores e de uma função combinatória.

A *performance* de um classificador pode ser ajustada para obter a melhor acurácia possível para uma determinada situação, mas os ajustes são uma tarefa complexa e, ainda, podem existir padrões em que mesmo o melhor classificador apresente falhas na categorização. Segundo (ALPAYDIN, 2004), para obter resultados produtivos na combinação de classificadores, as decisões de categorização tomadas pelos classificadores não podem ser as mesmas. Decisões iguais levam a resultados iguais; se for tomada a decisão errada, todos os classificadores erram. Decisões diferentes permitem a um classificador errar enquanto outros classificadores acertam, resultando em uma decisão final correta.

A principal discussão desse método é em relação à função combinatória. Dentre inúmeras funções combinatórias disponíveis, a votação, a seleção dinâmica de classificadores e a combinação adaptativa de classificadores são três funções combinatórias comumente utilizadas. Segue uma breve exposição sobre cada uma, a partir dos pressupostos apresentados por (SEBASTIANI, 2002) e (BENNETT, DUMAIS, & HORVITZ, 2005):

- A função combinatória mais simples é a escolha por voto majoritário. Nela a decisão final é obtida escolhendo-se a categoria com maior número de votos dentre os *k*

classificadores. Outra forma de aplicar a votação é atribuir pesos aos classificadores (votação com pesos); nesse caso o voto de cada classificador é relativo ao seu peso, para a decisão final.

- Um exemplo de seleção dinâmica de classificadores é a utilização do classificador mais eficiente para uma determinada categoria ou categorias. Esta função também pode ser denominada *Best By Class*.
- A combinação adaptativa de classificadores é descrita como uma função intermediária entre a votação com pesos e a seleção dinâmica de classificadores. Essa função agrega todos os votos dos classificadores, atribuindo pesos para a contribuição da decisão final conforme a eficiência de cada classificador em um corpus de avaliação.

Com base nessas funções combinatórias iniciais, diferentes autores descrevem suas próprias propostas de métodos combinatórios que, de uma forma ou outra, fazem uso dos conceitos expostos por (SEBASTIANI, 2002). A seguir são apresentados alguns exemplos de combinação de classificadores:

- O uso de *bagging* em (ALPAYDIN, 2004) descreve um método de votação onde os classificadores são treinados com pequenas alterações no corpus. Nesse caso, os classificadores devem ser treinados usando uma amostra aleatória do corpus, diferente a cada iteração. Para garantir tamanhos iguais de amostra para todos os classificadores, os documentos usados em uma iteração são repostos nas próximas iterações. Existe a possibilidade dos documentos serem usados mais de uma vez ou, até mesmo, nenhuma vez. O fator aleatório permite amostras, ao mesmo tempo, semelhantes e diferentes. Semelhantes porque podem compartilhar os mesmos documentos, e diferentes porque documentos diferentes são incorporados à

amostra. A principal desvantagem desse método é o fato de que o desempenho dos classificadores é baseado na probabilidade da escolha das amostras.

- O método de *boosting* (SEBASTIANI, 2002) apresenta um conceito intuitivo de aprimoramento, que se encaixa na definição de seleção dinâmica de classificadores. A ideia do *boosting* é aplicar um conjunto de *n* classificadores iterativamente sobre um corpus de treino. A cada iteração um novo classificador prioriza a categorização nos documentos onde o classificador anterior obteve a maior taxa de categorizações incorretas. Assim, o treinamento de novos classificadores não é baseado em probabilidade, como ocorre no método *bagging*. A desvantagem desse método é a exigência de um corpus suficientemente grande para o treinamento dos classificadores.
- O algoritmo *AdaBoost* (FREUND & SCHAPIRE, 1999), frequentemente citado na literatura, utiliza uma medida de peso para referenciar os documentos incorretamente categorizados, que recebem pesos maiores, enquanto documentos corretamente classificados recebem pesos menores. Utilizando a reposição de documentos, o algoritmo permite a escolha por documentos com pesos menores em corpora pequenos.
- (ALPAYDIN, 2004) exemplifica uma forma de combinação em cascata entre os classificadores. A ideia é ter *k* classificadores utilizados em sequencia, de acordo com sua complexidade ou custo de representação. Assim, os classificadores são aplicados a partir do mais simples ao mais complexo (FREUND & SCHAPIRE, 1999). Cada classificador garante um grau de confiabilidade em

sua decisão, para que os classificadores seguintes concentrem o esforço em categorizar os documentos com baixo índice de confiabilidade na categorização. Esse é um método muito semelhante ao *boosting*, a nova característica é o uso de classificadores diferentes no processo de categorização.

5.9. Ferramentas de Mineração de Textos

A seguir são descritas algumas destas ferramentas com suas características e aplicabilidade.

5.9.1. Weka

O software WEKA, desenvolvido por (HALL, FRANK, HOLMES, FAHRINGER, REUTEMANN, & WITTEN, 2009), é uma coleção de algoritmos de aprendizado de máquina para tarefas de Mineração de Dados. Os algoritmos podem ser aplicados diretamente a um conjunto de dados por meio de sua interface gráfica ou executados em outros aplicativos customizados via código Java. Contém ferramentas para pré-processamento de dados, classificação, regressão, clusterização, regras de associação e visualização. Também é bem adequado para o desenvolvimento de novos sistemas de aprendizagem de máquina.

Embora desenvolvido para lidar com informações estruturadas, o software dispõe de um filtro chamado *StringToWordVector* que realiza o processo de tokenização e remoção de *stopwords* de documentos textuais. Além disso, possui a habilidade de realizar o cálculo de relevância dos termos tokenizados segundo métricas como IDF e TF/IDF. Possui a opção de realizar a operação e *Case*

Folding que consiste em converter todos os termos para caixa baixa. Uma frequência mínima por termo pode ser definida fazendo com que este seja descartado pelo processo de tokenização. A figura exibe a tela de configuração desse filtro.

Após a aplicação desse filtro, obtém-se uma representação estruturada dos documentos, tornando possível a aplicação dos algoritmos tradicionais de Mineração de Dados disponíveis na ferramenta.

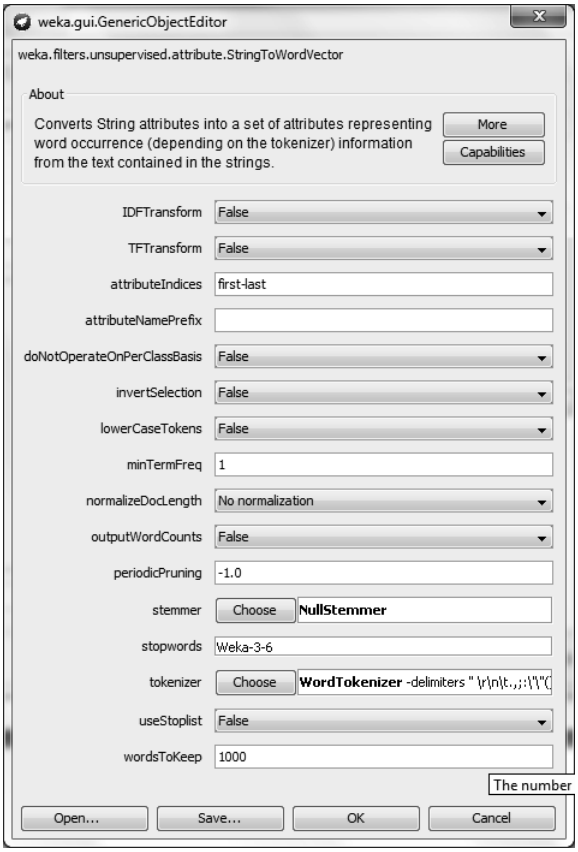


Figura 27 – Parâmetros de configuração do filtro *StringToWordVector* do software Weka

5.9.2. Text Mine

Text Mine, desenvolvido por (KONCHADY, 2006), é um conjunto de ferramentas de Mineração de Textos escritas em Perl. Necessita de um servidor *web* Apache e banco de dados MySQL.

Text Mine disponibiliza as seguintes funcionalidades:

- Coleta: a busca pode ser feita a partir de URLs, onde o *crawler* visita as páginas, construindo o conjunto de informações necessárias para as outras funcionalidades, ou em documentos locais à máquina.
- Extração de informação: é capaz de identificar entidades no texto como nomes, locais e organizações. Para isso, o usuário deve customizar um dicionário de entidades que serão identificadas.
- Clusterização: retorna uma coleção de grupos, onde cada grupo é composto por um número variável de documentos similares. Gera uma matriz de similaridades da coleção de documentos e utiliza algoritmo genético para agrupá-los segundo o grau de similaridade existente entre eles.
- Categorização: consiste em organizar as mensagens de texto em categorias.
- Sumarização: consiste em resumir artigos ou documentos da Web, extraindo palavras- chave que determinam o significado do documento.

5.9.3. TMSK

TMSK ou Text-Miner Software Kit é um pacote de softwares para mineração preditiva de textos. Possui funcionalidades para pré-processar documentos de textos em formato XML e disponibiliza implementações para as seguintes tarefas:

- Pré-processamento: implementa funcionalidades de tokenização, *stemming*, criação de dicionário e detecção de início e fim de sentenças.
- Classificação: possui classificadores baseados em regras de decisão, predição usando Naive Bayes e modelos de ranqueamento linear.
- Recuperação de informação: implementa o conceito de índices invertidos para busca de informação.
- Clusterização: realiza o agrupamento de documentos usando o algoritmo *k-means*.

- Extração de informações: identificação de entidades nomeadas.

5.9.4. RIKTEXT

O RIKTEXT é um pacote de softwares completo para categorização de documentos baseado em regras de decisão.

Seu objetivo é determinar o melhor conjunto de regras para a predição e classificação, onde o melhor conjunto é formado pelo menor número de regras com erro mínimo.

Os dados para este classificador devem estar na forma de tabela, onde cada linha corresponde a um documento e cada coluna corresponde a um termo do dicionário. Cada célula da planilha recebe valores booleanos indicando a presença ou a ausência da palavra no documento, ou a frequência linear do termo.

O RIKTEXT complementa o TMSK, disponibilizando métodos para construção e uso de regras para classificação de documentos. O formato dos dados de entrada do RIKTEXT é idêntico ao dos métodos de classificação apresentados no TMSK.

5.9.5. STATISCA Text Miner

O STATISTICA Text Miner é uma extensão opcional do STATISTICA Data Miner que compreende as etapas de coleta, pré-processamento e mineração do processo de *KDT*.

Dispõe de algoritmos para a redução de palavras ao seu próprio radical (lematização) e eliminação de *stopwords*. Assim, o aplicativo elimina uma parte insignificante do texto a ser analisado, reduzindo o número de termos da matriz de palavras de texto. Uma vez definidas as palavras com real valor para a análise, são aplicados algoritmos de mineração, por meio dos quais se derivam modelos preditivos para explicar a possibilidade de ocorrência de termos significantes em outros documentos.

As características principais do STATISTICA Text Miner:

- Contém várias opções de acesso a documentos textuais em diversos diferentes, incluindo TXT, PDF, PostScript, HTML, XML, RTF e DOC.
- Provê suporte à web-*crawling*, permitindo extrair documentos de páginas Web. Também é possível selecionar os documentos a partir de uma pasta local, conforme ilustrado na Figura 28.
- Inclui *stoplists* e algoritmos de *stemming* para as línguas: dinamarquês, holandês, inglês, francês, alemão, italiano, português, espanhol, sueco etc. As *stoplists* podem ser editadas manualmente pelo usuário. O software é projetado de modo que o suporte a línguas adicionais possa ser feito com o mínimo de esforço;
- Efetua a contagem de frequência de todas as palavras nos documentos, que serve de base para todas as análises numéricas subsequentes. Filtros adicionais podem ser aplicados. Por exemplo, cálculo de frequência normalizada dos termos, frequência inversa dos documentos, gerando uma sumarização numérica dos documentos;
- Disponibiliza métodos de análise estatística que são aplicadas sobre os sumários numéri¬cos dos documentos, técnicas de Clusterização, tais como *k-Means* para identificar documentos similares relacionamentos entre estes.

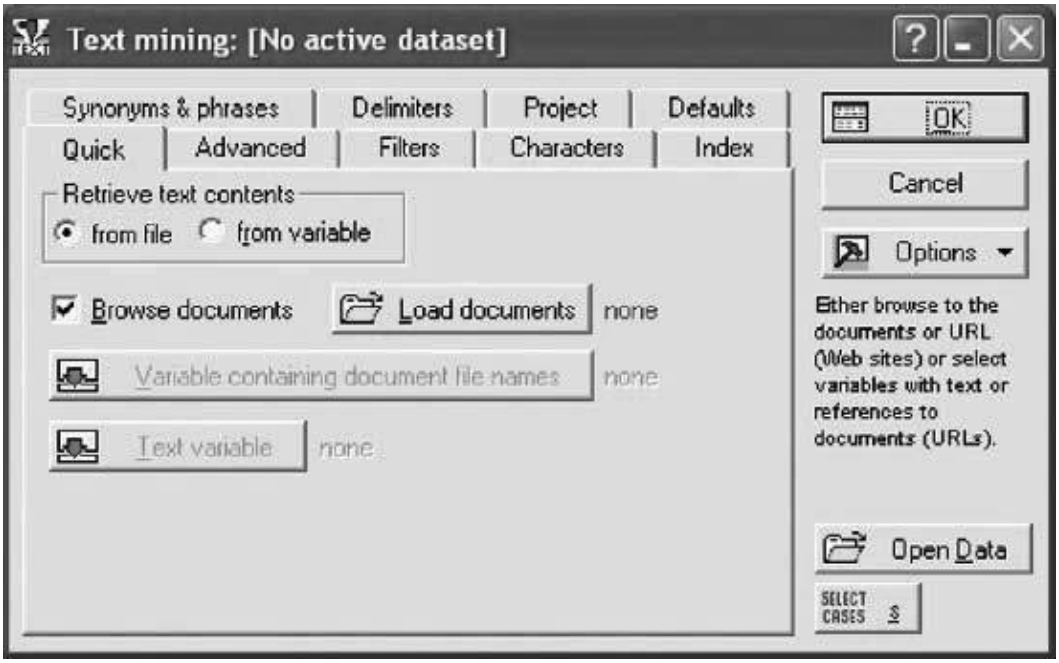


Figura 28 – Interface de coleta do software STATISCA

6
Framework proposto

6.1.
Definição

Um *framework* é um conjunto de componentes (classes e interfaces) que funcionam juntos para solucionar um determinado problema de software. Segundo (JOHNSON & FOOTE, 1988), um *framework* é uma aplicação reutilizável e semicompleta que pode ser especializada para produzir aplicações personalizadas.

