Instituto de Ciências Matemáticas e de Computação

ISSN - 0103-2569

Descrição de uma metodologia de mapeamento de informações não estruturadas em uma representação atributo-valor

Daniel de Faveri Honorato
Maria Carolina Monard

Nº 317

RELATÓRIOS TÉCNICOS DO ICMC

São Carlos
Fevereiro/2008

Descrição de uma metodologia de mapeamento de informações não estruturadas em uma representação atributo-valor *

Daniel de Faveri Honorato
Maria Carolina Monard

Universidade de São Paulo
Instituto de Ciências Matemáticas e de Computação
Departamento de Ciências de Computação e Estatística
Laboratório de Inteligência Computacional
Caixa Postal 668, 13560-970 - São Carlos, SP, Brasil
e-mail: {dfaverih@gmail.com, mcmonard@icmc.usp.br}

Resumo

O processo de Mineração de Textos auxilia na extração de padrões de conjunto de documentos de textos. Freqüentemente, esse processo requer que os dados encontrem-se armazenados em uma tabela atributo-valor. Neste trabalho é apresentado uma descrição detalhada de uma metodologia desenvolvida para auxiliar no mapeamento de informações não estruturadas encontradas em um conjunto de documentos textuais nos quais o vocabulário utilizado é bastante controlado, tais como em laudos médicos, em informações estruturadas representadas em uma tabela atributo-valor. Para auxiliar na construção dos atributos dessa tabela, é utilizada uma abordagem híbrida de extração de terminologia para identificar unidades terminológicas no conjunto de documentos processados.

Palavras-Chave: Pré-processamento de textos, Aprendizado de Máquina, Mineração de Textos.

*Trabalho desenvolvido com apoio do CNPq e da Fundação Parque Tecnológico Itaipu - FPTI.

Sumário

Sumário 4
Lista de Figuras 6
Lista de Tabelas 7
Lista de Algoritmos 8
Lista de Abreviaturas 9

1 Introdução **10**

2 Extração de Terminologia **18**
2.1 Considerações Iniciais 18
2.2 Extração de Terminologia 18
2.3 Considerações Finais 21

3 Ferramentas Auxiliares **22**
3.1 Considerações Iniciais 22
3.2 Toquenizador 22
3.3 Segmentador 23
3.4 Substituições 24
3.5 Etiquetador MXPOST 25
3.6 *N-gram Statistics Package* 26
3.7 Lematizador 28
3.8 *Graph Visualization Software* 29
3.8.1 Dot 29
3.8.2 Neato 30
3.8.3 Twopi 31
3.9 Estrutura do Dicionário 32
3.10 Considerações Finais 33

4 Metodologia Desenvolvida **34**
4.1 Considerações Iniciais 34
4.2 Pré-processamento 35
4.2.1 Preparação do *Corpus* 35
4.2.2 Construção do Conjunto de Frases Únicas 37
4.2.3 Remoção de *Stopwords* 38
4.2.4 Transformação para Minúsculo 39
4.2.5 Correção Ortográfica 40
4.2.6 Normalização 42
4.2.6.1 Aplicação de Substituições 43
4.2.6.2 Aplicação de Lematização 46
4.3 Extração de Terminologia 46

4.3.1 Etiquetação dos Documentos 48
4.3.2 Definição de Máscaras 50
4.3.3 Extração de Termos dos Documentos 51
4.3.4 Aplicação de Heurísticas 53
4.4 Identificação de Atributos 57
4.4.1 Definição de Termos Raíz 58
4.4.2 Geração de Árvores 58
4.4.3 Definição de Atributos 60
4.5 Construção do Dicionário 63
4.6 Construção da Tabela Atributo-valor 63
4.7 Considerações Finais 65

Referências **68**

Lista de Figuras

1.1 Tipos de estruturação de documentos 11
1.2 Tipos de laudos 15
1.3 Laudo estruturado de Andrologia 15
1.4 Laudo semi-estruturado de Endoscopia Digestiva Alta 16
1.5 Laudo estruturado e semi-estruturado de Colonoscopia 17

3.1 Gráfico hierárquico gerado utilizando o construtor dot 30
3.2 Gráfico não direcionado gerado utilizando o construtor neato . . 31
3.3 Gráfico circular gerado utilizando o construtor twopi 31
3.4 Exemplo de árvore trie 32
3.5 Exemplo de árvore trie após inserção da palavra BONITEZA . . . 33

4.1 Metodologia desenvolvida 34
4.2 Divisão dos segmentos do laudo de Endoscopia Digestiva Alta . 36
4.3 Exemplo de processamento dos segmentos do laudo 36
4.4 Exemplo de geração do Conjunto de Frases Únicas — CFU . . . 37
4.5 Exemplo do arquivo do Conjunto de Frases Únicas 38
4.6 Exemplo do arquivo de estatística do CFU 39
4.7 Exemplo do arquivo de *stopwords* 40
4.8 Exemplo do arquivo de correção de termos incorretos 42
4.9 Exemplo do arquivo de substituição 44
4.10 Processo de aplicação das substituições 45
4.11 Metodologia para extração de termos 48
4.12 Geração e análise do CFU etiquetado 48
4.13 Aplicação do algoritmo de correção de etiquetas 49
4.14 Exemplo do arquivo de correção de etiquetas 49
4.15 Processo de etiquetação do conjunto de documentos 50
4.16 Exemplos de frases extraídas de um laudo 51
4.17 Exemplos de frases etiquetadas extraídas de um laudo 51
4.18 Processo de extração de termos 52
4.19 Termo raíz da árvore 59
4.20 Filho do termo raíz 59
4.21 Filho do termo mucoso 60
4.22 Arquivo correspondente a árvore gerada a partir dos termos `lago` e `cardia` 61
4.23 Árvore gerada a partir do termo `lago` 62
4.24 Árvore gerada a partir do termo `cardia` 62
4.25 Arquivo com os comandos do GRAPHVIZ gerado a partir da descrição XML do termo `lago` 62

4.26 Árvore gerada a partir do termo `mucosa fundo` 63
4.27 Exemplo da tabela atributo-valor 64

Lista de Tabelas

1.1 Representação de documentos 10

2.1 Fragmento de uma lista de 2-gramas 20
2.2 Fragmento de uma lista de termos extraídos usando a abordagem lingüística 20
2.3 Fragmento de uma lista de termos extraídos usando a abordagem híbrida 21

3.1 Medidas de associação do pacote NSP 27

4.1 Lista de unigramas 41
4.2 Lista auxiliar 41
4.3 Lista de unigramas após processar a palavra `aproximandamente` 42
4.4 Exemplos de substituições. 43
4.5 Tabela de máscaras padrão 51
4.6 Lista de termos gerados com máscaras `N` e `N N` 53
4.7 Lista de termos candidatos gerados após realizar a primeira etapa 56
4.8 Lista de termos gerados após realizar a segunda etapa 57
4.9 Lista de termos gerados após realizar a terceira etapa 57
4.10 Lista de termos final 58
4.11 Lista de n-gramas. 59

Lista de Algoritmos

1 Construção do Conjunto de Frases Únicas 38
2 Aplicação de substituições 45
3 Lematização 46
4 Extração de termos 52
5 Aplicação de heurísticas - parte 1 54
6 Aplicação de heurísticas - parte 2 55
7 Filtragem de termos candidatos 56
8 Geração da tabela atributo-valor 64

Lista de Abreviaturas

AT Árvore Trie

CFU Conjunto de Frases Únicas

EDA Endoscopia Digestiva Alta

ET Extração de Terminologia

GraphViz *Graph Visualization Software*

LABI Laboratório de Bioinformática

LABIC Laboratório de Inteligência Computacional

LTC Lista de Termos Candidatos

MT Mineração de Textos

NILC Núcleo Interinstitucional de Lingüística Computacional

NSP *N-gram Statistics Package*

PLN Processamento de Língua Natural

POS *Part-of-speech*

SEACAT Sistema de Extração Automática de Candidatos a Termos

Capítulo 1

Introdução

O desenvolvimento e utilização de tecnologias para a aquisição e armazenamento de dados, nas mais diversas áreas do conhecimento, têm permitido o acúmulo de dados em uma velocidade maior que a capacidade humana possui para processá-los. Esses dados podem estar representados em diferentes formatos, sendo que um dos formatos bastante utilizados é o formato textual não estruturado. Para que esses dados textuais brutos possam tornar-se úteis, é necessário que eles sejam representados de maneira apropriada para a extração de padrões tal que um modelo que represente o conhecimento embutido nesses dados seja construído. Uma das maneiras de alcançar esse objetivo é por meio da realização do processo de Mineração de Textos — MT — (Weiss et al., 2005).