A principal finalidade de um framework é proporcionar a sua reutilização em outros projetos, diminuindo a complexidade na implementação de softwares. Além disso, um *framework* determina a arquitetura de sua aplicação, ou seja, define a estrutura geral, sua divisão em classes e objetos e, consequentemente, as responsabilidades entre si, assim como a forma de colaborarem e o fluxo de controle.

Um *framework* possui as seguintes características:

- É composto por múltiplas classes ou componentes, cada um provendo uma abstração de um conceito particular;
- Define como as abstrações trabalham juntas para resolver um problema;
- Seus componentes são reutilizáveis;
- Organiza padrões em alto nível.

O *framework* desenvolvido nesta Tese para a Categorização de Textos possui aproximadamente cento e vinte classes e interfaces e quatro mil linhas de código.

6.2. Ambiente de desenvolvimento

O ambiente de software neste trabalho foi implementado na linguagem de programação **C#**[26]**.** C# é uma linguagem de programação orientada a objetos criada pela empresa Microsoft e faz parte da plataforma de desenvolvimento ".NET". Embora existam mais de vinte linguagens de programação suportadas pela tecnologia ".NET", a linguagem C#, baseada na linguagem C++ e Java, é considerada a linguagem símbolo da plataforma ".NET".

Com uma ideia semelhante a da plataforma Java, o ambiente de desenvolvimento de aplicações ".NET" permite que *softwares* sejam desenvolvidos para qualquer sistema operacional que suporte o .NET *Framework*.

Portanto, constituem justificativas para a escolha deste Ambiente de Desenvolvimento e Linguagem de Programação a orientação a objetos que permite fácil reuso, bem como grande organização do código e possibilidade de portabilidade para diversos sistemas operacionais.

6.3. Objetivos

O *framework* proposto é capaz de:

- Empregar técnicas de Mineração de Textos para o fornecimento de ferramentas que auxiliem a execução da tarefa de Categorização de Textos.
- Fornecer tratamento linguístico específico à Língua Portuguesa do Brasil, isto é, que faça proveito das ricas informações semânticas presentes em qualquer linguagem natural;
- Automatizar as etapas necessárias para a realização de Categorização de Textos, otimizando a combinação das técnicas de pré-processamento textual e a escolha dos algoritmos de Aprendizado de Máquina e seus respectivos parâmetros.

[26] Pronuncia-se C Sharp.

6.4. Coleta

Nenhum mecanismo de coleta de dados foi implementado, pois a coleção de documentos utilizada nos estudos de caso foi obtida por meio do *corpus* CETENFolha (item 6.6).

6.5. Pré-Processamento

6.5.1. Tokenização

O processo de Tokenização (seção 3.2.1) empregado no *framework* segue a metodologia de geração de *tokens* (ver Figura 12) proposta em (KONCHADY, 2006). Esta metodologia, fortemente apoiada na utilização de um dicionário de palavras, obtém *tokens* com alto valor semântico, conforme resultados obtidos nos experimentos realizados no Estudo de Caso. Esta metodologia é composta dos seguintes passos:

1. Geração simples de *tokens:* baseada no conjunto de *tokens* delimitadores, como espaço e fim de linha.
2. Identificação de abreviações: realizada com auxílio de dicionários desenvolvidos para este propósito. Todas as abreviações encontradas são substituídas pelos seus termos não contraídos.
3. Identificação de palavras combinadas: muito comum em nomes próprios de organizações. Em geral, são palavras separadas por símbolos como o "&": "Casa & Vídeo".
4. Identificação de símbolos de Internet: geralmente, símbolos de internet atendem às normas regras estabelecidas no momento de criação do serviço. Desta forma, o uso de expressões regulares

auxilia na identificação de tais símbolos (JARGAS, 2006). Alguns exemplos de símbolos de Internet são IP, *e-mails* e *URLs*.

5. Identificação de números: este processo é realizado pela verificação de conteúdo numérico em meio aos *tokens*. Números na forma extensa, quando identificados, são convertidos em formato numérico.
6. Identificação de *tokens* multivocabulares: realizado também com forte apoio de um dicionário de termos. Busca reunir em um único *token* palavras que, quando utilizadas em conjunto, transmitem ideias diferentes ou incompletas quando utilizadas separadas. Exemplos comuns são "bolsa de valores", "casa da moeda".

Além disso, para automatizar o processo de Identificação de *tokens* multivocabulares é realizado o cálculo do coeficiente de correlação entre os *tokens.* Essa medida indica a força e a direção do relacionamento linear entre duas variáveis aleatórias. Desta forma, *tokens* que apresentam alto grau de correlação são candidatos a *tokens* multivocabulares.

Bibliotecas que lidam com o processamento de linguagem natural contam com algoritmos de tokenização, como é o caso do Lucene e do OpenNLP com o *WhiteSpaceTokenizer* que divide o texto em *tokens* de acordo com os espaços em branco, ou seja, sequência de caracteres adjacentes entre os espaços em branco formam *tokens* (The Apache Software Foundation, 2011). Essa abordagem é simplista demais.

6.5.2. Análise/Remoção de *stopwords*

O processo de remoção de *stopwords* (item 3.2.2) possui o objetivo de reduzir a grande dimensionalidade de dados textuais através da remoção de palavras de baixo poder discriminatório.

O *corpus* utilizado nos estudos de casos é o CETENFolha (item 6.6). Essa coleção dispõe de três listas padrões de *stopwords*, elaboradas pela entidade

organizadora do *corpus*. Cada uma das listas possui cem, duzentos e trezentos termos considerados irrelevantes. A utilização de uma lista ou de outra deve ser baseada nos resultados obtidos.

Uma lista de aproximadamente cem *stopwords* da Língua Portuguesa é exibida na Tabela 10 e pode ser encontrada no endereço ao lado: "http://linguateca.di.uminho.pt/Paulo/stopwords/folha.MF100.txt".

Tabela 10 - Lista de cem *stopwords* utilizadas na etapa de Pré-processamento

a	da	em	já	nos	quando	Sua
à	das	entre	O RF D (	R	TX H	7D PE
DLQ G !	GH	HU D	PDL R	R QW H P	T XH P	7HP
DQ R	G HSR !	HVW £	PDL V	RV	U	7HU
D QR V	G HY H	HV WD (	PD V	RX	UL R	7R GR V
DR	GL D	HVW ¥ !	P H UF	S Dfl V	V¥ R	7U< V
DR V	GLV V !	HX	P H V P	S DU D	VH	8 P
D S HQ	GL]	IR L	PL O	S DX O R	V H JX	8 PD
DV	GR	I RO K D	PL OK (	S HO D	VH P	8 V
¢V	GRL V	I RU D !	PXL W !	S HO R	VH U	9DL
DW «	GR V	J R YH V	P XQ G	S H VV R	V HU £	
E UDV L	H	J U DQ	QD	S RG H	VH X	
FR P	«	K£	Q¥ R	SR U	V HX V	
F RP R	HO D	KRM H	QD V	S R UT X	V	
F RQW U	HOH	L VV R	QR	S UH V LC	V HE U !	

A existência dessas três listas de *stopwords*, criadas especificamente para o *corpus* utilizado nos estudos de casos, não justificou a adoção de outras listas pré-concebidas. No ***Snowball***[27]é possível encontrar *stoplists* para vários idiomas, incluindo o português.

Outro detalhe relacionado ao processo de remoção de *stopwords* é que a *stoplist* ideal deve ser construída levando em consideração o domínio do problema. Por exemplo, no caso de um sistema de Mineração de Textos para a área médica, termos como "exame" e "medicamento" poderiam ser incluídos na lista de *stopwords*, visto que, estas palavras, provavelmente, possuirão pouco poder discriminatório em relação a outros termos.

[27] Disponível em http://snowball.tartarus.org/

Portanto, assim que for obtida a coleção de documentos sob a qual será realizado o processo de Mineração de Textos, mesmo após a remoção das *stopwords* consideradas padrão, é interessante que seja analisada a distribuição das frequências de ocorrência dos termos que estão presentes na coleção. De acordo com (SALTON & BUCKLEY, 1988), termos com elevada frequência em muitos documentos constituem elementos de pouco poder discriminatório.

Portanto, além das três *stoplists*, o *framework* dispõe de mecanismo para análise de termos muito frequentes e com pouco caráter discriminatório, como citado acima. Esse mecanismo, simples, é baseado na seleção de termos com muita frequência nos documentos. Geralmente, esses termos são apresentados ao usuário para que ele dê decisão final sobre remover ou manter tais termos.

Para que o processo de seleção de *stopwords* relacionadas ao domínio ocorra sem intervenção humana, é preciso definir um valor de corte que será utilizado para remover automaticamente todos os termos que possuem frequência maior do que o valor definido. O valor adotado para eliminação de termos no *framework* é de oitenta por cento de frequência (SILVA A., 2007).

6.5.3. Processamento de Linguagem Natural

O uso de técnicas de Processamento de Linguagem Natural em Mineração de Textos tem o objetivo de identificar a real importância de cada termo em determinados contextos, possibilitando um ganho na qualidade dos resultados produzidos.

6.5.3.1. Identificação de Classes Gramaticais

Abordada desde a década de 50, a tarefa de *Part of Speech* (item 3.2.3.2) já foi testada por praticamente todos os métodos e algoritmos de Aprendizado de Máquina. Diversas outras tarefas de mais alto nível são dependentes desta tarefa base:

- Distinção de significados em aplicações de Recuperação de Informação;
- Solução de casos simples de ambiguidade;
- Facilita o entendimento/predição (*speech recognition*) de uma sentença:
    - Pronomes possessivos são seguidos por substantivos;
    - Pronomes pessoais são seguidos por verbos.
- Beneficia o processo de *text-to-speech*: fornecendo dados sobre a entonação com que uma determina palavra deve ser falada:
    - INsult(substantivo) / inSULT (verbo)
    - OBject (substantivo) / obJECT (verbo)

Entende-se por classe gramatical como a forma de classificação de um termo segundo seu significado dentro de uma sentença (CEGALLA, 2005). A Nomenclatura Gramatical Brasileira (NGB) divide os vocábulos em dez classes gramaticais, com a seguinte distribuição, ilustrada na Figura 29:

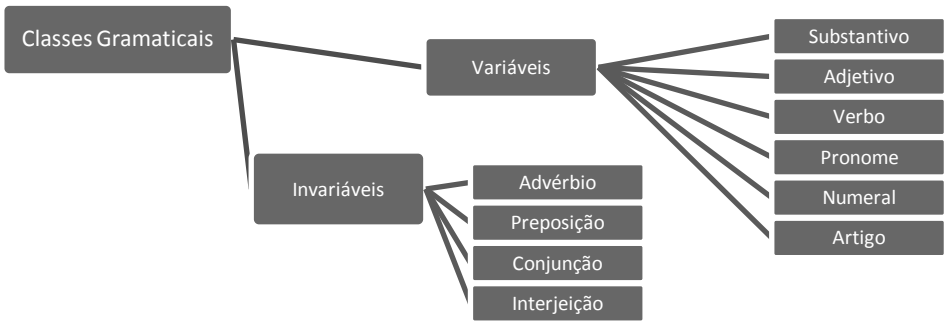


Figura 29 – Classes gramaticais segundo a NGB

Part of Speech Tagging ou Identificação de Classes Gramaticais é a primeira tarefa de Processamento de Linguagem Natural, baseada em Estatística, e possui como objetivo a identificação da classe gramatical correta de um vocábulo de acordo com o seu contexto, ou seja, a identificação de estruturas implícitas de um texto. A identificação da classe sintática de *tokens* é encarada na literatura como um típico problema de Classificação.

O problema a ser resolvido não é trivial, pois, muitas palavras possuem várias classes gramaticais, ou seja, quando fora de contexto, estas palavras

possuem ambiguidade em relação a como devem ser classificadas. Resolver as ambiguidades é a essência do problema de *Part of Speech Tagging*. Por exemplo, nas sentenças abaixo, exemplificadas na Tabela 11, uma mesma palavra pode assumir quatro classes gramaticais distintas:

The back door.	Adjetivo
On my back.	Substantivo
Win the voters back.	Advérbio
Promised to back the bill.	Verbo/Infinitivo

Tabela 11 - Exemplos ambíguos de identificação de classes gramaticais

O etiquetador mais conhecido para o Português brasileiro é o PALAVRAS, baseado em regras com *Constraint Grammar* de (BICK, 2000), e possui acurácia superior a 99%, mas não está disponível para ser embutido em aplicações.

No *framework*, uma das formas de seleção de termos será baseada nas classes gramaticais destes. Além disso, a heurística desenvolvida para a lematização necessita que verbos sejam identificados para que um léxico verbal construído nesta Tese (item 6.5.3.2) seja utilizado adequadamente. Portanto, é necessário implementar um Identificador de Classes Gramaticais no *framework*.

Em (MARKHAM & RUS, 2005), é proposto um *BaseLine System* que, embora de simples formulação, obtém taxa de Acurácia superior a 90% (sob o ***Corpus Penn TreeBank***[28], da língua inglesa). A heurística deste *BaseLine System* é simples e segue as seguintes etapas:

1. Treinamento: construção de um dicionário que armazena para cada vocábulo do *corpus* a lista de todas as classes gramaticais que cada vocábulo pode assumir e a frequência com que cada uma dessas classes gramaticais ocorre para cada vocábulo.
2. Recall » atribuir a cada palavra de uma nova sentença a classe gramatical que mais ocorre para a mesma (consulta ao dicionário). Caso a palavra não tenha sido previamente adicionada ao dicionário, isto é, seja desconhecida, atribuir à classe de substantivo.