As tarefas de MT podem ser divididas em quatro fases: coleta de documentos, pré-processamento, extração de conhecimento e pós-processamento. Na fase de coleta de documentos são recuperados os documentos relevantes ao domínio da aplicação. A fase de pré-processamento é responsável por transformar os documentos para um formato adequado para serem submetidos aos algoritmos de extração de padrões. Uma das transfomações mais utilizadas para transformar textos em tabelas é a abordagem *bag-of-words*, a qual pode utilizar o formato de uma tabela atributo-valor, como representada na Tabela 1.1. Cada documento d_i é um exemplo da tabela e cada termo (palavra) t_j é um elemento do conjunto de atributos. Cada documento também pode ter associado à classe a qual pertence (C).

	t_1	t_2	$\dots$	t_M	C
d_1	a_{11}	a_{12}	$\dots$	a_{1M}	c_1
d_2	a_{21}	a_{22}	$\dots$	a_{2M}	c_2
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
d_N	a_{N1}	a_{N2}	$\dots$	a_{NM}	c_N

Tabela 1.1: Representação de documentos

Mais especificamente, na Tabela 1.1 estão representados N documentos (exemplos) compostos por M termos (atributos). Cada documento d_i é um vetor $d_i = (a_{i1}, a_{i2}, ...a_{iM})$, no qual o valor a_{ij} refere-se ao valor associado ao j-ésimo termo do documento i. O valor a_{ij}, do termo t_j no documento d_i, pode ser calculado utilizando diferentes medidas, tais como *boolean*, freqüência do

termo no documento (*term frequency*) e freqüência do termo no documento em relação ao número de vezes que aparece no conjunto de documentos (Sebastiani, 2002).

Na fase de extração de conhecimento o objetivo é descobrir padrões úteis e desconhecidos presentes nos documentos. Por fim, é necessária a fase de pós-processamento para verificar se o objetivo foi alcançado ou se é necessário retornar a alguma fase anterior.

Os textos, em geral, podem apresentar algum nível de estruturação na apresentação das informações, como também podem ser totalmente livres de estruturação. Para distinguir em qual nível de estruturação se encontra determinado texto, pode ser considerado como parâmetro de classificação a maneira como a informação é apresentada dentro do arquivo que contem o texto. Na Figura 1.1 é ilustrado um esquema que auxilia para um melhor entendimento dos níveis de estruturação de informações textuais (Chang et al., 2006). Conforme pode ser observado na figura, dados não estruturados são compostos por textos livres, por exemplo, artigos de notícias, os quais não possuem uma estrutura pré-definida. Os dados semi-estruturados possuem uma certa estrutura, por exemplo, registros[1] médicos, anúncios e determinadas páginas *Web*. Finalmente, os dados estruturados possuem uma estrutura pré-definida, tal como as bases de dados ou arquivos textos altamente estruturados, nos quais as informações podem ser facilmente tratadas de acordo com a estrutura previamente definida.

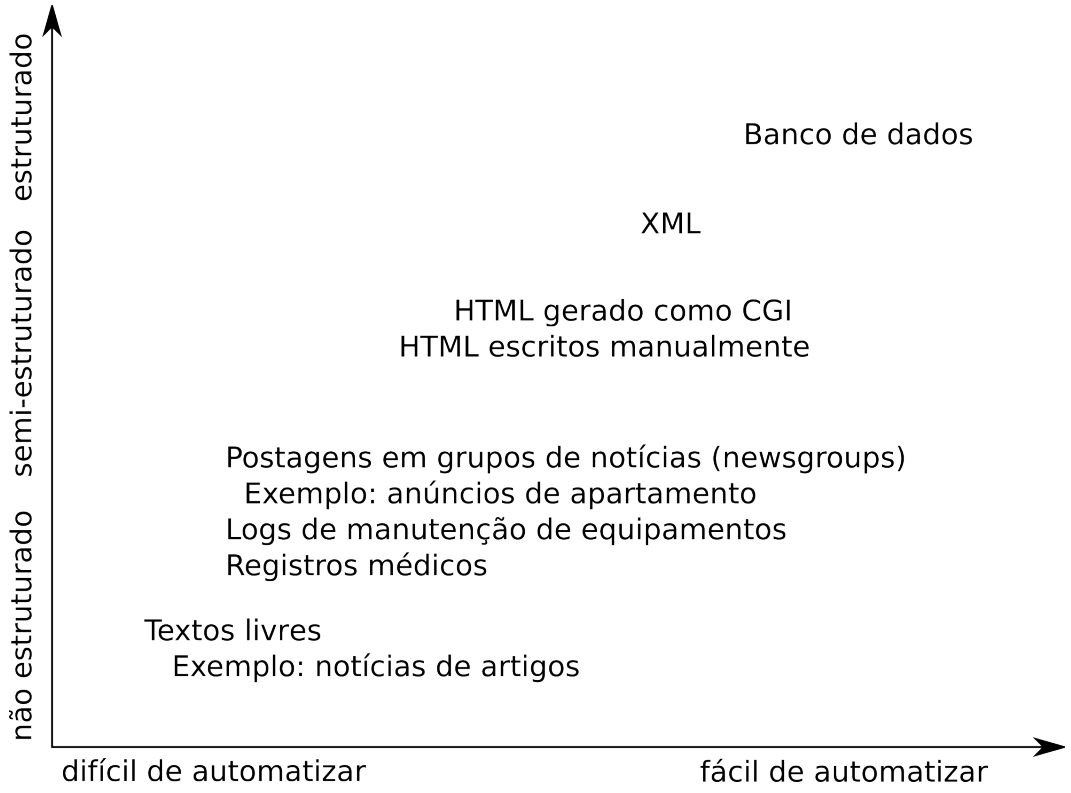

Figura 1.1: Tipos de estruturação de documentos (Chang et al., 2006)

[1]Neste trabalho estamos considerando registros, relatórios, documentos e laudos como sinônimos.

Na Figura 1.1 também é possível observar que as informações não estruturadas são as mais difíceis de serem tratadas utilizando um processo automatizável, enquanto que esse problema decresce quando incrementa a estruturação das informações.

A linguagem natural é a maneira mais comum de comunicação humana. Na área biomédica[2], conhecimento e dados são disseminados na forma escrita por meio de artigos na literatura científica, relatórios técnicos e adminstrativos bem como informações de pacientes (Cimino & Shortliffe, 2006). Cada vez mais os computadores estão sendo empregados para facilitar o processo de coleta, armazenamento e distribuição de informações biomédicas. Enquanto a capacidade para acessar as informações é altamente benéfica para pesquisadores, médicos e administradores, as informações, muitas vezes, não estão em uma forma adequada para que possam ser posteriormente processadas pelo computador.

O maior impacto da computação na medicina é visto no processamento de dados estruturados, ou seja, informações representadas em um formato regular como uma tabela atributo-valor (Cimino & Shortliffe, 2006). Desse modo, um sistema que realize transformação de informações médicas textuais em dados estruturados possibilitaria o desenvolvimento de novas aplicações na área médica que, de outra maneira, não seria possível. Em outras palavras, uma vez que as informações estão estruturadas, é possível desenvolver uma grande faixa de aplicações médicas, as quais deverão se tornar ferramentas imprescindíveis para auxiliar médicos e pesquisadores.