[28] Disponível em http://www.cis.upenn.edu/~treebank/

O *BaseLine System* implementado neste trabalho segue as mesmas heurísticas descritas acima. Entretanto, a taxa de acurácia obtida para a língua Portuguesa, em torno de 70%, leva a necessidade de outra alternativa: Aprendizagem de Máquina.

Para a solução desse problema baseada em Aprendizagem de Máquina, o primeiro passo deste processo foi a preparação dos *data sets*. Analisando o conjunto de treinamento, este foi dividido em dois subconjuntos:

- O primeiro conjunto é composto apenas por vocábulos que não possuem ambiguidade de classe gramatical, ou seja, possuem apenas uma classe gramatical em todo o corpus.
- O segundo conjunto é composto por vocábulos que possuem mais de uma classe gramatical atribuída aos mesmos.

Esta separação foi realizada para que as tags ausentes de ambiguidade não sejam enviadas para o treinamento da SVM, já que um simples *BaseLine* irá obter 100% de Acurácia sobre este conjunto, o que irá diminuir o tempo de treinamento e aumentar o desempenho dos resultados obtidos pela SVM.

A função *kernel* escolhida foi a Linear, que embora de menor complexidade, obtém resultados satisfatórios com muito menos tempo de treinamento. A otimização do parâmetro *C* adotou a heurística gulosa, assumindo, respectivamente, os seguintes valores: 0.1, 1, 10, 100.

A modelagem dos dados segue a metodologia de *sliding window* de três termos e é ilustrada na Tabela 12:

Tabela 12 - Modelagem dos dados baseada em *sliding window*

Termos que antecedem e sucedem o termo a ser predito	$word_{-3}$
	$word_{-2}$
	$word_{-1}$
	$word_{0}$
	$word_{+1}$
	$word_{+2}$
	$word_{+3}$

Classe dos termos que antecedem e sucedem o termo a ser predito (fornecida pelo BaseLine)	pos_{-3}
	pos_{-2}
	pos_{-1}
	pos_0
	pos_{+1}
	pos_{+2}
	pos_{+3}
Começa com letra maiúscula?	sim/não
Comprimento da palavra	inteiro

O software utilizado para o treinamento e teste da SVM foi o SVMTool, proposto por (GIMÉNEZ, J. & MÀRQUES, 2004). Possui como principal vantagem a escolha da direção de *tagging*, permitindo que a mesma possa ser feita da esquerda para direita (padrão), ou da direita para esquerda, ou para ambas as direções. É divido nos quatro módulos citados abaixo:

- SVMT-learner [treinamento dos classificadores SVM];
- SVMT-tagger [POS-tagging de uma entrada qualquer];
- SVMT-evaluator [avaliação dos resultados de *tagging*];
- SVMT API [API para uso do SVMT-tagger treinado].

Outras vantagens deste software são o tempo de treinamento muito inferior a outros semelhantes (WEKA, por exemplo) e a classificação multiclasses automática (método de redução da SVM em problemas binários: "um contra todos").

O resultado geral obtido é de 91% para a taxa de acurácia. Para a necessidade da heurística de lematização, que necessita apenas identificar se um termo é, ou não, um verbo, o resultado obtido foi de 95%. No último caso, apenas uma classe gramatical foi considerada: verbo ou não verbo.

6.5.3.2. Lematização

Lematização ou *stemming* (item 3.2.3.4) é o processo de reduzir ao radical original palavras derivadas ou flexionadas deste. O principal objetivo da utilização

de um processo de *stemming* é reduzir a grande dimensionalidade das aplicações de Mineração de Textos, pois, com a remoção de prefixos e sufixos de palavras derivadas de um mesmo radical, e que, antes, seriam consideradas como *tokens* distintos, obtém-se um único *token* para a representação de todas elas.

Apesar dos benefícios acima, algoritmos de *stemming* também possuem desvantagens. Casos de *overstemming* e *understemming* são comuns em todos os algoritmos propostos (SILVA A. A., 2007). Todos os algoritmos de *stemming* são baseados em regras, e uma das principais causas de erro são os verbos irregulares, que geralmente são exceções às regras.

Para minimizar esse problema, um léxico de aproximadamente 10 mil verbos e suas conjugações foi obtido em parceria com o Departamento de Engenharia de Software da PUC-Rio para que, quando um verbo fosse identificado, a redução à forma canônica desse verbo fosse realizada por consulta ao léxico, ignorando os algoritmos de *stemming* nesses casos.

A utilização dessa abordagem minimizou os erros de *overstemming* e *understemming* em 40% para os termos verbais.

Para outras classes gramaticais, serão aplicados os algoritmos de *stemming*. A seleção dos algoritmos será realizada automaticamente baseada nos resultados obtidos na Categorização. Inicialmente, os algoritmos de *stemming* implementados no *framework* foram: Stemmer S, Método de Porter (adaptado ao Português) e Método de Lovins. Em seguida foi incluído também o RSLP Stemmer (Removedor de Sufixos da Língua Portuguesa), criado por (ORENGO & HUYCK, 2001), que constitui um dos mais modernos algoritmos de *stemming* concebidos para a língua Portuguesa.

6.5.3.3. Thesaurus

Outra técnica empregada é a utilização de um dicionário de relações hierárquicas entre termos, o dicionário *Thesaurus* (item 3.3.2), para substituir termos que possuam valores semânticos semelhantes ou relacionados, mas, são grafados de maneira distinta, sejam estes sinônimos ou exemplos de sinonímia. A

utilização deste dicionário evita equívocos ao realizar o cálculo de frequência destes termos. Para exemplificar, pode-se supor que em determinado documento sobre animais, encontramos dois termos de grande frequência: cão e cachorro. Porém, como possuem grafias distintas, estes termos serão computados separadamente, ainda que pelo fato de transmitirem a mesma ideia, deveriam ser considerados como somente um termo e ter frequências e outras métricas somadas. Outro benefício proporcionado por estes dicionários é relação entre termos. Retornando ao exemplo acima, com a ajuda do dicionário, verifica-se que cachorro e cão possuem valores semânticos aproximados e pertencem a uma categoria superior: a de animais de estimação que também pertence à categoria de animais. O dicionário *Thesaurus,* construído durante a execução deste trabalho, e que foi utilizado neste estudo de caso possui cerca de 13 mil termos preferenciais e 17 mil termos não preferenciais, todos classificados em dezenas de categorias. A estrutura de dados utilizada na construção deste dicionário é ilustrada na Figura 30.

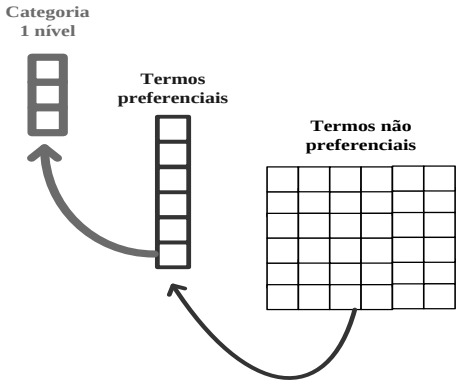


Figura 30 - Estrutura do dicionário Thesaurus utilizado no Sistema de MT

No contexto de Thesaurus, conforme citado em (BARROS, 2010), há também a WordNet.BR com aproximadamente 44 mil termos que possuem relações de antonímia e sinonímia. Está disponível por meio do Thesaurus Eletrônico para o Português do Brasil[29] (TeP). A base de dados desse dicionário pode ser utilizada para estudos linguísticos.

Esta base foi também incorporada pelo *framework* desenvolvido nesta Tese.

[29] Disponível em http://www.nilc.icmc.usp.br/tep2/index.htm

Cerca de trinta e dois por cento dos *tokens* foram consultados e ou substituídos por termos do dicionário. Além disso, com a utilização deste dicionário, houve redução média de sete por cento no número de *tokens* encontrados (*tokens* multivocabulares, etapa 6, item 6.5.1).

6.5.4. Redução de características

Para uma representação adequada dos textos faz-se necessária a seleção dos termos mais representativos, pois em virtude do tamanho destes e de sua quantidade de *tokens*, os mesmos podem ser intratáveis pelos algoritmos de classificação. Este processo é conhecido como redução de características.

A redução de características pode ser do tipo:

- Global: consiste na escolha dos termos mais importantes para a coleção.
- Local: consiste na escolha de um conjunto de termos para cada categoria.

A natureza da redução é realizada por meio da seleção das características dos documentos da coleção, ou seja, as características selecionadas são um subconjunto das características originais. Esta redução geralmente é realizada através da seleção das melhores características, de acordo com algum critério. A grande dificuldade dessa abordagem é selecionar um subconjunto de termos que possa fazer uma discriminação adequada entre as várias categorias, e ao mesmo tempo, ser pequeno o suficiente para que possa ser utilizado pelo classificador. As técnicas mais utilizadas para este fim são baseadas na frequência de palavras (item 4.3.1.3), como TF/IDF, Ganho de informação e de Escore de Relevância (WIENER, 1995).

A escolha do conjunto de características possui enorme impacto no processo de Categorização. O *framework* é capaz de selecionar os termos mais importantes de cada documento segundo as duas abordagens. A escolha de uma ou outra abordagem é determinada pelo resultado obtido no processo de otimização da Categorização de Textos.

6.5.5. Indexação

Indexação é fase responsável por criar estruturas de dados denominadas índices, capazes de permitir que uma consulta seja realizada sem que seja necessário analisar toda uma base de dados (MANNING, RAGHAVAN, & SCHÜTZE, 2007).

Atualmente, a técnica mais utilizada para a indexação textual é a de **índices invertidos**[30] (KONCHADY, 2006). Essa técnica é suportada no *framework* proposto por meio da utilização da biblioteca **Lucene**[31]. O processo de tokenização é realizado com a heurística desenvolvida no *framework* (item 6.5.1).

6.5.6. Classificadores implementados

O *framework* dispõe de três classificadores para realizar a atividade de classificação: *k-NN*, SVM e Classificador Bayesiano.

A implementação do algoritmo *k-NN* permite que as seguintes métricas possam ser usadas para calcular a distância entre cada instância: Distância Euclidiana e Cálculo do Cosseno.

A implementação do SVM é baseada na versão implementada por (JOACHIMS, 1998) chamada de SVM^{light}, disponível em seu próprio endereço na Internet[32].

A implementação do Classificador Bayesiano é baseada na versão disponível em Java no software Weka. A classe *NaiveBayes* foi reescrita para C#.

---

[30] Recebem essa denominação por inverter a hierarquia da informação. No lugar de uma lista de documentos contendo termos, tem-se uma lista de termos referenciando documentos.

[31] Disponível em http://lucenenet.apache.org/

[32] http://svmlight.joachims.org

6.5.7.
Técnicas de combinação de classificadores

O método de escolha por voto majoritário está implementado como premissa para toda operação de classificação que utiliza como classificador o SVM, pois este é um classificador binário.

Além disso, também foi implementada a votação em que pesos são atribuídos aos classificadores (votação com pesos); ou seja, o voto de cada classificador é relativo ao seu peso, para a decisão final. O peso é proporcional ao índice de acerto de cada classificador na categoria.

Para categorias em que um determinado classificador é muito eficiente, utiliza-se um mecanismo de seleção dinâmica de classificadores denominado *Best By Class*.

Além disso, o uso de *bagging* é natural, pois diversas heurísticas de pré-processamento são combinadas, produzindo alterações na forma de representação dos documentos.

6.6.
Corpus

A primeira etapa (ver 3.1) do processo de Mineração de Textos é a coleta. Essa etapa é responsável por obter os documentos que farão parte da coleção de documentos.

O processo de Categorização de Textos constitui um paradigma de Aprendizagem de Máquina supervisionado, portanto necessita que ao apresentar a instância de treinamento, esta além de conter os atributos que a representem, deve conter também a resposta desejada (categoria) para a categorização da mesma.

Na língua inglesa, diversos *corpora etiquetados são* utilizados como *benchmark* de Categorização de Textos. Temos como exemplo o *corpus* **Reuters-215783**[33] que é o mais utilizado em todo o mundo nesta atividade. Na língua

[33] Disponível em http://www.daviddlewis.com/resources/testcollections/reuters21578/

portuguesa do Brasil, o *corpus* que predomina nos estudos sobre Categorização de Textos é o CETENFolha, conforme mostrado nos trabalhos relacionados.

CETENFolha, Corpus de Extratos de Textos Eletrônicos NILC/Folha de S. Paulo, é um corpus de cerca de vinte e quatro milhões de palavras em Língua Portuguesa do Brasil, criado pelo projeto de Processamento Computacional do Português com base nos textos do jornal Folha de S. Paulo. Este corpus possui aproximadamente 341 mil porções de textos, classificados por semestre e caderno do jornal do qual provém. Informações adicionais sobre o corpus CETENFolha são apresentadas na Tabela 13.

Tabela 13 - Informações adicionais sobre o CETENFolha

Quantidade de palavras	24 milhões
Idioma	Português (Brasil)
Origem	Textos publicados no Jornal Folha de S. Paulo
Período de Publicação	1994
Quantidade de edições	365
Classificado?	Pelo tipo de caderno a que pertence a notícia
Etiquetado?	Não

A escolha desse *corpus* dispensa a execução da etapa de coleta de dados. Para o estudo de casos desta Tese, os documentos que serviram como base do experimento foram os textos jornalísticos obtidos do *corpus* CETENFolha. Desta forma, foram selecionados documentos textuais deste *corpus*, ou seja, notícias, que possuem mais de sessenta termos para o estudo de casos. Será utilizado como categoria dos textos obtidos desse corpus o caderno de onde cada um provém.

Cada texto deste arquivo possui uma quantidade considerável de metadados associados, como identificador do texto, título, o próprio texto, semestre, caderno no jornal e autor da notícia. A categoria a que cada notícia pertence está definida em *"cad"*. Todas as outras informações fornecidas pelos metadados foram ignoradas e não fazem parte do processo de Categorização.