Para exemplificar, suponha que uma clínica médica deseja obter determinadas informações a partir de um conjunto de laudos de pacientes, tais como quais pacientes foram atendidos em determinada data e quais medicamentos foram aplicados. A clínica também poderia estar interessada na identificação de padrões que pudessem auxiliar no processo de tomada de decisões, por exemplo, pacientes com sintomas X e Y são diagnosticados com a doença Z. Nesse caso, para que possa ser realizada uma análise mais completa dos laudos, tais como geração de estatísticas, gráficos, relatórios e extração de padrões, é necessário que os mesmos sejam transformados para um formato estruturado, tal como o formato atributo-valor, para que técnicas que auxiliem nessas tarefas sejam aplicadas. Depois de transformadas as informações para o formato atributo-valor, seria possível, portanto, a aplicação de algoritmos para a descoberta de padrões embutidos nos dados, os quais dificilmente o profissional da saúde iria descobrir por meio de uma análise manual. Alguns benefícios de se aplicar algoritmos para extração de padrões são:

- Identificação de padrões implícitos, ou seja, relacionamentos nas informações fornecidas nos exames e que possibilitariam chegar a um determinado diagnóstico;

- A partir dos padrões identificados o médico poderia chegar mais rapidamente a um diagnóstico, descartando informações que não auxiliam no diagnóstico;

[2]Segundo Cimino & Shortliffe (2006), a informática biomédica engloba amplamente todas áreas da saúde, de práticas clínicas e pesquisa biomédica.

- Poderiam ser identificados agrupamentos nos dados, de modo que, com base nas informações do paciente, fosse possível dizer a qual grupo o paciente pertence.

Assim, a transformação de informações para o formato atributo-valor é de fundamental importância para a aplicação de algoritmos de extração de padrões. Vários desafios são encontrados no desenvolvimento de sistemas de transformação de informações médicas para o formato estruturado, conforme apresentados a seguir (Cimino & Shortliffe, 2006).

- **Disponibilidade de laudos e confidencialidade**: O desenvolvimento de um sistema de transformação de informações é baseado na análise de amostras de textos para serem processados. No domínio médico, isso significa que uma grande quantidade de registros de pacientes na forma digital deve estar disponível para os desenvolvedores. Um problema é que nem todos registros médicos estão no formato eletrônico. Em muitos hospitais, anotações clínicas, tais como notas escritas por enfermeiras ou mesmo progressos do paciente, são escritas em papel e não são armazenadas no formato digital. Entretanto, sistemas de transformação de informação necessitam que os registros médicos estejam disponíveis eletronicamente na forma textual de modo que possam processá-los. Outra questão é que registros de pacientes são confidenciais, e a fim de torná-los acessíveis para propósitos de pesquisa, informações de identificação pessoal devem ser removidos.

- **Expressividade**: A linguagem natural é extremamente expressiva no sentido de que há, freqüentemente, diferentes maneiras para descrever um mesmo conceito médico, bem com há numerosas maneiras para expressar modificadores de um conceito. Por exemplo, laudos associados ao câncer podem ser expressos utilizando um grande número de termos, tais como `neoplasma`, `tumor`, `lesão`, `massa` e `linfoma`. Além disso, os modificadores desses termos tornam a tarefa de extração de informação mais complexa, uma vez que também devem ser levados em consideração.

- **Formatos heterogêneos**: Não há um formato padrão para os laudos. Embora seções e subseções de laudos são importantes para muitas aplicaçãos porque elas provêm contexto, seus nomes não são padronizados. Por exemplo, `diagnóstico`, `diagnóstico final` e `diagnóstico médico` podem nomear as mesmas seções de um laudo. Outro problema ocorre porque o formato do texto nos laudos não é padronizado. Pontuações estão freqüentemente faltando ou estão inapropriadas. Por exemplo, uma nova linha pode ser utilizada para significar o final de uma sentença em vez de um ponto final. Outro problema é que alguns laudos contém tabelas com diferentes configurações.

- **Abreviação do texto**: Geralmente, laudos médicos são bastante compactos, contém abreviações e freqüentemente omitem informações que podem ser facilmente inferidas por profissionais do domínio com base nos seus conhecimentos do contexto e do domínio. Um problema com

as abreviações é que são altamente ambíguas. Por exemplo, a abreviação "pe", do inglês, pode significar *physical examination* (exame físico), *pleural effusion* (efusão pleural) ou *pulmonray embolism* (embolismo pulmonar). Outro desafio está relacionado às informações omitidas ou implícitas, porque um sistema automático que utiliza a informação estruturada gerada por meio dos laudos, deverá automaticamente capturar as informações implícitas baseadas no conhecimento do domínio. Essa é uma tarefa muito complexa. Um exemplo simples é a localização de uma parte do corpo que está freqüentemente faltando no diagnóstico. Por exemplo, quando o termo `massa` ocorre em um laudo radiológico do tórax, significa massa do pulmão, uma vez que se ocorresse em um laudo de mamografia, iria significar massa do seio.

- **Eventos raros**: Sistemas que realizam a transformação de informações de linguagem natural geralmente precisam de um grande número de exemplos (laudos) de treinamento para treinar, refinar e testar o sistema. Uma vez que alguns eventos ocorrem raramente, pode ser difícil encontrar um grande número de laudos com esses eventos. Um recurso é utilizar bases de dados terminológicas, as quais podem ser úteis para fornecer conhecimento léxico para termos médicos pouco freqüentes.

- **Erros de Ortografia**: Registros médicos as vezes contém erros de ortografia, os quais podem fazer com que um sistema de transformação de informação não identifique determinada informação. A correção automática de erros de ortografia é difícil e poderia criar novos erros. Por exemplo, a palavra `hiprtensão` irá causar um erro de interpretação. Não é fácil corrigir essa palavra automaticamente sem um conhecimento adicional, uma vez que ela pode se referir a `hipertensão` ou `hipotensão`.

- **Determinando o tipo de informações a ser capturada**: Determinar o tipo de informação que o sistema deverá capturar é uma decisão importante. O sistema pode processar parte do laudo, relacionado a determinada informação que quer se extrair, e não analisar o laudo inteiro.

- **Falta de um conjunto de padronizações do domínio**: O conhecimento do domínio que está sendo processado é importante para os sistemas de transformação de informação, uma vez que provêm informações do contexto que são freqüentemente necessárias. Por exemplo, o conhecimento do domínio facilitaria a recuperação de informações implícitas, *i.e.* a palavra `massa` em um mamograma denota massa do seio. Também facilita a resolução de uma palavra ambígua ou abreviada, por exemplo, `pvc`, do inglês, em um raio-x do tórax, denota *pulmonar vascular congestion*. Por outro lado, em um eletrocardiograma, denota *premature ventricular complexes*.

Esses são alguns dos problemas que devem ser enfrentados quando está sendo construído um sistema para processar informações textuais. Embora sejam vários os desafios, muitas vezes eles aparecem com maior intensidade em um domínio do que em outros.

Nesse contexto, os laudos médicos podem ser do tipo estruturado, semi-estruturado, não estruturado ou uma combinação desses formatos. Na Figura 1.2 é ilustrada a interação entre esses formatos de laudos.

Figura 1.2: Tipos de laudos

Os laudos estruturados consistem de informações em um formato pré-definido. Isso facilita a extração e a transformação dessas informações para o formato atributo-valor. Geralmente, técnicas simples podem ser utilizadas para esse fim, uma vez que os laudos possuem uma estrutura rígida em relação as informações que eles contêm. Na Figura 1.3 é apresentado um fragmento de laudo estruturado da área de Andrologia. Nesse laudo, a extração de uma determinada informação pode ser realizada facilmente por meio da utilização de expressões regulares.

LABORATÓRIO DE ANDROLOGIA
(resultado de exame – 008/01)

data: 17.01.01 **nome do paciente**: xxxxxxxxxxxx **idade**: 38 anos
exame solicitado: Processamento do sêmen
requisitado por: Dr.xxxxxxxxxxxxxx
diagnóstico clínico: Avaliação inicial
local da coleta: (X) sala de coleta do laboratório ( ) em casa
método de coleta: (X) masturbação ( ) outros
ejaculado: ( ) total ( ) parcial
Obs: Houve perda da última parte do ejaculado.

AVALIAÇÃO MACROSCÓPICA:
hora da coleta: 8: 30 h **processamento**: 40 minutos após a coleta
tempo de abstinência: 02 dias **volume:** 4,2 mL **(normal: > 2 mL)**
cor: (X) branco-opalescente ( ) amarelo-opalescente ( ) translúcido
odor: característico **pH:** 8,5 **(normal 7,2)**
viscosidade: ( ) normal **liquefação:** (X) completa
(X) aumentada ( ) incompleta (grumos)

Figura 1.3: Laudo estruturado de Andrologia

Os laudos não estruturados são aqueles que não exibem regularidade na apresentação das informações, ou seja, as mesmas estão descritos em língua natural. Neste caso, as informações a serem extraídas não são facilmente detectadas, a menos que se tenha algum conhecimento lingüístico sobre elas.