Na Figura 31 é possível visualizar um exemplo de texto jornalístico, em seu formato original, presente neste corpus. As *tags* “<ext>” “</ext>” delimitam o início e fim de cada notícia ou documento; as *tags* “<t>” e “</t>” delimitam o título da notícia; as *tags* “<a>” e “” delimitam o autor da mesma; cada

parágrafo é identificado pelas *tags* “ ” e “ ”; e cada sentença da notícia fica compreendida entre as *tags* “<s>” e “</s>”.

```
<ext id=1 cad="Opinião" sec="opi" sem="94a">
 <s>
  <t> PT no governo </t>
 </s>
 <s>
  <a> Gilberto Dimenstein </a>
 </s>
 <p>
  <s> BRASÍLIA Pesquisa Datafolha publicada hoje revela um dado supreendente:
recusando uma postura radical, a esmagadora maioria (77%) dos eleitores quer o PT
participando do Governo Fernando Henrique Cardoso . </s>
  <s> Tem sentido -- aliás, muitíssimo sentido . </s>
 </p>
 <p>
  <s> Muito mais do que nos tempos na ditadura, a solidez do PT está, agora,
ameaçada . </s>
  <s> Nem Lula nem o partido ainda encontraram um discurso para se diferenciar . </s>
  <s> Eles se dizem oposição, mas ainda não informaram o que vão combater . </s>
  <s> Muitas das prioridades do novo governo coincidem com as prioridades do PT . </s>
 </p>
</ext>
```

Figura 31 - Exemplo de documento do corpus CETENFolha

Removendo as *tags* e excluindo as informações adicionais de metadados, o que será utilizado no processo de categorização é transcrito abaixo:

> PT no governo
>
> BRASÍLIA Pesquisa Datafolha publicada hoje revela um dado supreendente: recusando uma postura radical, a esmagadora maioria (77%) dos eleitores quer o PT participando do Governo Fernando Henrique Cardoso .
>
> Tem sentido -- aliás, muitíssimo sentido .
>
> Muito mais agora do que nos tempos da ditadura, a solidez do PT está, agora, ameaçada .
>
> Nem Lula nem o partido ainda encontraram um discurso para se diferenciar .
>
> Eles se dizem oposição, mas ainda não informaram o que vão combater .
>
> Muitas das prioridades do novo governo coincidem com as prioridades do PT .

Os cadernos de notícias (categorias) selecionados foram os de esporte, imóveis, informática, política e turismo. Optou-se por esses cadernos para que os resultados obtidos pudessem ser comparados com outros resultados obtidos, como em (CAMARGO, 2007) e (LINDEN, 2008).

6.7.
Assistência Inteligente

O *framework* dispõe de uma base de conhecimentos sobre Categorização de Textos baseada em (BORKO & BERNICK, 1963), (SEBASTIANI, 2002) (SHOLOM, INDURKHYA, ZHANG, & DAMERAU, 2005), (KONCHADY, 2006) e (KUDO & MATSUMOTO, 2004) que relatam as principais heurísticas utilizadas neste paradigma. (JOACHIMS, 1998) provê informações sobre a utilização de SVM e as técnicas de pré-processamento adequadas para Categorização de Textos com SVM. (BAEZA-YATES & BERTIER, 1999) relata as abordagens de pré-processamento para Recuperação de Informação, como os modelos de representação de documentos e as métricas utilizadas para representar a importância de cada termo. E os trabalhos relacionados no item 1.3 abordam diversas técnicas utilizadas para Categorização de Textos em Português.

A construção dessa base de conhecimento permite ao *framework* decidir as melhores opções de pré-processamento e ajustes de parâmetros dentre as disponíveis na ferramenta, reduzindo desta forma, o espaço de busca por uma solução ótima. Contudo, há novas abordagens às tarefas de pré-processamento desenvolvidas nesta Tese e que, portanto, nunca foram avaliadas anteriormente.

Foi elaborado um fluxo das etapas que devem ou podem ser seguidas no processo de Categorização de Textos para cada um dos algoritmos de classificação. Além disso, os valores desejáveis para a variação dos parâmetros de cada algoritmo de classificação são especificados.

Por exemplo, para o algoritmo *k-NN*, com base na literatura, os valores do parâmetro *k* devem variar entre um e quinze (SEBASTIANI, 2002). Para evitar a ocorrência de empate e ser necessária uma heurística adicional para a seleção da categoria predita nesses casos, opta-se por utilizar valores ímpares para *k*, portanto *k* deve variar sempre em *+2* ou *-2* a partir de seu valor inicial no intervalo especificado. Além disso, quatro tipos de medidas são utilizados para calcular a distância entre uma instância e seus vizinhos: Euclidiana, Cosseno, Jaccard e

Manhattan. Essas medidas são especificadas como funções na ordem de execução preferida.

KNN	
Parâmetros	Tipo
Parâmetro k	Numérico
Distância	Função

Valor mínimo	Valor máximo	Valor inicial	Incremento	Decremento
1	15	5	+2	-2

Função	Definição	Ordem execução
1. Euclidiana	...	1
2. Cosseno	...	2
3. Jaccard	...	3
4. Manhattan	...	4

Figura 32 - Exemplo de modelagem de execução para o *k*-NN

O planejamento das ações é modelado como um uma lista de tarefas que possui precedência de ações. A Tabela 14 exibe a modelagem do plano de algumas tarefas do *framework*. A precedência de ações delimita em que momento uma tarefa pode ser executada:

- Tarefas que possuem uma ou mais tarefas como precedentes podem ser executadas com os resultados distintos de cada uma das tarefas precedentes. Por exemplo, a tarefa *bb* pode ser executada com a saída do processamento das tarefas *b* e *c*. Isso constitui dois caminhos de execução para a tarefa *bb*.

Tabela 14 - Planejamento de ações

Id	Etapa	Tarefa	Antecedente	Tipo Entrada	Tipo Saída
a	*Pré-Processamento*	*Início de etapa*	*N/D*	*N/D*	*N/D*
b	Pré-Processamento	Tokenização	a	*A0*	*A1*
c	Pré-Processamento	Remoção de stopwords	b	*A1*	*A1*
d	Pré-Processamento	Remoção de stopwords (domínio)	c	*A1*	*A1*
z	*Pré-Processamento*	*Fim de etapa*	*N/D*	*N/D*	*N/D*
aa	*PLN*	*Início de etapa*	*N/D*	*N/D*	*N/D*
bb	PLN	POS tagging (verbo/não verbo)	b ou c	*A1*	*A2*
cc	PLN	POS tagging (completo)	b ou c	*A1*	*A2*

dd	PLN	Lematização verbal	bb	*A2*	*A1*
ee	PLN	Lematização PORTER	bb	*A2*	*A1*
ff	PLN	Lematização PORTER	bb	*A2*	*A1*
gg	PLN	Lematização LOVINS	bb	*A2*	*A1*
hh	PLN	Lematização RSLP	bb	*A2*	*A1*
...	...	...	...	...	...
zz	*PLN*	*Fim de etapa*	*N/D*	*N/D*	*N/D*

Um construtor de ações foi desenvolvido no *framework* para que o planejamento de ações sempre seja válido. Para isso, as informações sobre precedência e tipos de entrada e saída são utilizadas. Os tipos de entrada e saída definem o formato dos dados que são processados pelas tarefas. Por exemplo, o tipo de entrada *A0* constitui um conjunto de cadeias de caracteres (documento textual). Desta forma, observando a Tabela 14 é possível concluir que somente a tarefa *b* está apta a trabalhar com a informação neste formato.

7
Estudos de Caso

7.1.Coleta

A coleção de documentos contém textos jornalísticos do *corpus* citado em 6.6. Os textos foram extraídos dos cadernos de esporte, imóveis, informática, política e turismo. Foram selecionados trezentos documentos de cada caderno, perfazendo um total de 1500 documentos.

7.2.
Treinamento

A metodologia de treinamento que será utilizada é a Validação Cruzada citada em 5.8.2: o conjunto de amostras inicial é dividido em *k* subamostras. Destas *k* subamostras, uma subamostra é retida para ser utilizada na validação do modelo (conjunto de teste) e as *k-1* subamostras compõem o conjunto de treinamento. O processo é então repetido *k* vezes, de modo que cada uma das *k* subamostras seja utilizado ao menos uma vez como teste. O resultado final é a média do desempenho do classificador nas *k* iterações.

7.3.
Resultados

7.3.1.
Tokenização

No *framework*, o início do processo de Categorização é dado pela execução da tarefa *a* que corresponde à etapa de Tokenização (Tabela 15). Essa tarefa fará uma

transformação na apresentação dos dados: recebe os dados com o formato *A0* (conjunto de cadeia de caracteres ou documentos) e os transforma em *A1* (conjunto de *tokens*).

Tabela 15 - Planejamento de ações: Tokenização

Id	Etapa	Tarefa	Antecedente	Tipo Entrada	Tipo Saída
a	*Pré-Processamento*	*Início de etapa*	*N/D*	*N/D*	*N/D*
b	**Pré-Processamento**	**Tokenização**	**a**	***A0***	***A1***

Geralmente, o processo de Tokenização é muito rápido. Porém, a metodologia empregada nesta Tese para a realização desta tarefa é baseada em consultas aos léxicos construídos além de grande processamento estatístico (cálculo do coeficiente de correlação) para automatizar o processo de Identificação de *tokens* multivocabulares, o que torna o processo bastante oneroso.

Para efeito de comparação, a realização da tokenização utilizando apenas o passo *1* da metodologia proposta nesta Tese, que consiste na geração simples de *tokens* baseada no conjunto de *tokens* delimitadores, como espaço e fim de linha, consome em média cinco minutos. Esta é a abordagem utilizada pela maioria dos softwares de MT. A metodologia empregada neste trabalho utiliza aproximadamente oitenta minutos de processamento.

Ao final do processo, onze mil *tokens* são gerados. Pela consulta realizada pelo *framework* ao léxico, trezentas e quinze abreviações foram identificadas e substituídas pelo termo não abreviado correspondente.

Além disso, aproximadamente quatrocentos *tokens* multivocabulares foram encontrados. Como o processo de identificação desses *tokens* é realizado sem intervenção do usuário, utiliza-se limite de 40 no valor de correlação entre termos para que os mesmos sejam unificados em um único termo. Contudo, caso necessário, o usuário poderá selecionar manualmente os termos justapostos que irão ser compilados em um único *token*.

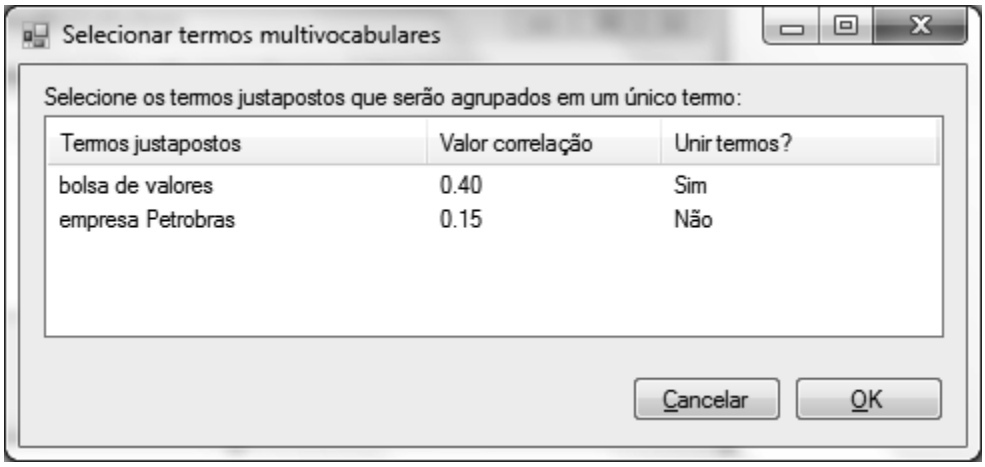


Figura 33 - Exemplo de documento do corpus CETENFolha

A etapa de identificação de números foi responsável pelo reconhecimento de 680 termos que expressam grandezas numéricas. Números e datas na forma extensa são substituídos pela sua representação numérica. A fase de identificação de símbolos de Internet encarregou-se de discriminar cerca de 535 *tokens* distintos. A maioria proveniente do caderno de informática. A etapa de identificação de palavras combinadas contribuiu de forma insignificante para o processo de tokenização.

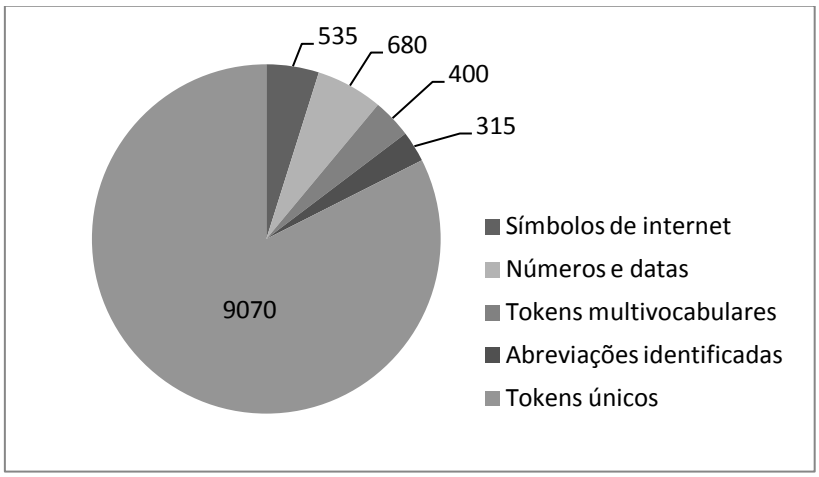


Figura 34 - Exemplo de documento do corpus CETENFolha

O final do processo de tokenização obtém-se a situação ilustrada na Figura 35: um conjunto de documentos em sua representação baseada em *tokens*, também chamada de *bag of words* ou saco de palavras.