Por outro lado, os laudos semi-estruturados aparecem em um ponto intermediário entre laudos sem nenhuma estruturação (descritos em língua natural) e laudos completamente estruturados. Os laudos semi-estruturados são aqueles que apresentam alguma regularidade na disposição das informações, *i.e.*, algumas informações no texto podem apresentar uma regularidade, enquanto outras informações aparecem de forma livre.

Na Figura 1.4 é apresentado um exemplo de laudo semi-estruturado da área de Endoscopia Digestiva Alta — EDA. Nesse laudo, é fácil observar uma divisão nas informações relacionadas ao esôfago, ao estômago, ao duodeno e a biópsia e conclusão do exame. Nos três primeiros segmentos as informações estão descritas em língua natural. No último segmento, a informação referente a biópsia está estruturada, enquanto que a informação referente a conclusão do exame está descrita em língua natural.

```
* ESÔFAGO
- Mucosa de aspecto normal em toda a sua extensão.
- Calibre e distensibilidade normais.
- Motilidade normal.
- TEG situada ao nível do pinçamento diafragmático.
* ESTÔMAGO
- Cardia fechado à retrovisão.
- Mucosa de fundo de aspecto normal.
- Mucosa de corpo alto/médio, pequena curvatura,
com presença de cicatriz de úlcera.
- Incisura angularis normal.
- Mucosa de antro de aspecto normal.
- Motilidade normal.
- Lago mucoso claro.
- Piloro centrado, pérvio.
* DUODENO
- Bulbo amplo, sem lesões.
- Segunda (2ª.) porção normal.
*BIÓPSIA:( x )SIM   (  )NÃO
*CONCLUSÃO: ÚLCERA GÁSTRICA CICATRIZADA (S1 DE SAKITA) - 2/36.
```

Figura 1.4: Laudo semi-estruturado de Endoscopia Digestiva Alta

Na Figura 1.5 é apresentado outro exemplo de laudo médico, nesse caso da área de Colonoscopia, no qual uma parte do texto é estruturada e outra é semi-estruturada.

Muitas vezes, na linguagem médica são suprimidos alguns termos como artigos e verbos de modo a alcançar uma forma compacta na descrição das informações. No laudo da Figura 1.4 pode ser notado que o médico usou uma maneira bem compacta na descrição das informações, praticamente eliminando a utilização de verbos. Uma vez que a comunidade largamente usa e aceita essas formas alternativas, elas não são consideradas gramaticalmente incorretas, mas constituem uma sublinguagem (Cimino & Shortliffe, 2006).

Na linguagem médica existem várias sublinguagens, cada uma exibindo um conteúdo especializado e uma forma lingüística diferente. Essas sublinguagens médicas são mais fáceis de interpretar, uma vez que os padrões semânticos são mais restritos e, portanto, podem ser representados de uma maneira mais simples (Chen et al., 2005). Por exemplo, `medicação`, `doença` ou `parte do corpo` tem um pequeno número de padrões semânticos, por exemplo, `medicação-trata-doença` e `parte-do-corpo-tem-característica`.

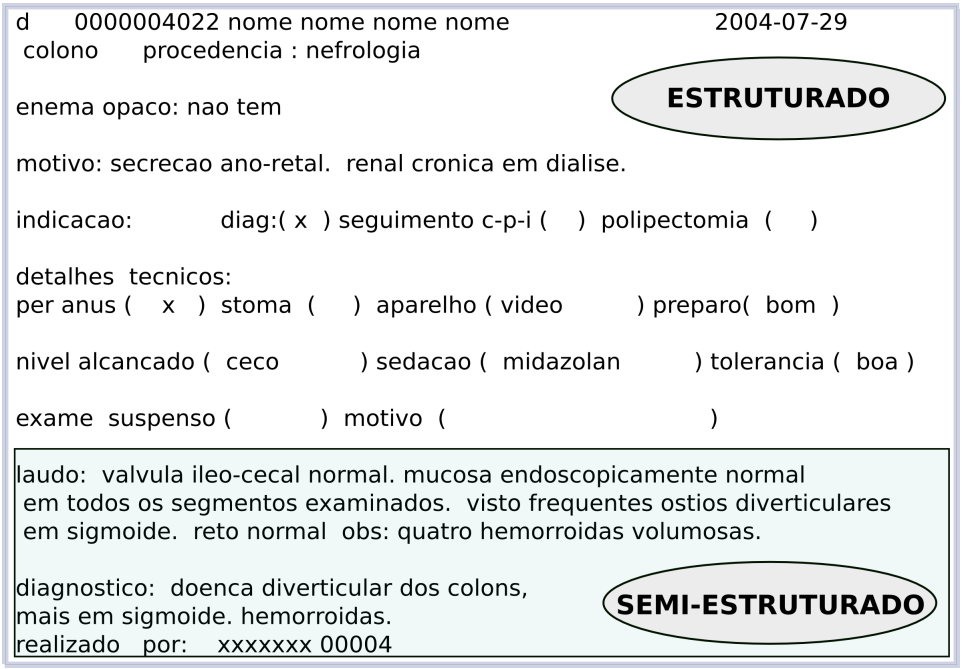

Figura 1.5: Laudo estruturado e semi-estruturado de Colonoscopia

O objetivo deste trabalho é apresentar a metodologia proposta e implementada (Honorato & Monard, 2008) para auxiliar na transformação de conjunto de documentos com informações médicas não estruturadas em uma representação atributo-valor que possa ser utilizada na aplicação do processo de mineração de texto. Essa metodologia faz parte do trabalho de mestrado de Honorato (2008). Deve ser ressaltado que a metodologia implementada neste trabalho faz parte de um projeto de pesquisa mais amplo, desenvolvido conjuntamente por pesquisadores do Laboratório de Bioinformática — LABI, da Universidade Estadual do Oeste do Paraná e o Laboratório de Inteligência Computacional — LABIC, da Universidade de São Paulo. Uma metodologia inicial foi desenvolvida no LABI (Honorato et al., 2005), na qual é necessária uma grande interação com especialista(s) do domínio. A metodologia implementada neste trabalho tem por objetivo diminuir, sempre que possível, a intervenção do(s) especialista(s) do domínio.

Esse trabalho está organizado da seguinte maneira: No Capítulo 2 são brevemente apresentados alguns conceitos da área de Extração de Terminologia utilizados no desenvolvimento deste trabalho. No Capítulo 3 são apresentadas as ferramentas computacionais utilizadas para auxiliar no desenvolvimento da metodologia proposta neste trabalho. No Capítulo 4 é apresentada em detalhes todas as fases da metodologia proposta e a implementação de cada uma dessas fases.

Capítulo 2

Extração de Terminologia

2.1 Considerações Iniciais

Na área de Extração de Terminologia — ET — podem ser identificadas diversas abordagens para extração de termos de interesse de um determinado domínio. Neste capítulo são apresentadas sucintamente algumas abordagens e soluções relacionadas com o desenvolvimento deste trabalho.

2.2 Extração de Terminologia

Segundo Pavel & Nolet (2002) a palavra terminologia significa:

> *conjunto de palavras técnicas pertencentes a uma ciência, uma arte, um autor ou um grupo social.*

Dois exemplos de terminologia seriam a terminologia da medicina, objeto de estudo deste trabalho, e a terminologia da computação. Em um sentido mais restrito e especializado, as mesmas autoras designam terminologia como:

> *disciplina lingüística consagrada ao estudo científico dos conceitos e termos usados nas línguas de especialidade.*

Nesse contexto, língua especializada é a que é utilizada para proporcionar uma comunicação sem ambiguídades numa área determinada do conhecimento ou da prática, com base em um vocabulário e o uso lingüístico específico, enquanto a língua comum é aquela utilizada no quotidiano.

Mas, o que é exatamente uma unidade lexical ou um termo? Segundo (Sager et al., 1980) *apud* (Frantzi & Ananiadou, 1997), termos são representações lingüísticas de conceitos em uma área particular do conhecimento e são caracterizados por uma referência especial. Desse modo, enquanto palavras coletivamente formam o vocabulário, os termos de um domínio formam sua terminologia. De acordo com Frantzi & Ananiadou (1997) um termo pode ser definido como:

> *Um termo T é um par ordenado <c,t>, onde c é um conceito de uma linguagem especial e t é o termo.*

Os termos estão sempre relacionados a uma linguagem específica, a qual é uma linguagem para um tipo restrito de comunicação, tais como medicina, direito e engenharia mecânica. Nesse contexto, pode-se dizer que linguagens específicas são baseadas e derivadas de uma linguagem geral.

Os termos podem ser constituídos de uma palavra, neste caso denominados termos simples, ou compostos por mais de uma palavra, denominados termos multi-palavras, termos compostos ou termos complexos. Os termos compostos têm sido considerados como sendo as unidades preferidas para designação de conceitos terminológicos. Neste trabalho usamos termo simples e composto como unidade terminológica.

Uma das áreas que busca minimizar as dificuldades de aquisição de terminologia é a área de computação. Segundo (Dubuc, 1999) *apud* (Almeida et al., 2006), a área de terminologia e de computação estão ligadas de forma a facilitar o armazenamento e a difusão de dados terminológicos e na elaboração de grandes bases de dados especializados, denominados banco de terminologias.

A área de extração de terminologia é importante em várias aplicações computacionais, tais como (Bagot et al., 1998)):

- criação de glossários, vocabulários e dicionários terminológicos;
- criação de programas de tradução automática;
- indexação de textos;
- criação de bases de conhecimento;
- criação de sistemas de hipertextos;
- criação de sistemas especialistas; e
- análise de *corpus*.

De acordo com Bagot (1999), um sistema de extração automática de candidatos a termos — SEACAT — é composto por um conjunto de programas para reconhecimento de unidades terminológicas de *corpus*[1] e seu objetivo é automatizar a fase de seleção de todas as unidades terminológicas de um texto especializado. Desse modo, o SEACAT promove a rapidez e sistematicidade ao trabalho de extração de terminologias. SEACATs são classificados de acordo com a metodologia que utilizam para reconhecer as unidades terminológicas. Tradicionalmente são distingüidos entre sistemas que utilizam (Paulo et al., 2002; Pantel & Lin, 2001):

- conhecimento estatístico;
- conhecimento lingüístico; e
- conhecimento híbrido, ou seja, conhecimento estatístico juntamente com conhecimento lingüístico.

[1]Em lingüística, um *corpus* consiste em um conjunto de textos, os quais são utilizados em análises estatísticas, checagem de ocorrências e validação de regras lingüísticas em um universo específico.

Na abordagem estatística, medidas estatísticas de associação são aplicadas para realizar a ordenação de termos extraídos de um *corpus*. Essas medidas são aplicadas sobre n-gramas[2] de palavras e o resultado é uma lista de n-gramas ranqueadas em ordem decrescente de importância, de acordo com a medida de associação utilizada, entre as quais podem ser aplicadas as medidas *Log-likelihood* (Dunning, 1993), Informação Mútua (Church & Hanks, 1989) e Dice (Smadja, 1993). Na Tabela 2.1 é ilustrado o fragmento de uma lista de 2-grama gerada a partir de um conjunto de documentos de Endoscopia Digestiva Alta, usando a medida *Log-likelihood*. Nesse caso, a medida *Log-likelihood* considera que o 2-grama `pinçamento diafragmático` tem uma importância maior do que o 2-grama `distensibilidade normal`.

Bigrama	Freqüêcia	Log-likelihood
pinçamento diafragmático	499	3929.93
calibrar distensibilidade	496	3898.03
teg situar	499	3787.35
motilidade normal	497	2100.43
distensibilidade normal	497	2100.43

Tabela 2.1: Fragmento de uma lista de 2-gramas

Na abordagem lingüística, o reconhecimento de termos pode ser realizado por meio da captura de propriedades sintáticas no *corpus* (Pazienza et al., 2005). Mais especificamente, nessa abordagem, informações lingüísticas são adicionadas no *corpus* em um processo que geralmente consiste em associar a cada palavra do *corpus* sua categoria gramatical (*part-of-speech*); por exemplo, nome (N), verbo (VERB), adjetivo (ADJ), e outros. Essas anotações lingüísticas possibilitam que sejam especificados padrões sintáticos que descrevem o tipo de unidade terminológica a ser extraída. Assim, a extração de termos envolve uma pesquisa, no *corpus*, de todas as unidades que atendam a um padrão pré-especificado, por exemplo, N, N N, N ADJ, entre outras.

Na Tabela 2.2 são apresentados alguns exemplos da extração de terminologia usando a abordagem lingüística. Na primeira coluna estão colocados os padrões sintáticos e na segunda coluna os termos que foram identificados.

Padrão Sintático	Termos
N	motilidade
N	terço
N N	nível pinçamento
N N	presença cordão
N ADJ	motilidade normal
N ADJ	erosão linear

Tabela 2.2: Fragmento de uma lista de termos extraídos usando a abordagem lingüística

Segundo Patry & Langlais (2005), sistemas de terminologia tradicionalmente combinam técnicas lingüísticas e estatísticas. Geralmente, a análise lingüística é realizada antes da aplicação das medidas estatísticas, para que sejam selecionados todos os candidatos que apresentam algum padrão

[2] n-gramas podem ser definidos como uma seqüência contígua ou não contígua de palavras que ocorrem próximas uma da outra no texto. Neste trabalho trabalhamos com n-gramas contíguos, *i.e.*, aqueles nos quais as palavras ocorrem uma do lado da outra no texto.

lingüístico sobre os quais serão realizados os testes estatísticos. Outra abordagem híbrida é realizar a extração de termos que apresentam algum padrão sintático e extrair a freqüência de cada um desses termos. Embora a freqüência não deriva de um princípio estatístico teórico, estamos levando em consideração a simples suposição de que expressões freqüentes denotam um importante conceito para o domínio que está sendo examinado e essas expressões devem assumir uma alta posição na lista de termos candidatos.

Na Tabela 2.3 é ilustrada uma lista de termos extraídos utilizando essa abordagem híbrida. Na primeira coluna está colocada a máscara utilizada. Na segunda coluna o termo correspondente extraído e na terceira a freqüência do termo.

Padrão Sintático	Termos	Freqüência
N ADJ	motilidade normal	497
N ADJ	distensibilidade normal	496
N N	nível pinçamento	426
N	erosão	58
N	cor	18

Tabela 2.3: Fragmento de uma lista de termos extraídos usando a abordagem híbrida

2.3 Considerações Finais

Na área de ET são definidos métodos para extração de unidades terminológicas de um domínio. Nesse capítulo foram apresentadas sucintamente as principais abordagens para extração de terminologia. Neste trabalho, para a extração de termos de interesse foi aplicada a abordagem híbrida e, sobre as listas geradas, foram aplicadas heurísticas com o intuíto de reduzir o número de termos encontrados.

Capítulo 3

Ferramentas Auxiliares

3.1 Considerações Iniciais

Neste capítulo são apresentadas as ferramentas utilizadas para auxiliar no desenvolvimento da metodologia proposta neste trabalho. Todas essas ferramentas são de uso livre e são freqüentemente utilizadas pela comunidade.

3.2 Toquenizador

O algoritmo de toquenização transforma o texto em uma seqüência de caracteres os quais são agrupados em um primeiro nível segundo fronteiras delimitadas por caracteres primitivos como espaço (' '), vírgula e ponto. Tanto os grupos de caracteres como os delimitadores são denominados *tokens*[1] na nova seqüência, na qual o único caractere não considerado é o espaço em branco. Neste trabalho é utilizado o pacote implementado em Perl (Schwartz et al., 1997) String::Tokenizer[2]. Esse pacote permite a geração de *tokens* a partir de uma seqüência de texto. Opcionalmente, o algoritmo aceita um conjunto de caracteres, que são utilizados como delimitadores, os quais são retornados com a seqüência toquenizada. Isso permite que a seqüência seja dividida de várias maneiras diferentes. Para exemplificar, considere o seguinte texto:

> E agora, José?
> A festa acabou,
> a luz apagou,
> o povo sumiu,
> a noite esfriou,
> e agora, José?
> e agora, Você?
> Você que é sem nome,
> que zomba dos outros,
> Você que faz versos,
> que ama, protesta?