Figura 35 - Representação de documentos na forma de *bag of words*

7.3.1. Remoção de *stopwords*

Em seguida, inicia-se o processo de remoção e análise de *stopwords.* Essa fase corresponde às tarefas *c* e *d* da tabela de planejamento de ações (Tabela 14). A tarefa *d* só é executada após a conclusão da tarefa *c* (*c* é antecedente de *d)*. Nessa fase, os documentos textuais já estão segmentados em *tokens* (tipo de saída *A1*), resultado da execução da tarefa *b* (Tokenização), portanto os dois processos, *c* e *d*, recebem como entrada o formato de dados *A1* (Tabela 16).

Tabela 16 - Planejamento de ações: Remoção de *stopwords*

Id	Etapa	Tarefa	Antecedente	Tipo Entrada	Tipo Saída
b	Pré-Processamento	Tokenização	a	*A0*	*A1*
c	**Pré-Processamento**	**Remoção de stopwords**	**b**	***A1***	***A1***
d	**Pré-Processamento**	**Remoção de stopwords (domínio)**	**c**	***A1***	***A1***

Automaticamente, são gerados três subconjuntos de *tokens* resultantes desse processo, conforme ilustrado na Figura 36. Todos são o resultado da utilização das três listas de *stopwords* fornecidas para a coleção: uma com cem termos, a segunda com duzentos e a última com trezentos termos. Cada um desses subconjuntos será mantido para que sejam avaliados pelos métodos de Categorização de Textos posteriormente.

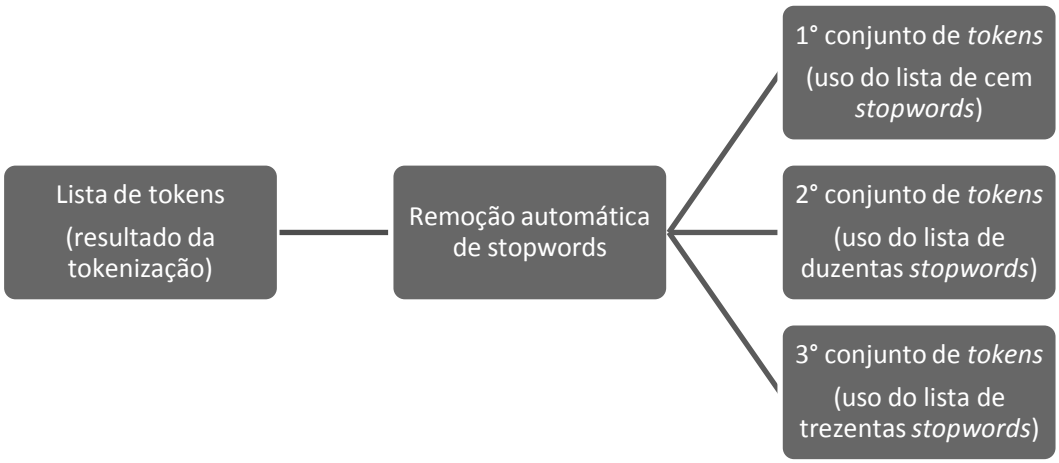


Figura 36 - Resultado do processo de remoção de *stopwords* baseado em listas

Sob os três subconjuntos de *tokens* gerados pela tarefa *c*, será aplicada a tarefa *d*, que consiste na remoção de *stopwords* relacionadas ao domínio. Esse procedimento é necessário, pois muitos termos relacionados ao domínio da aplicação são considerados irrelevantes, pois devido á alta frequência com que estão presentes nos documentos, possuem pouco caráter discriminatório.

Portanto, termos frequentes em mais de oitenta por cento dos documentos ou com frequência menor ou igual a três, devem ser eliminados, segundo (JOACHIMS, 1998).

Ao fim do término da execução das duas tarefas de remoção de *stopwords*, o número de termos foi reduzido substancialmente. O primeiro subconjunto de dados, oriundo da aplicação da lista de cem *stopwords*, possui cerca de 3600 termos. O segundo subconjunto de dados, e o terceiro conjunto, oriundo da aplicação da lista de trezentas *stopwords*, possuem cerca de 3500 termos.

Pode-se concluir que apesar de utilizarem listas de *stopwords* diferentes, todos os subconjuntos de termos foram reduzidos a valores próximos. A execução da tarefa *d*, remoção de *stopwords* do domínio, foi capaz de obter representações semelhantes aos três subconjuntos de dados.

Ao final dessa tarefa, a situação dos documentos é ilustrada na Figura 37. Há, portanto, seis subconjuntos de dados disponíveis para os métodos de Categorização de Textos.

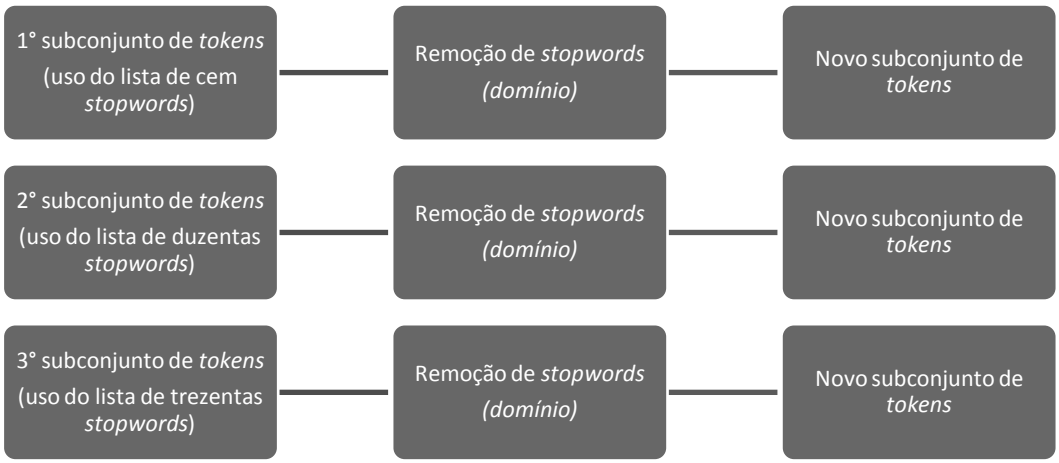


Figura 37 - Resultado do processo de remoção de *stopwords* do domínio

7.3.2. PLN - Identificação de classes gramaticais

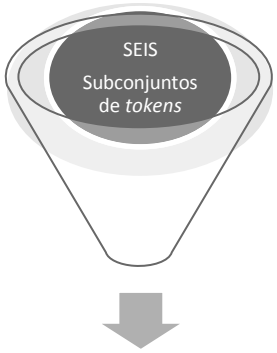
A fase de Processamento de Linguagem Natural é executada. As tarefas *b* e *c* podem ser executadas em paralelo, pois possuem as mesmas tarefas antecedentes (*c* ou *d*). Essas tarefas esperam como entrada o tipo de dados *A1* e ao final do processo irão gerar uma representação dos *tokens* que contém a identificação da classe gramatical a que cada um pertence (formato de dados *A2).* Essas tarefas serão executadas nos seis subconjuntos de dados existentes até o momento (Tabela 17).

Tabela 17 - Planejamento de ações: PLN - Identificação de classes gramaticais

Id	Etapa	Tarefa	Antecedente	Tipo Entrada	Tipo Saída
aa	*PLN*	*Início de etapa*	*N/D*	*N/D*	*N/D*
bb	**PLN**	**POS tagging (verbo/não verbo)**	**b ou c**	***A1***	***A2***
cc	**PLN**	**POS tagging (completo)**	**b ou c**	***A1***	***A2***

A tarefa *bb* distingue apenas *tokens* verbais de não verbais para que todos os verbos sejam reduzidos a sua forma canônica por meio de consulta ao léxico, ignorando os algoritmos tradicionais de *stemming.* A tarefa *cc* irá identificar as dez classes gramaticais ilustradas na Figura 29 para que as tarefas de seleção de

termos possam selecionar os termos com base nas classes gramaticais de cada um. De forma geral, o processo pode ser ilustrado como na Figura 38.



Identificação de classes gramaticais
Verbos e não verbos
Classes gramaticais básicas

Figura 38 - Identificação de classes gramaticais

7.3.3. PLN - Lematização

Após a identificação das classes gramaticais, a etapa de lematização é iniciada. Cinco tarefas, *dd*, *ee*, *ff*, *gg* e *hh*, serão executadas para cumprir esta etapa (Tabela 18). Todas as tarefas irão trabalhar com o formato de dados gerado pela etapa de Identificação de Classes Gramaticais, isto é, o formato *A2*¸ que é uma representação de termos e suas respectivas gramaticais, e irão retornar novamente um conjunto de *tokens* (formato *A1*).

Tabela 18- Planejamento de ações: PLN - Lematização

Id	Etapa	Tarefa	Antecedente	Tipo Entrada	Tipo Saída
aa	*PLN*	*Início de etapa*	*N/D*	*N/D*	*N/D*
dd	**PLN**	**Lematização verbal**	**bb**	***A2***	***A1***
ee	**PLN**	**Lematização PORTER**	**bb**	***A2***	***A1***
ff	**PLN**	**Lematização PORTER**	**bb**	***A2***	***A1***
gg	**PLN**	**Lematização LOVINS**	**bb**	***A2***	***A1***
hh	**PLN**	**Lematização RSLP**	**bb**	***A2***	***A1***

A tarefa *dd*, isto é, a Lematização verbal, será responsável somente pela lematização dos verbos e, portanto só possui como antecedente a tarefa *bb*. Desta forma, todo *token* diferente de verbo será ignorado. *Tokens* verbais serão lematizados por consulta ao léxico. Esse processo está ilustrado na Figura 39. A execução da tarefa *dd* irá gerar um novo subconjunto de *tokens* que possuem verbos reduzidos à sua forma canônica.

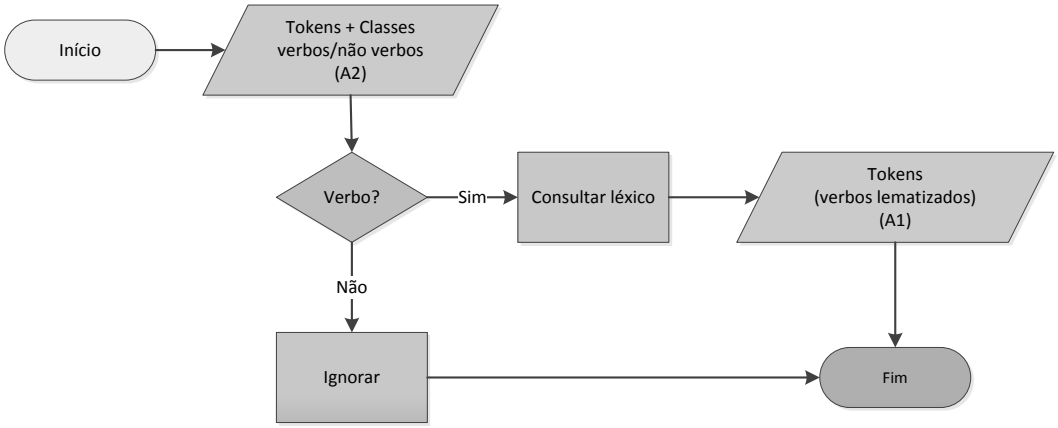


Figura 39 - Fluxograma da lematização verbal

As tarefas *ee*, *ff*. *gg* e *hh*, serão responsáveis pela lematização de todos os termos não verbais. Possuem como antecedente a tarefa *bb* que é responsável pela identificação de termos verbais e não verbais, pois termos verbais não deverão ser lematizados por esses algoritmos. Esse processo está ilustrado na Figura 40 em que o processo *Lematizar* envolve a aplicação dos quatro algoritmos de lematização implementados no *framework*. Desta forma, ao final desse processo haverá quatro novos subconjuntos de *tokens* gerados por cada um dos algoritmos.

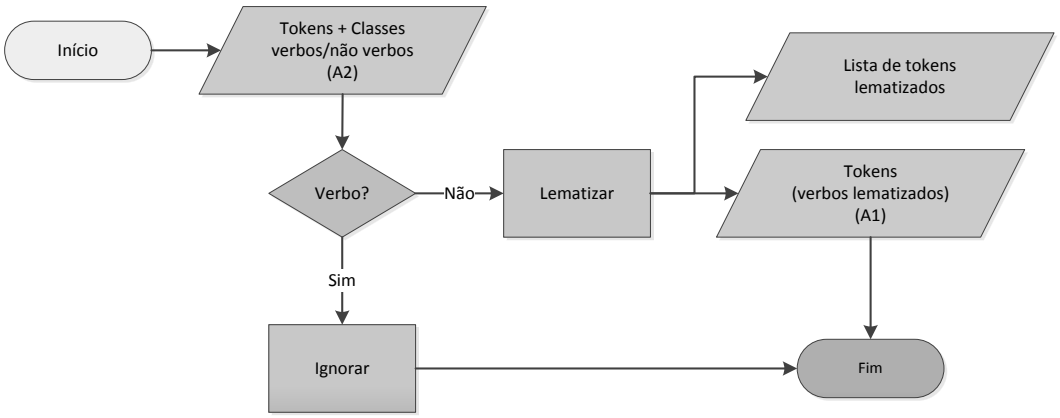


Figura 40 - Fluxograma da lematização não verbal

Além dos *tokens* substituídos pela forma canônica ou lematizados por um dos algoritmos de *stemming* também é gerada uma lista de *tokens* que foram modificados. Essa lista é necessária para que seja possível identificar todos os *tokens* que sofrerão alguma transformação. A cada um dos quatro subconjuntos de *tokens* lematizados pelos algoritmos serão incluídos os verbos substituídos, já que esses algoritmos ignoraram os verbos durante a sua execução.

Ao fim de todo o processo de lematização, há, portanto, cinco novas opções de subconjuntos (Lematização verbal, Porter, Stemmer S, Lovins e RSLP) para cada um dos seis subconjuntos iniciais.

7.3.4. Thesaurus

A tarefa *ii* compreende a execução desse processo em que termos com o mesmo valor semântico são identificados e substituídos por um termo preferencial (Tabela 19). O léxico construído durante esta Tese e a base de Thesaurus Eletrônico para o Português do Brasil são utilizados.

Tabela 19- Planejamento de ações: PLN - Thesaurus

Id	Etapa	Tarefa	Antecedente	Tipo Entrada	Tipo Saída
Aa	*PLN*	*Início de etapa*	*N/D*	*N/D*	*N/D*
ii	**PLN**	**Thesaurus**	**dd ou ee ou ff ou gg ou hh**	***A1***	***A1***

Os dicionários utilizados, além dos termos na forma original, contém uma representação dos mesmos lematizados segundo cada um dos métodos utilizados na etapa anterior para que seja possível encontrá-los. Essa etapa foi responsável, em média, pela substituição de vinte por cento dos *tokens*, conforme ilustrado na Figura 41.

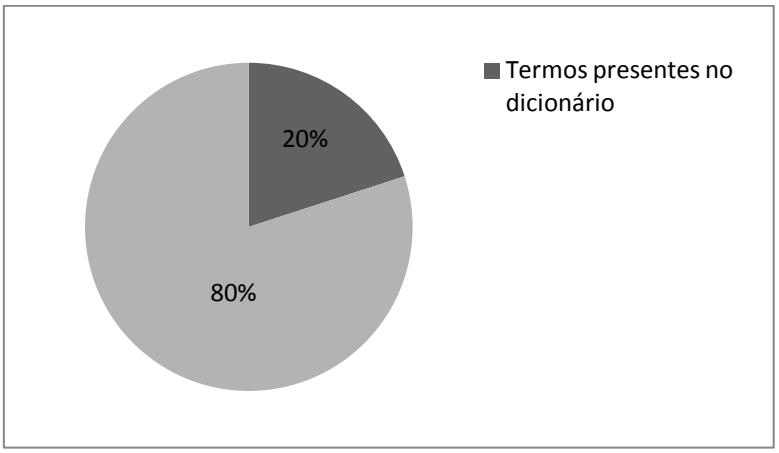


Figura 41 - Substituição de termos por consulta ao Thesaurus

Ao fim de todo o processo de Thesaurus, há, portanto, duas novas opções de subconjuntos (com ou sem uso de Thesaurus) para cada um dos trinta subconjuntos do início do processo.

7.3.5. Seleção de características

Em seguida, inicia-se o processo de seleção ou redução de características. Essa fase corresponde às tarefas *jj*, *kk*, *ll* e *mm* da tabela de planejamento de ações (). Essas tarefas visam diminuir a alta dimensionalidade dos modelos de representação de documentos.

A tarefa *mm*, seleção de características POS (*Part of Speech*), irá fazer a seleção de características em que a informação linguística define a importância dos termos. Utiliza a abordagem de seleção de características baseada em padrões morfossintáticos, ou seja, utiliza as classes gramaticais dos termos para fazer a seleção do que será considerado na representação reduzida dos documentos.

Portanto, esta tarefa é obrigatoriamente executada após a conclusão da tarefa *cc* que é a identificação de classes gramaticais. Logo, a tarefa *mm* possui como entrada, o modelo de dados *A2* que é o modelo que fornece os *tokens* associados às suas respectivas classes gramaticais. Além disso, na elaboração do planejamento de tarefas do *framework*, incluiu-se também que esta tarefa aguarde a execução de pelo menos uma das tarefas de lematização (tarefas de *dd* ou *ee* ou *ff* ou *gg* ou *hh*), conforme representado na Tabela 20.

As tarefas restantes (*kk, ll* e *mm)* irão fazer a extração de características baseadas em critérios estatísticos (item 4.3.1.3), a saber: TF-IDF, Ganho de Informação e Escore de Relevância. Podem ser iniciadas tão logo a tokenização seja concluída, porém no planejamento de ações do *framework*, optou-se por permitir a execução dessas tarefas somente após a execução dos métodos de lematização. (tarefas de *dd* ou *ee* ou *ff* ou *gg* ou *hh).*

Tabela 20- Planejamento de ações: Seleção de características

Id	Etapa	Tarefa	Antecedente	Tipo Entrada	Tipo Saída
aa	*PLN*	*Início de etapa*	*N/D*	*N/D*	*N/D*
jj	**PLN**	**Seleção de características: TF/IDF**	**dd ou ee ou ff ou gg ou hh**	***A1***	***A1***
kk	**PLN**	**Seleção de características: Ganho de Informação**	**dd ou ee ou ff ou gg ou hh**	***A1***	***A1***
ll	**PLN**	**Seleção de características: Escore de relevância**	**dd ou ee ou ff ou gg ou hh**	***A1***	***A1***
mm	**PLN**	**Seleção de características: POS**	**cc e (dd ou ee ou ff ou gg ou hh)**	***A2***	***A1***

A tarefa de seleção de características POS utilizará a combinação de classes gramaticais, como em (CAMARGO, 2007), para escolher os termos. Serão formados sete subconjuntos constituídos de: substantivo, substantivo + adjetivo, substantivo + nome próprio, substantivo + verbo, substantivo + verbo + adjetivo, substantivo + nome próprio + adjetivo, nome próprio + adjetivo e verbo.

Os resultados estatísticos obtidos pelas tarefas *kk*, *ll* e *mm* serão utilizados em ordem decrescente para determinar os termos que irão fazer parte dos modelos reduzidos de representação dos documentos.

Assim que concluído todo o processo de cálculo das métricas, é necessário definir a quantidade de termos que será utilizada na representação reduzida dos documentos para cada uma das quatro tarefas de seleção de características. Há dois critérios para isso: seleção global e seleção local. Esse processo está ilustrado na Figura 42.

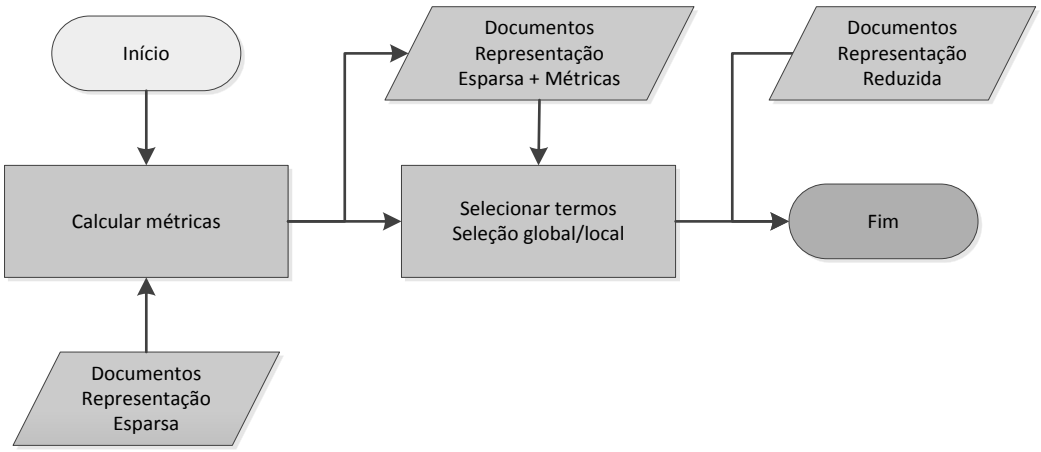


Figura 42 - Seleção de características

A seleção global compreende a escolha dos termos mais representativos de toda a coleção segundo as métricas calculadas nos processos de seleção de características: TF/IDF, Ganho de Informação e Escore de Relevância. Os experimentos utilizam os 30, 60, 90, 120 e 150 termos mais relevantes, conforme sugerido em (SEBASTIANI, 2002) e (JOACHIMS, 1998).

A seleção local compreende a escolha dos termos mais representativos de cada categoria segundo as mesmas métricas calculadas no processo de seleção de características: TF/IDF, Ganho de Informação e Escore de Relevância. Foram selecionados os 6, 12, 18, 24 e 30 termos mais relevantes de cada categoria; como há cinco categorias, serão construídas ao final da seleção dos termos mais relevantes da cada categoria representações de documentos com os seguintes números de termos: 30, 60, 90, 120 e 150.

Ao fim de todo o processo de seleção de características, há, portanto, para cada um dos três métodos baseados em estatísticas para seleção de características (seleção por TF/IDF, Ganho de Informação, Escore de Relevância) dez .novos subconjuntos de dados: cinco conjuntos formados por seleção global e cinco formados por seleção local. O método baseado em classes gramaticais para

seleção de características possui também dez novos subconjuntos de dados, formados por seleção global e seleção local, para cada uma das sete combinações de classes gramaticais.

7.3.6. Mineração

Compreende a execução das tarefas *bbb, cccc e dddd*, ou seja, a aplicação dos algoritmos classificadores *k*-NN, SVM e Bayesiano (Tabela 21) . Os conjuntos de dados que serão utilizados para treinar os classificadores foram definidos na etapa anterior.

Tabela 21- Planejamento de ações: Mineração

Id	Etapa	Tarefa	Antecedente	Tipo Entrada	Tipo Saída
aaaa	*Mineração*	*Início de etapa*	*N/D*	*N/D*	*N/D*
bbbb	**Mineração**	**KNN**	**jj ou kk ou ll ou mm**	***A3***	***A4***
cccc	**Mineração**	**SVM**	**jj ou kk ou ll ou mm**	***A3***	***A4***
dddd	**Mineração**	**Classificador Bayesiano**	**jj ou kk ou ll ou mm**	***A3***	***A4***
zzzz	*Mineração*	*Fim de etapa*	*N/D*	*N/D*	*N/D*

É nesta etapa também que os algoritmos serão ajustados para obter o melhor desempenho em cada uma das representações obtidas ao término da etapa de PLN. A Tabela 22 exibe a modelagem das características de execução do algoritmo SVM e a Tabela 23 exibe a modelagem referente ao algoritmo *k*-NN.

Tabela 22 - Configurações de execução do algoritmo SVM

SVM	
Parâmetros	Tipo
Parâmetro C	Numérico
Função	Função

Valor mínimo	Valor máximo	Valor inicial	Incremento	Decremento
0,1	100	1	x 10	/ 10

Função	Definição	Ordem execução
1. Linear	...	1
2. Polinomial	...	2
3. RBF	...	3

Tabela 23 - Configurações de execução do algoritmo *k*-NN

KNN	
Parâmetros	Tipo
Parâmetro k	Numérico
Distância	Função

Valor mínimo	Valor máximo	Valor inicial	Incremento	Decremento
1	15	5	+2	-2

Função	Definição	Ordem execução
1. Euclidiana	...	1
2. Cosseno	...	2
3. Jaccard	...	3
4. Manhattan	...	4

Há disponível para treinamento dos classificadores seis mil subconjuntos de dados, conforme ilustrado na Figura 43. Encontrar o melhor subconjunto para um determinado classificador é desejável, pois além da melhoria da eficácia do classificador, maior facilidade de compreensão e visualização das características representativas dos documentos é obtida.

Figura 43 - Subconjuntos disponíveis

Escolher a melhor representação de características pode ser visualizado como um problema de busca. A Figura 44 mostra um diagrama de estados com quatro características (CHAGAS, 2009). Cada estado determina um subconjunto de características escolhido em um determinado instante. O círculo branco indica a ausência de uma determinada característica, enquanto que o círculo preto indica a presença. No contexto do *framework*, cada coluna representa uma etapa do processo de MT em que se faz uso, ou não, das técnicas disponíveis para a etapa.

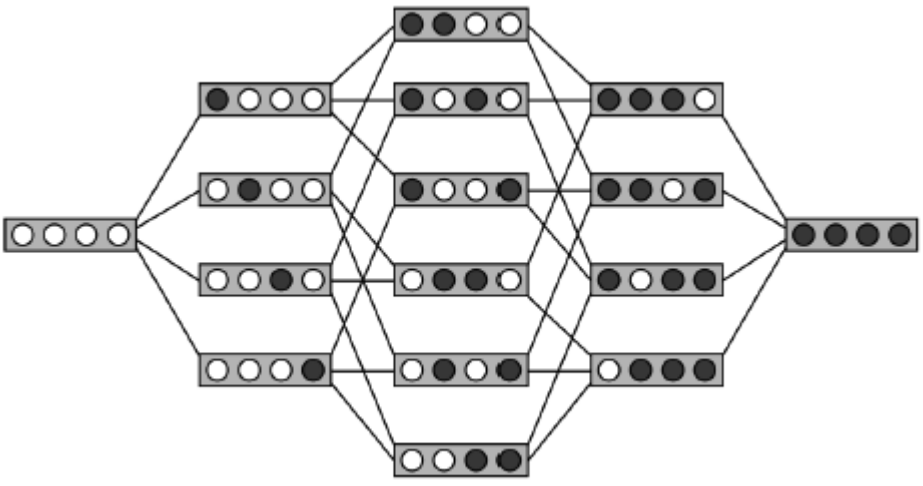


Figura 44 - Diagrama de estados

A busca completa ou exaustiva possibilita avaliar subconjuntos ótimos. Para isso, todos os subconjuntos de características possíveis são avaliados. Entretanto, em muitos casos, o espaço de busca é grande demais para ser explorado exaustivamente tornando esse algoritmo computacionalmente intratável (essa solução é NP-Completa). Nessas situações, algoritmos subótimos devem ser utilizados para encontrar a melhor solução possível.

A base de conhecimentos sobre Categorização de Textos construída para o *framework* dispõe de vinte e três caminhos do espaço de busca (soluções) que podem ser considerados subótimos. Ao iniciar uma Categorização de Textos, esses caminhos são avaliados. Baseado nos melhores resultados obtidos por essas soluções, a cada processo de treinamento de um novo classificador, características são acrescentadas ou substituídas até que um determinado critério de parada seja atingido. Um critério de parada pode ser, por exemplo, um valor *k* que determine quantas novas soluções alternativas serão avaliadas.