[1] Utilizaremos o termo *token* uma vez que é amplamente utilizado na literatura.

[2] `http://search.cpan.org/~stevan/String-Tokenizer-0.05/lib/String/Tokenizer.pm`

e agora, José?
...
Carlos Drummond de Andrade

Usando o pacote String::Tokenizer e a *string* de caractres ",.()?;", o resultado desse processo é uma seqüência de palavras intercaladas por espaço e algumas vezes por símbolos delimitadores, conforme apresentado a seguir:

E agora , José ?
A festa acabou ,
a luz apagou ,
o povo sumiu ,
a noite esfriou ,
e agora , José ?
e agora , Você ?
Você que é sem nome ,
que zomba dos outros ,
Você que faz versos ,
que ama , protesta ?
e agora , José ?
...
Carlos Drummond de Andrade

3.3 Segmentador

O objetivo do segmentador é dividir o texto em segmentos menores. Neste trabalho foi utilizado o SENTER[3] (Pardo, 2006), desenvolvido no NILC[4], o qual é um segmentador sentencial automático para textos em português do Brasil. O segmentador SENTER faz uso de várias regras para realizar a segmentação do texto, geralmente utilizando como caracteres delimitadores de sentença o ponto final, caracteres de exclamação e interrogação ou quando um marcador de nova linha (*carriage return* ou *line feed*) são encontrados. A seguir é apresentado um exemplo do processo de segmentação utilizando o seguinte texto:

Gastei uma hora pensando em um verso que a pena não quer escrever. No entanto ele está cá dentro inquieto, vivo. Ele está cá dentro e não quer sair. Mas a poesia deste momento inunda minha vida inteira.
...
Carlos Drummond de Andrade

Depois de aplicar o segmentador, o texto acima será segmentado conforme a seguir:

[3]SENtence splitTER
[4]`http://www.nilc.icmc.usp.br/nilc/index.html`

Gastei uma hora pensando em um verso que a pena não quer escrever.
No entanto ele está cá dentro inquieto, vivo.
Ele está cá dentro e não quer sair.
Mas a poesia deste momento inunda minha vida inteira.
...
Carlos Drummond de Andrade

3.4 Substituições

Na aplicação da metodologia, algumas vezes é necessário realizar a substituição de um termo no texto, simples ou composto, por outro. Por exemplo, os termos `pontos esbranquiçados` e `placas esbranquiçadas` podem ser substituidos pelo termo `anormal`. Neste trabalho utilizamos o módulo Text::RewriteRules[5], implementado na linguagem Perl, para aplicar as substituições. Nesse módulo podem ser declaradas diversos tipos de substituições, desde substituições simples, de um termo por outro, até substituições mais complexas, como substituição de uma seqüência que casa com determinada expressão regular por outra seqüência qualquer. As substituições são declaradas por meio de regras, onde cada regra possui um identificador e as substituições são mapeadas no formato antecedente e conseqüente, por exemplo:

```
RULES calibre
calibre distensibilidade normais.==>calibre normal.
                                    distensibilidade normal.
ENDRULES
```

Nesse exemplo, foi mapeada a regra cujo identificador é "calibre", a qual possui como antecedente "calibre e distensibilidade normais." e conseqüente "calibre normal." e "distensibilidade normal.". Os identificadores das regras podem ser vistos como funções as quais possuem um parâmetro de entrada e que podem ser chamadas como uma função qualquer definida na linguagem Perl. Caso o parâmetro de entrada seja um padrão ou uma seqüência que "case" com o antecedente da regra, será retornado a seqüência que aparece no conseqüente da regra. Se for realizada a chamada "calibre" com os parâmetros "calibre distensibilidade normais.", por exemplo, "calibre(calibre distensibilidade normais.), será retornado (sem aspas):

"calibre normal."
"distensibilidade normal."

O arquivo de regras deve seguir a sintaxe de um *script* implementado na linguagem Perl, conforme o arquivo ilustrado a seguir:

[5] `http://search.cpan.org/~ambs/Text-RewriteRules-0.10/lib/Text/RewriteRules.pm`

```
use Text::RewriteRules;

RULES calibre
calibre distensibilidade normais.==>calibre normal.
                                    distensibilidade normal.
ENDRULES

RULES junta

(\d+)\s+,\s+(\d+)==>$1,$2

ENDRULES

RULES teg
teg situada nível pinçamento diafragmático==>teg normal
ENDRULES
1;
```

Conforme pode ser observado, na regra cujo identificador é "junta", é utilizada uma expressão regular. Caso a expressão regular mapeada no antecedente da regra "casar" com alguma seqüência de texto, a seqüência é alterada para a expressão regular que está mapeada no conseqüente da regra. Nesse caso, a expressão regular está indicando que caso for identificado algum dígito (`\d`+)[6], separado por um ou mais espaços (`\s`+), depois por uma vírgula (,), e novamente por um ou mais espaços (`\s`+) e um dígito (`\d`+), essa seqüência será transformada para a seqüência indicada pela expressão `$1,$2`, a qual indica que o conteúdo entre parentes será juntado. Por exemplo, a seqüência "10 , 2" será transformada para "10,2". Depois de construído esse arquivo, ele pode ser utilizado para realizar as substituições no texto.

3.5 Etiquetador MXPOST

A etiquetação é um processo que associa a cada termo de um texto a classe gramatical, como substantivo, adjetivo e verbo. Neste trabalho, para realizar a etiquetação do conjunto de documentos utilizados na extração de terminologia, é utilizado o etiquetador, para língua portuguesa do Brasil, desenvolvido no Núcleo Interinstitucional de Lingüística Computacional — NILC. Para construir esse etiquetador, foi utilizado o MXPOST (Ratnaparkhi, 1996), o qual é um sistema que utiliza um modelo de máxima entropia[7]. No NILC, o etiquetador

[6]Quando a expressão regular está entre parentes o valor é guardado em variáveis do tipo $1, $2, etc.

[7]Modelos de máxima entropia oferecem um meio de combinar diversos fragmentos de evidências contextuais para estimar a probabilidade de uma certa classe lingüística ocorrer com um determinado contexto lingüístico. No MXPOST, as características do contextos utilizadas no processo de etiquetação são a palavra que será etiquetada e as duas palavras antes e depois, além das duas etiquetas anteriores.

foi treinado com 104.966 palavras de um *corpus* de textos em português do Brasil[8]. Nesse *corpus* estavam presentes três diferentes gêneros de textos: jornalísticos (56.653 palavras), didáticos (16.256 palavras) e literários (32.045 palavras). Atualmente esse etiquetador tem uma precisão de 97%.

Para usar o MXPOST, cada *token* deve estar separado por um espaço em branco, ou seja, nenhum caracter, incluindo pontuação, deve estar anexo às palavras; O arquivo de saída gerado apresenta o seguinte formato:

"token1_TAG1 token2_TAG2 ..."

onde TAG_i = {N, VERB, ADJ, PREP, ADV, ART, NUME, CONJ, PRON}.

3.6 *N-gram Statistics Package*

O N-GRAM STATISTICS PACKAGE — NSP[9] (Banerjee & Pedersen, 2003) é composto por um conjunto de programas que auxiliam na análise de n-gramas em arquivos de textos. No pacote, um n-grama é definido como uma seqüência de n palavras que ocorrem dentro de uma janela de pelo menos n palavras do texto. Esse conjunto de programas foi desenvolvido na linguagem Perl e utiliza dois programas principais para realizar a análise dos n-gramas: *count.pl* e *statistic.pl*.

O programa *count.pl* obtem, a partir de um conjunto de textos puro, uma lista de n-gramas que ocorrem em todos os arquivos. Como resultado, os n-gramas, com suas freqüências, são apresentados em ordem decrescente de freqüência.