7.3.6.1. Resultados

Abaixo são apresentados os melhores resultados obtidos pelo *framework* no processo de Categorização Automática de Textos, isto é, a atribuição de cada notícia a um dos cinco cadernos do corpus. A Tabela 24 exibe os valores obtidos. Foram realizadas cinquenta e três tentativas (critério de parada *k = 30*) de encontrar a melhor representação dos documentos no espaço de busca. A melhor alternativa foi encontrada após a realização de trinta e cinco experimentos.

Tabela 24 - Configuração do melhor resultado obtido

Termos	Métrica de seleção	Número de termos	% Erro
Substantivo + Nome próprio + Adjetivo	Ganho de informação	150	5,20

O resultado obtido utiliza a combinação de termos que teve o melhor resultado em (CAMARGO, 2007) e a métrica de seleção que conseguiu o

desempenho mais alto em (CHAGAS, 2009). A solução de melhor desempenho é ilustrada na Figura 45. Houve utilização de todas as técnicas de tratamento linguístico fornecidas pelo *framework*.

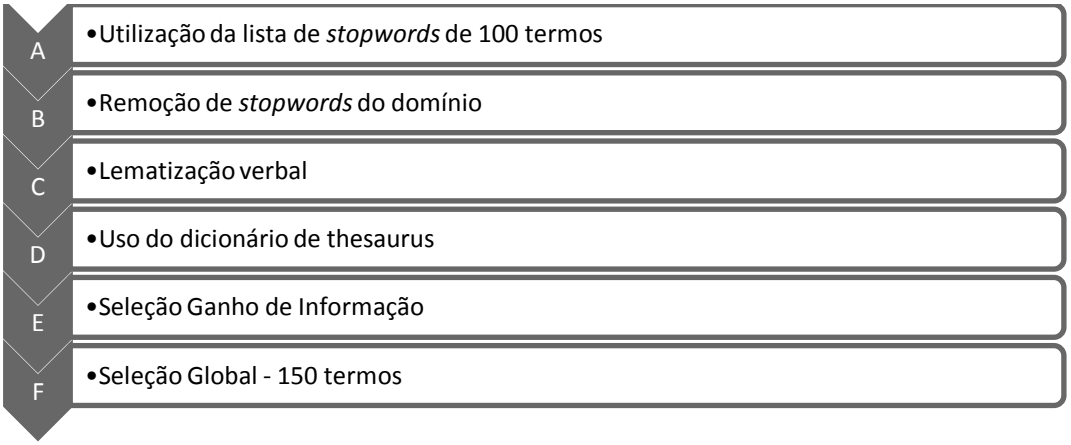


Figura 45 - Solução de melhor desempenho

Das cinco categorias, três foram categorizadas por SVM e as duas restantes pelo *k*-NN. A Tabela 25 exibe a relação "classificador x categoria".

Tabela 25 - Relação categoria x classificador

Categoria	Classificador
Esportes	SVM
Imóveis	SVM
Informática	KNN
Política	KNN
Turismo	SVM

Para as duas categorias que foram categorizadas pelo *k*-NN, obteve-se o valor de *k = 15*. Esse era o valor máximo definido para *k*; talvez, seja necessário ampliar a faixa de valores de *k*.

Mudando apenas o técnica de lematização verbal, verifica-se que a lematização verbal obteve o melhor resultado seguida pela lematização RSLP, conforme ilustrado na Figura 46.

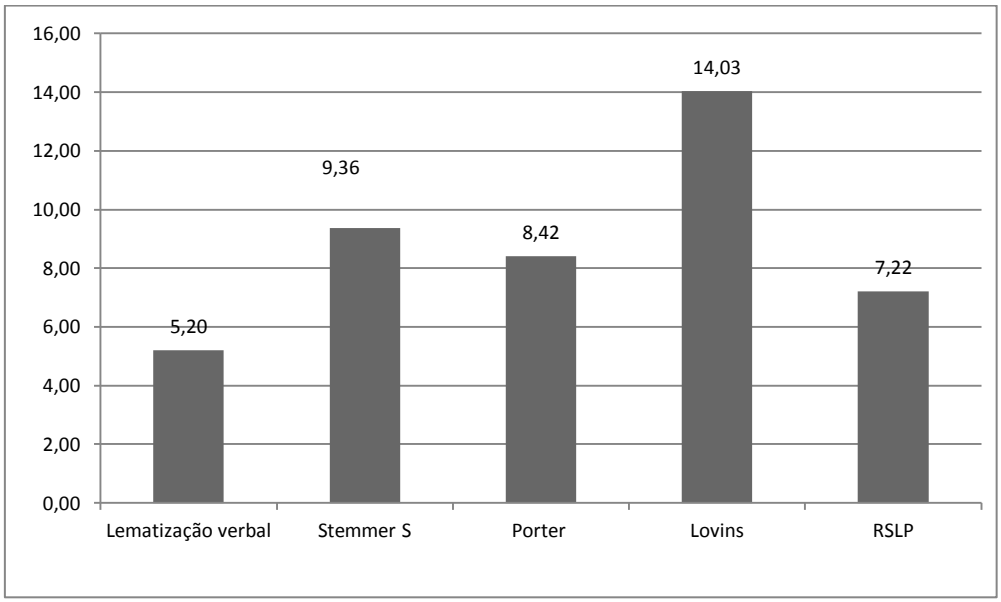


Figura 46 - Desempenho lematização

A utilização da lista de *stopwords* de 100, 200 ou 300 termos não exerceu grande influência nos resultados obtidos quando é acompanhada da remoção de *stopwords* do domínio. A consulta ao dicionário de Thesaurus incrementou o resultado de todos os experimentos quando passou a ser utilizado.

O resultado atingido (5,20%) supera o obtido em (CAMARGO, 2007): 7,49%.

8
Conclusões e Trabalhos Futuros

No presente capítulo são apresentadas as conclusões obtidas e sugestões para trabalhos futuros.

8.1.
Conclusões

Para o desenvolvimento deste trabalho foi buscada uma fundamentação teórica sobre a MT e CT enfatizando os procedimentos e características comuns aos classificadores e métodos de categorização utilizados nesta Tese. Foi apresentada uma visão geral sobre a Categorização de Textos, destacando conceitos e características de Aprendizagem de Máquina. Na abordagem adotada, os classificadores fazem uso de uma coleção de textos previamente categorizados para construir um modelo estatístico de predição capaz de categorizar novos documentos.

A combinação de classificadores e o emprego de tratamento linguístico comprovam que esta abordagem obtém ótimos resultados.

Os trabalhos correlatos também contribuíram para a execução dessa dissertação. A escolha dos trabalhos teve um impacto na aplicação de ideias e conceitos.

Na elaboração da metodologia, foram apresentadas a coleção, as categorias, a representação dos documentos, a heurística proposta e a avaliação; enfim, todo o planejamento da metodologia que foi utilizada nos experimentos. Essa metodologia é crucial para a realização, descrição e análise dos experimentos, permitindo identificar e discutir os problemas encontrados.

Entre as principais contribuições inovadoras proporcionadas, direta ou indiretamente, por esta tese podem ser destacadas:

- Utilização de técnicas de processamento linguístico em favor da obtenção de resultados de maior qualidade.
- Elaboração de heurísticas que auxiliam os processos de Mineração de Textos, como não utilizar os algoritmos de lematização em verbos.
- Assistência ao processo de descoberta de conhecimento permitindo que usuários com pouco conhecimento nas áreas de Mineração de Textos ou Categorização de Textos possam utilizar a ferramenta com êxito.
- A preparação para utilização de um Léxico verbal com termos do Português Brasileiro contendo informações sobre verbos, suas flexões e formas canônicas.
- Comprovação de que a criação de léxicos auxilia a obtenção de termos com maior valor semântico.
- A construção de um Léxico de sinônimos e antônimos com termos do Português Brasileiro.
- Um ambiente de software para a realização de Categorização Automática de Textos em Português do Brasil baseada em Mineração de Textos.

Conclui-se que o trabalho atingiu o fim a que se propôs e apresentou resultados satisfatórios quando consideradas as obras existentes na literatura.

As ferramentas utilizadas para a construção do *framework* assim como a abordagem empregada na realização dos estudos de casos foram descritas e fundamentadas, de maneira a permitir que outros pesquisadores possam vir a agregar novos algoritmos ao sistema, com um mínimo de esforço.

8.2. Trabalhos Futuros

Como trabalhos futuros, este trabalho sugere:

- A inclusão de outros algoritmos de classificação ao *framework*.

- Maior estudo sobre a combinação de classificadores também é desejável.
- Expansão do léxico criado seria de grande utilidade para muitos trabalhos futuros.
- Utilização de processamento distribuído ou paralelo, principalmente para aplicações de tempo real.

Outro ponto interessante é avaliar o *framework* em um domínio diferente, exigindo a necessidade de etiquetar uma coleção de documentos para esse fim.

Referências Bibliográficas

ALPAYDIN, E. (2004). *Introduction to Machine Learning (Adaptive Computation and Machine Learning).* The MIT Press.

AMITAY, E. (1998). Using Common Hypertext Links to Identify the Best Phrasal Description of Target Web Documents. *In Proceedings of the ACM Special Interest Group on information Retrieval.*

ARANHA, C. N. (2007). *Uma abordagem de pré-processamento automático para mineração de textos em português: sob o enfoque da inteligência computacional, Tese de Doutorado, Departamento de Engenharia Elétrica, PUC-Rio.*

ARANHA, C. N., & PASSOS, E. P. (2006). A Tecnologia de Mineração de Textos. *Revista Eletrônica de Sistemas de Informação , 2,* 2.

ARDÖ, A. (2005). Focused Crawling in the ALVIS Semantic Search Engine. *In Proceedings of 2nd European Semantic Web Conference (ESWC).* Heraklion, Greece.

BAEZA-YATES, R., & BERTIER, R. N. (1999). *Modern Information Retrieval.* Harlow: Addison-Wesley.

BARROS, C. D. (2010). *Antonímia nos adjetivos descritivos do português do Brasil: uma proposta de análise e representação. 2010. 89 f. Dissertação (Mestrado em Linguística) – Universidade Federal de São Carlos, São Carlos, 2010. .*

BASTOS, V. M. (2006). *Ambiente de Descoberta de Conhecimento na Web para a Língua Portuguesa, Tese de Doutorado, Departamento de Engenharia Civil, UFRJ.* Rio de Janeiro.

BENNETT, P. N., DUMAIS, S. T., & HORVITZ, E. (2005). *The combination of text classifiers using reliability indicators. Information Retrieval, Kluwer Academic Publishers. v. 8, n. 1, p. 67-100.* MA, USA.

BERGMARK, D., LAGOZE, C., & SBITYAKOS, A. (2000). Focused Crawls, Tunneling, and Digital Libraries. *In Proceedings of the 6th European Conference on Digital Libraries.*

BICK, E. (2000). *The parsing System PALAVRAS: Automatic Grammatical Analysis of Portuguese in a Constraint Grammar Framework.* Denmark: Arhus University Press.

BISHOP, C. M. (2007). *Pattern Recognition and Machine Learning.* Springer.

BORKO, H., & BERNICK, M. (1963). *Automatic document classification. J.Assoc.Comput.Mach.10, 2, 151–161.*

BRILL, E. (1995). *Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging.* Computacional Linguistics.

BRIN, S., & PAGE, L. (1998). The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems , Vol. 30*.

BUCKLAND, M., & GEY, F. (1944). The relationship between Recall and Precision. *Journal of the American Society for Information , Vol. 45*, 12-19.

BURGES, C. J. (1998). A tutorial on Support Vector Machines for pattern recognition. *Data Mining and Knowledge Discovery Conference, Vol. 2.*

CAMARGO, Y. B. (2007). *Abordagem Linguística na Classificação Automática de Textos em Protuguês. Dissertação de Mestrado. Departamento de Engenharia Elétrica, UFRJ.*

CARDOSO, O. N. (2000). Recuperação de Informação. Departamento de Ciência da Computação, Universidade Federal de Lavras.

CARRILHO, J. (2007). *Desenvolvimento de uma Metodologia para Mineração de Textos, Dissertação de Mestrado, Departamento de Engenharia Elétrica, PUC-Rio.*

CASTILLO, C. (2004). *Effective Web Crawling. Ph.D. Thesis, Dept. of Computer Science, University of Chile.* Santiago, Chile.

CEGALLA, D. P. (2005). *Novíssima Gramática da Língua Portuguesa* (46 ed.). IBEP.

CHAGAS, F. (2009). *Variações do Método kNN e suas Aplicações na Classificação Automática de Textos. Dissertação de Mestrado, Instituto de Informática, UFG.*

CHAKRABARTI, S. (2003). *Mining the web: Discovering knowledge from hypertext data.* San Francisco, CA: Morgan Kaufmann Publishers.

CHAVES, A. C. (2006). *Extração de regras fuzzy para máquinas de vetores suporte (SVM) para classificação em múltiplas classes.* Tese de Doutorado. Departamento de Engenharia Elétrica. PUC-Rio.

COTHEY, V. (2004). Web-crawling reliability. *Journal of the American Society for Information Science and Technology , Vol. 55*, 1228-1238.

DATE, C. J. (2005). *Introdução a Sistemas de Bancos de Dados* (8 ed.). São Paulo: Campus.

DAVISON, B. D. (2000). Topical Locality in the Web. *In Proceedings of the 23th Annual international Conference on Research and Development in Information Retieval.*

DILIGENT, M., COETZEE, F. M., LAURENCE, S., GILES, C. L., & GORI, M. (2000). Focused crawlers using context graph. *Proceedings of the 26th International Conference on Very Large Databases (VLDB)*, (pp. 527-534). Cairo, Egypt.

DOM, B., CHAKRABARTI, S., & BERG, M. (1999). Focused crawling: a new approach to topic-specific Web resource discovery. *Computer Networks , Vol. 31*, 1623-1640.

FELDMAN, R., & SANGER, J. (2007). *The Text Mining Handbook – Advanced Approaches in Analyzing Unstructured Data.* Cambridge University Press.

FERRO, M., & LEE, H. D. (2001). O Processo de KDD – Knowledge Discovery in Database para Aplicações na Medicina. *SEMINC - Semana de Informática de Cascavel, Universidade Estadual do Oeste do Paraná.* Cascável.

FINATTO, M. J. (2005). *Análise Textual Assistida por computador: Reconhecimento Lingüístico-Terminológico do texto técnico-científico de Química em Português - Da coesão à Enunciação (TextQuim), Relatório Final de Atividades, CNPQ, Instituto de Letras, UFRGS.* Porto Alegre.

FONSECA, B. M., & REIS, D. C. (2002). *O fantástico mundo da distância de edição.*

FONSECA, F., & FIDALGO, R. (2002). *Gerenciamento de Dados e Informação, Centro de Informática, Universidade Federal de Pernambuco.*

FOUCAULT, M. (2002). *A ordem do discurso* (8 ed.). São Paulo: Loyola.

FREUND, Y., & SCHAPIRE, R. E. (1999). *A short introduction to boosting. In Journal of Japanese Society for Artificial Intelligence, v.5, n.14, p.771-780.*

FUNREDES. (2007). *Línguas e Culturas na Web.* Paris, França: Terminologia e Indústrias da Língua - DTIL.

GDS PUBLISHING. (2008). Managing the Data Explosion. *Business Management* .

GIBSON, D., KLEINBERG, J., & RAGHAVAN, P. (1998). Inferring Web Communities from Link Topology. *In Proceedings of the 9th ACM Conference on Hypertxt and Hypermedia.*

GOLDSCHMIDT, R., & PASSOS, E. (2005). *Data Mining: um guia prático.* Rio de Janeiro: Campus.

GOMES, R. M. (2008). *Mineração de Textos na Desambiguação, Dissertação de Mestrado.* Rio de Janeiro: Departamento de Engenharia Elétrica, PUC-Rio.

GONÇALVES, T., SILVA, C., QUARESMA, P., & VIEIRA, R. (2006). Analyzing Part-of-Speech for Portuguese Text Classification. *Proceedings of Computational Linguistics and Intelligent Text Processing (CICLing)*, (pp. 551-562). Mexico City, Mexico.

HALL, M., FRANK, E., HOLMES, G., FAHRINGER, B., REUTEMANN, P., & WITTEN, I. H. (2009). *The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1.*

HARPER, M. P., & THEDE, S. M. (1999). A Second-Order Hidden Markov Model for Part-of-Speech Tagging. *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics,* (pp. 175-182). College Park.

HARTIGAN, J. A., & WONG, M. A. (1979). A k-means clustering algorithm. *Applied Statistics* , 100-108.

HEARST, M. (1999). Untangling Text Data Mining. *In proceedings of ACL'99: the 37th Annual Meeting of the Association for Computational Linguistics, University of Maryland.*

HEATON, J. (2002). *Programming Spiders, Bots, and Aggregators in Java.* Sybex.

HERLOCKER, J., KONSTAN, J., TERVEEN, L., & RIEDL, J. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems* , 5–53.

IBM. (13 de Fevereiro de 2008). *Unstructured Information – The Knowledge Rush.* Acesso em 03 de Maio de 2008, disponível em The Knowledge Rush: http://domino.research.ibm.com/comm/research_projects.nsf/pages/uima.knowledgeRush.html

INSITE. (2001). *Grupo de Lingüística da Insite Processamento de Linguagem Natural*. Acesso em 01 de 04 de 2008, disponível em Processamento de Linguagem Natural: http://linguistica.insite.com.br/nlp.phtml

JARGAS, A. M. (2006). *Expressões Regulares - Uma Abordagem Divertida.* Novatec.

JOACHIMS, T. (1998). *Text Categorization with Support Vector Machines: Learning with Many Relevant Features.*

JOHNSON, R., & FOOTE, B. (1988). *Designing reusable classes. Journal of Object- Oriented Programming v. 1, n. 5, pp. 22-35, Jun 1988.*

KANTARDZIC, M. (2002). *Data Mining: Concepts, Models, Methods, and Algorithms.* Wiley-IEEE Press.

KIM, S.-B., HAN, K.-S., RIM, H.-C., & MYAENG, S. H. (2006). Some Effective Techniques for Naive Bayes Text Classification. *IEEE Transactions on Knowledge and Data Engineering, Vol. 18*, pp. 1457-1466.

KLEINBERG, J. M. (1998). Authoritative Sources in a Hyperlinked Environment. . *In Proceedings of the ACM-SIAM Symposium on Discrete Algorithms.*

KONCHADY, M. (2006). *Text Mining Application Programming* (1 ed.). Charles River Media.

KUDO, T., & MATSUMOTO, Y. (2004). A Boosting Algorithm for Classification of Semi-Structured Text. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).* Barcelona, Spain.

LAUDON, K. C., & LAUDON, J. P. (2002). *Management Information Systems: managing the digital firm* (7ª ed.). New Jersey: Prentice-Hal.

LEE, Y. K., NG, H. T., & CHIA, T. K. (2004). *Supervised Word Sense Disambiguation with SVM and Multiple Knowledge Sources. In Internation*

Workshop on the Evaluations of Systems for the Semantic Analysis of Text. Barcelona.

LINDEN, G. S. (2008). *Combinação de classificadores na categorização de textos. Dissertaçaõ de Mestrado. Faculdade de Informática. PUC-RS.*

LINGUATECA. (19 de 07 de 2007). *Atomização e separação de frases.* Acesso em 07 de 03 de 2008, disponível em Projecto AC/DC: Linguateca: http://acdc.linguateca.pt/acesso/atomizacao.html

LOPES, M. C. (2004). *Mineração de Dados Textuais utilizando técnicas de Clustering para o idioma Português. Tese de Doutorado. Departamento de Engenharia Civil. UFRJ*. Rio de Janeiro.

LOVINS, J. B. (1968). Development of a stemming algorithm. *Mechanical Translation and Computacional Linguistics , Vol. 11*, 22-31.

MANDEL, A., SIMON, I., & DELYRA, J. (1997). Informação: computação e comunicação. *Revista da USP , 35*, 11-45.

MANNING, C. D., RAGHAVAN, P., & SCHÜTZE, H. (2007). *Introduction to Information Retrieval.* Cambridge University Press.

MARKOV, Z., & LAROSE, D. T. (2007). *Data Mining the Web: Uncovering patterns in Web content, structure, and usage.* New Jersey: Wiley-Interscience.

MCCALLUM, A., & NIGAM, K. (1998). A Comparison of Event Models for Naive Bayes Text Classification. *ICML-Workshop on Learning for Text Categorization.* AAAI Press.

MCCALLUM, A., & NIGAM, K. (1998). A Comparison of Event Models for Naive Bayes Text Classification. *AAAI/ICML-98 Workshop on Learning for Text Categorization.* AAAI Press.

MCCALLUM, A., NIGAN, K., RENNIE, J., & SEYMORE, K. (1999). Building Domain-Specific Search Engines With Machine Learning Techniques. *In AAAI Spring Symposium on Inteligent Agents in Cyberspace.*

MELO, L. B. (2007). *Reconhecimento de padrões textuais para Categorização Automática de Documentos. Dissertação de Mestrado. Departamento de Engenharia Elétrica, UFRJ.*

MLAENIC, D., & GROBELNIK, M. (1998). *Feature Selection for Classification Based on Text Hierarchy. In Working Notes of Learning from Text*

and the Web, Conf. Automated Learning and Discovery (CONALD-98). Pittsburgh.

MODESTO, M., PEREIRA, A. R., ZIVIANI, N., CASTILLHO, C., & BAEZA-YATES, R. (2005). Um novo retrato da Web brasileira. *Proceedings of XXXII SEMISH*, (pp. 2005-2017). São Paulo.

MONTEIRO, L., GOMES, I., & OLIVEIRA, T. (2006). Etapas do Processo de Mineração de Textos – uma abordagem aplicada a textos em Português do Brasil. *Anais do XXVI Congresso da SBC, I Workshop de Computação e Aplicações*, (pp. 78-81). Campo Grande, MS.

NAJORK, M., & WIENER, J. L. (2001). Breadth-first search crawling yields high-quality pages. *In Proceedings of the 10th International World Wide Web Conference*. Hong-Kong.

NEVES, P. I. (2012). *Um estratégia para apoiar a decisão baseada em Mineração de Textos Livres. Dissertação de Mestrado. Departamento de Ciência e Tecnologia, Instituto Militar de Engenharia.*

NG, H. T., GOH, W. B., & LOW, K. L. (1997). Feature Selection, Perception Learning and a Usability Case Study for Text Categorization. *Proceedings of SIGIR-97, 20th ACM International COnference on Research and Development in Information Retrieval.* Philadelphia, PA, USA.

NISBET, R., ELDER, J., & MINER, G. (2009). *Handbook of Statical Analysis & Data Mining Applications.* California: Elsevier.

ORENGO, V. M., & HUYCK, C. R. (2001). *A Stemming Algorithm for the Portuguese Language, in 8th International Symposium on String Processing and Information Retrieval (SPIRE). 2001: Laguna de San Raphael, Chile. p. 183-193.*

ORENGO, V., & HUYCK, C. (2001). A Stemming Algorithm for the Portuguese Language. *In Proceedings of the SPIRE Conference.* Laguna de San Raphael, Chile.

PEIXOTO, M. D., BATISTA, M. D., & CAPELO, M. J. (s.d.). Categorizção de Textos. *Departamento de Informática, Universidade da Beira Interior* .

PINKER, S. (1998). *Como a mente funciona.* São Paulo: Companhia das Letras.

PORTER, M. (1980). An algorithm for suffixing stripping. *Program: electronic library and information systems , Vol. 14* (3), 130-137.

POWEL, G. (2007). *Beggining XML Databases.* Wiley Publishing.

REIS, C. C. (2011). *Carla Corrêa Tavares dos Reis Julho. Tese de Doutorado. Departamento de Engenharia de Sistemas e Computação, UFRJ.*

REZENDE, S. O. (2005). *Sistemas inteligentes: fundamentos e aplicações.* Barueri, SP: Manoele.

RICOTTA, F. C. (2007). *Como os search engines funcionam? Projeto Final de Graduação, Departamento de Matemática e Computação, Universidade Federal de Itajubá.* Itajubá, MG.

RIJSBERGEN, C. J. (1979). *Information Retrieval.* London: University of Glasgow.

RINO, L. H., & PARDO, T. A. (2003). A Sumarização Automática de Textos: Principais Características e Metodologias. NILC – Núcleo Interinstitucional de Lingüística Computacional.

RUSSELL, N., & NORVIG, P. (2004). *Inteligência Artificial* (2 ed.). Rio de Janeiro: Elsevier.

SALTON, G., & BUCKLEY, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing & Management , Vol. 24*, 513–523.

SEBASTIANI, F. (2002). *Machine Learning in Automated Text Categorization.* Pisa, Italy.

SETIONO, R., & LEOW, W. K. (1998). *FERNN: An Algorithm for Fast Extraction of Rules from Neural Networks. National University of Singapure.*

SEYMORE, K., MCCALLUM, A., & ROSENFELD, R. (1999). Learning Hidden Markov Model Structure for Information Extraction. *AAAI 99 Workshop on Machine Learning for Information Extraction.*

SHKAPENYUK, V., & SUEL, T. (2002). Design and implementation of a high performance distributed web crawler. *In Proceedings of the 18th International Conference on Data Engineering (ICDE)* (pp. 357-368). San Jose, California: IEEE CS Press.

SHOLOM, M. W., INDURKHYA, N., ZHANG, T., & DAMERAU, F. J. (2005). *Text Mining – Predictive Methods for Analyzing Unstructured Information.* Springer.

SILVA, A. A. (2007). *Aîuri: Um portal para Mineração de Textos integrado a Grids, Dissertação de Mestrado, Engenharia Civil, UFRJ.*

SILVA, F. R. (2007). *GEODISCOVER – Mecanismo de busca especializado em Dados Geográficos. Tese de Doutorado, Departamento de Computação Aplicada, INPE*. São José dos Campos.

SINGH, H. S. (2001). *DATA WAREHOUSE: Conceitos, tecnologias, implementação e gerenciamento.* São Paulo: Makron Books.

SOARES, F. A. (2008). *Mineração de Textos na Coleta Inteligente de Dados da Web. Dissertação de Mestrado. Engenharia Elétrica. PUC-Rio.*

SPIEGEL, M. R. (2003). *Estatística.* Makron Books.

SPINK, A., WOLFRAM, D., JANSEN, M. B., & SARACEVIC, T. (2001). Searching the web: the public and their queries. *Journal of the American Society for Information Science and Technology , Vol. 52,* 226–234.

SULLIVAN, D. (Dezembro de 2000). The need for Text Mining in Business Intelligence. *DM Review .*

SUTTON, C., & MCCALLUM, A. (2006). *An introduction to conditional random fields for relational learning. In Introduction to Statistical Relational Learning.* MIT Press.

TAN, A.-H. (1999). Text Mining: The state of the art and the challenges. *PAKDD'99 Workshop on Knowledge Discovery from Advanced Databases*, (pp. 65-70). Beijing.

TAN, P.-N., STEINBACH, M., & KUMAR, V. (2005). *Introduction to Data Mining.* Pearson Addison Wesley.

TANENBAUM, A. (2003). *Redes de Computadores.* Rio de Janeiro: Campus.

TOYODA, M., & KITSUREGAWA, M. (2001). Creating a Web Community Chart for Navigation Related Communities. *In Proceedings of ACM Conference on Hypertext and Hypermedia.*

WEN, J.-R. (2006). *Search Engine Overview.* Microsoft Research Asia.

WIENER, E., PEDERSEN, L. O., & WEIGEND, A. S. (1995). Neural Network Approach to Topic Spotting. *Proceeding of the Symposium on Document Analysis and Information Retrieval*, (pp. 317-322). Las Vegas, US.

XU, Q., & ZUO, W. (2007). First-order Focused Crawling. *In Proceedings of World Wide Web Conference.*

ZHU, X., & DAVIDSON, I. (2007). *Knowledge Discovery and Data Mining: Challenges and Realities.* New York: Hershey.