No programa *count.pl* podem ser especificadas várias opções para serem utilizadas na geração dos n-gramas. Uma delas é a opção "ngram NUMERO", a qual deve ser utilizada quando se quer gerar um número maior de n-gramas que não o padrão, que é 2. Outra opção, "token nome_do_arquivo.pl", permite que seja utilizado um arquivo contendo regras de formação de *tokens*. A opção "stop nome_do_arquivo.pl" possibilita a passagem de uma arquivo de *stopwords* que não devem ser consideradas na geração de n-gramas. A opção "remove N" possibilita que seja limitada a lista de n-gramas, utilizando somente aqueles que apresentam freqüência equivalente ou superior a um valor especificado N. Além dessas, na documentação do NSP podem ser encontradas outras opções que podem ser utilizadas no momento da geração da lista de n-gramas. Para mostrar o funcionamento do NSP, é apresentado um exemplo utilizando o seguinte texto, armazenado no arquivo *texto.txt*.

> No meio do caminho tinha uma pedra
> tinha uma pedra no meio do caminho

Depois de executar a linha de comando "count.pl texto.txt saida.txt" é gerado o seguinte resultado:

[8] http://www.nilc.icmc.usp.br/nilc/download/MXPOST.zip
[9] http://www.d.umn.edu/~tpederse/nsp.html

```
13
tinha<>uma<>2 2 2
meio<>do<>2 2 2
uma<>pedra<>2 2 2
do<>caminho<>2 2 2
no<>meio<>1 1 2
caminho<>tinha<>1 1 2
pedra<>tinha<>1 2 2
No<>meio<>1 1 2
pedra<>no<>1 2 1
```

O número 13 na primeira linha indica que o arquivo de entrada *texto.txt* tem um total de 13 bigramas. Nas próximas linhas estão listados os bigramas, considerando que cada *token* é separado pelo sinal <>. Depois do último sinal em cada linha são informados três valores. O primeiro representa o número de vezes que o bigrama ocorre no arquivo texto de entrada. Por exemplo, no bigrama "tinha<>uma<>2 2 2", o primeiro número indica que este bigrama aparece duas vezes no texto. O segundo número indica o número de bigramas em que o *token* "tinha" ocorre do lado esquerdo. Por fim, o terceiro número indica o número de bigramas em que o *token* "uma" ocorre do lado direito.

O programa *statistic.pl* usa como entrada a lista de n-gramas com suas freqüências geradas pelo programa *count.pl* e calcula uma medida estatística de associação, a qual pode ser selecionada pelo usuário, para calcular o escore de cada n-grama. Os n-gramas, com seus escores, são apresentados em ordem decrescente de seus escores.

Além dos dois programas principais, vários outros programas auxiliares podem ser utilizados, os quais recebem como entrada as saídas geradas pelo programa *count.pl* ou *statistic.pl*. No pacote NSP, atualmente na versão 1.03, podem ser encontradas 10 medidas de associação, algumas para 2-gramas e outras para 3-gramas, as quais são apresentadas na Tabela 3.1.

2-**gramas**	3-**gramas**
Dice Coefficient	*Log-likelihood ratio*
Fishers exact test - left sided	*Mutual Information*
Fishers exact test - right sided	
Fishers twotailed test - right sided	
Jaccard Coefficient	
Log-likelihood ratio	
Mutual Information	
Odds Ratio	
Pointwise Mutual Information	
Phi Coefficient	
Pearson's Chi Squared Test	
Poisson Stirling Measure	
T-score	

Tabela 3.1: Medidas de associação do pacote NSP

Para exemplificar a aplicação de uma medida estatística, considere a lista de bigramas gerada pelo *count.pl* no exemplo anterior. Executando o programa *statistic.pl* e usando a medida estatística *Dice Coefficient*, o seguinte resultado é gerado.

```
13
tinha<>uma<>1 1.0000 2 2 2
uma<>pedra<>1 1.0000 2 2 2
do<>caminho<>1 1.0000 2 2 2
no<>mei<>1 1.0000 1 1 1
No<>meio<>1 1.0000 1 1 1
mei<>do<>2 0.6667 1 1 2
caminho<>tinha<>2 0.6667 1 1 2
meio<>do<>2 0.6667 1 1 2
pedra<>no<>2 0.6667 1 2 1
pedra<>tinha<>3 0.5000 1 2 2
```

Comparando esse resultado com o anterior, é possível notar que existem dois valores adicionais, sendo que o primeiro representa a classificação do bigrama, que é obtido a partir do segundo número, o qual representa o escore do bigrama, que foi calculado, neste caso, com a medida estatística de associação *Dice Coefficient*. Os três valores restantes são iguais aos apresentados no resultado do programa *count.pl*.

3.7 Lematizador

A lematização faz parte da normalização do texto e consiste na redução das palavras à sua forma canônica. Se a palavra for verbo, é reduzida a sua forma infinitiva. Se a palavra for substantivo ou adjetivo, é transformada para a forma masculino singular. Neste trabalho utilizamos como lematizador a ferramenta TREETAGGER[10] (Schmid, 1994). O TREETAGGER é uma ferramenta utilizada para realizar a etiquetação de textos e também realizar a extração de lemmas. Para exemplificar, considere o seguinte texto.

> O segredo é não correr atrás das borboletas... É cuidar do jardim para que elas venham até você.

Executando o `TreeTagger` sobre o texto apresentado, é extraído o seguinte texto lematizado.

> o segredo ser não correr atrás de borboleta... ser cuidar de jardim para que ela vir até você.

[10] `http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/`

3.8 *Graph Visualization Software*

No desenvolvimento da metodologia proposta neste trabalho foi necessário identificar uma ferramenta para realizar a geração de figuras de árvores que são utilizadas durante a aplicação da metodologia, caso sejam realizadas interações com o especialista. Uma excelente ferramenta encontrada para esse propósito foi a *Graph Visualization Software* — GRAPHVIZ. A GRAPHVIZ[11] (Gansner & North, 2000) é uma ferramenta utilizada na geração de grafos e distribuída na comunidade através da licença *The Common Public License*. São fornecidos pela ferramenta três construtores para geração de grafos, cada um fornecendo um tipo de grafo diferente. São eles:

- dot;
- neato; e
- towapi.

Os três construtores trabalham de maneira semelhante. Primeiramente deve ser elaborado e indicado na ferramenta o arquivo no formato texto utilizando a linguagem dot[12], no qual deve constar a descrição do grafo a ser gerado. O usuário deve então selecionar qual o construtor que será utilizado, dentre os três citados. Em seguida deve ser informado qual o arquivo de saida e escolher qual o formato desse arquivo, o qual pode ser escolhido entre diversos formatos tais como GIF, JPEG, PNG e POSTSCRIPT. Depois de informados esses dados, o grafo pode ser gerado.

Conforme mencionado, cada construtor de grafo da ferramenta GRAPHVIZ serve para geração de um determinado tipo de grafo. A seguir é apresentado sucintamente a função de cada um deles.

3.8.1 Dot

O construtor dot é utilizado para desenhar grafos hierárquicos e direcionados. A seguir é apresentada a especificação na linguagem dot do código usado para geração do gráfico hierárquico ilustrado na Figura 3.1. O símbolo → presente no código, na linguagem dot, indica o sentido da aresta entre dois vértices.

```
digraph exemplo {
graph [charset=latin1];
"mucosa  terço" -> "distal";
"distal" -> "presença";
"presença" -> "erosões";
"presença" -> "lesões";
"presença" -> "úlcera";
"mucosa  terço" -> "proximal";
"mucosa  terço" -> "médio";
"médio" -> "presença";
```

[11] http://www.GraphViz.org/
[12] http://www.GraphViz.org/doc/info/lang.html

```
"proximal" -> "presença";
}
```

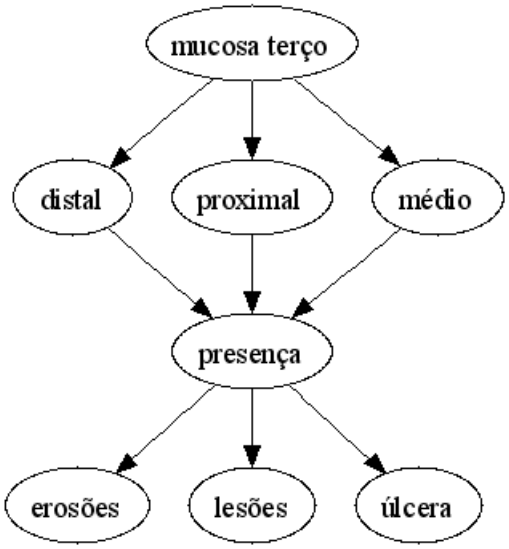

Figura 3.1: Gráfico hierárquico gerado utilizando o construtor dot

3.8.2 Neato

Neato é um construtor que gera desenhos de grafos não direcionados. A seguir é apresentado a especificação na linguagem dot do código usado para geração do gráfico não direcionado, ilustrado na Figura 3.2.

```
graph exemplo {
graph [charset=latin1];
"mucosa terço" -- "distal";
"distal" -- "presença";
"presença" -- "erosões";
"presença" -- "lesões";
"presença" -- "úlcera";
"mucosa terço" -- "proximal";
"mucosa terço" -- "médio";
"médio" -- "presença";
"proximal" -- "presença";
}
```

O neato é compatível com o construtor dot e pode compartilhar o mesmo arquivo de entrada. Uma vez que o formato de arquivo inclui tanto grafos direcionados e não direcionados, o neato pode desenhar grafos preparados para o construtor dot e vice-versa. Ambos os programas tem as mesmas opções de configurações de rótulos, cores, formas, fontes de texto e formato de figuras que podem ser geradas.

Figura 3.2: Gráfico não direcionado gerado utilizando o construtor neato

3.8.3 Twopi

O construtor twopi é destinado à geração de grafos circulares. Nesse construtor um vértice é escolhido como centro da circunferência e todos os outros vértices são dispostos em volta dele. Caso a quantidade de vértices desenhados em volta do vértice central gere sobreposições de vértices e arestas, outros círculos com raios diferentes podem ser criados, até que todos os vértices sejam desenhados.

A seguir é apresentado a especificação na linguagem dot do código usado para geração do grafo circular, ilustrado na Figura 3.3.

```
graph exemplo {
graph [charset=latin1];
"mucosa terço distal" -- "erosões";
"mucosa terço distal" -- "lesões";
"mucosa terço distal" -- "úlcera";
"mucosa terço distal" -- "pontos esbranquiçados";
}
```

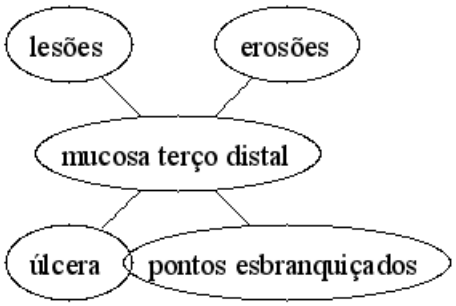

Figura 3.3: Gráfico circular gerado utilizando o construtor twopi

3.9 Estrutura do Dicionário

Neste trabalho consideramos que, na aplicação da metodologia, o mapeamento de um determinado atributo é realizado quando existe o casamento (*matching*) entre uma seqüência de texto encontrado no documento e uma seqüência de texto mapeada em um dicionário. Ou seja, se a seqüência do texto pertence a uma entrada no dicionário, o valor de um determinado atributo deverá ser preenchido na tabela. Na construção do dicionário utilizado para auxiliar no casamento de padrões de texto, é utilizada uma estrutura de Árvore Trie — AT (Ziviani, 2007).

Uma AT é uma árvore M-ária cujos nós são vetores de M componentes com campos correspondentes aos dígitos ou caracteres que formam as chaves. Cada nó no nível i representa o conjunto de todas as chaves que começam com a mesma seqüência de i dígitos ou caracteres. Esse nó especifica uma ramificação com M caminhos dependendo do $(i+1)$-ésimo dígito ou caracter de uma chave.

A seguir é apresentado o exemplo de uma AT considerando as chaves como seqüências de caracteres, ou seja M deve ser igual ao número de caracteres diferentes que podem ser utilizados para compor uma chave. Considere as seguintes chaves a partir das quais a AT será construída.

CASA, CARRO, BOM, BONITO, BONDADE.

A partir dessas chaves é construída a árvore ilustrada na Figura 3.4, na qual o caracter $ indica o final de uma palavra.

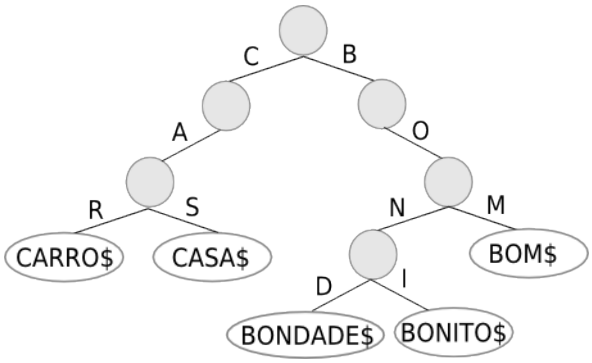

Figura 3.4: Exemplo de árvore trie

A construção da AT é realizada da seguinte maneira. Primeiramente é realizada uma pesquisa na árvore com a chave que será inserida. Se a pesquisa terminar em um nó vazio, ou seja, não tem filhos, é criado um novo nó com a chave que está sendo inserida. Caso a pesquisa termine em um nó que possui uma chave, é criado um ou mais nós internos da árvore para os quais os descendentes irão conter a chave já existente e a nova chave. Na Figura 3.5 é ilustrada a nova AT depois da inserção da palavra BONITEZA. Como pode ser observado foi criado um novo nó intermediário. Nesse caso, o nó BONITO$ foi substituido por um novo nó interno cuja subárvore direita é outro novo nó interno cujos filhos são BONITEZA$ e BONITO$, porque essas chaves possuem os mesmos caracteres até a quinta posição. Neste trabalho é utilizado o

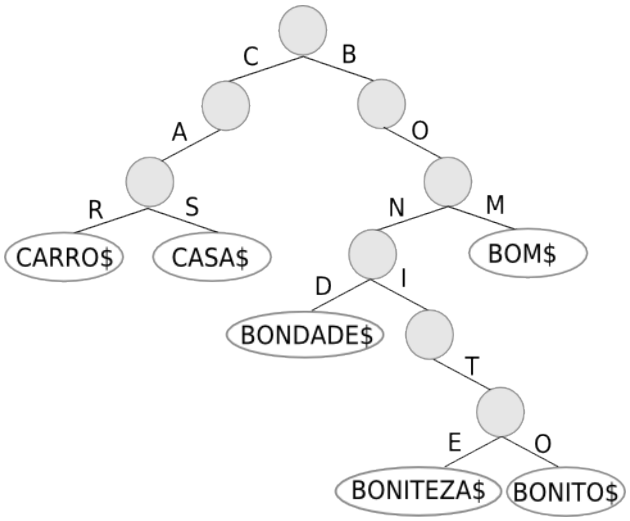

Figura 3.5: Exemplo de árvore trie após inserção da palavra BONITEZA

pacote implementado em Perl Tree::Trie[13]. Essa pacote permite a construção e pesquisa em árvores trie.

3.10 Considerações Finais

Neste capítulo foram apresentadas as ferramentas utilizadas para auxiliar no aplicação da metodologia de transformação de informações não estruturadas em informações estruturadas descritas em uma tabela atributo-valor. Foram apresentadas as ferramentas de toquenização, segmentação sentencial, lematização, etiquetação e a estrutura de dados utilizada para a construção do dicionário de conhecimento. No próximo capítulo é apresentada a metodologia desenvolvida e como essas ferramentas foram utilizadas para auxiliar na aplicação da metodologia.

[13] http://search.cpan.org/~avif/Tree-Trie-1.5/Trie.pm

Capítulo 4

Metodologia Desenvolvida

4.1 Considerações Iniciais

Como mencionado, o objetivo deste trabalho consiste na proposta e implementação de uma metodologia desenvolvida para auxiliar no mapeamento de informações contidas em laudos médicos não estruturados para tabela atributo-valor, a qual posteriormente será utilizada para realizar a extração de padrões usando, entre outros, algoritmos de aprendizado. A metodologia é composta pelas seguintes cinco fases, conforme apresentadas na Figura 4.1:

1. Pré-processamento;
2. Extração de terminologia;
3. Identificação de atributos;
4. Construção do dicionário; e
5. Construção da tabela atributo-valor.

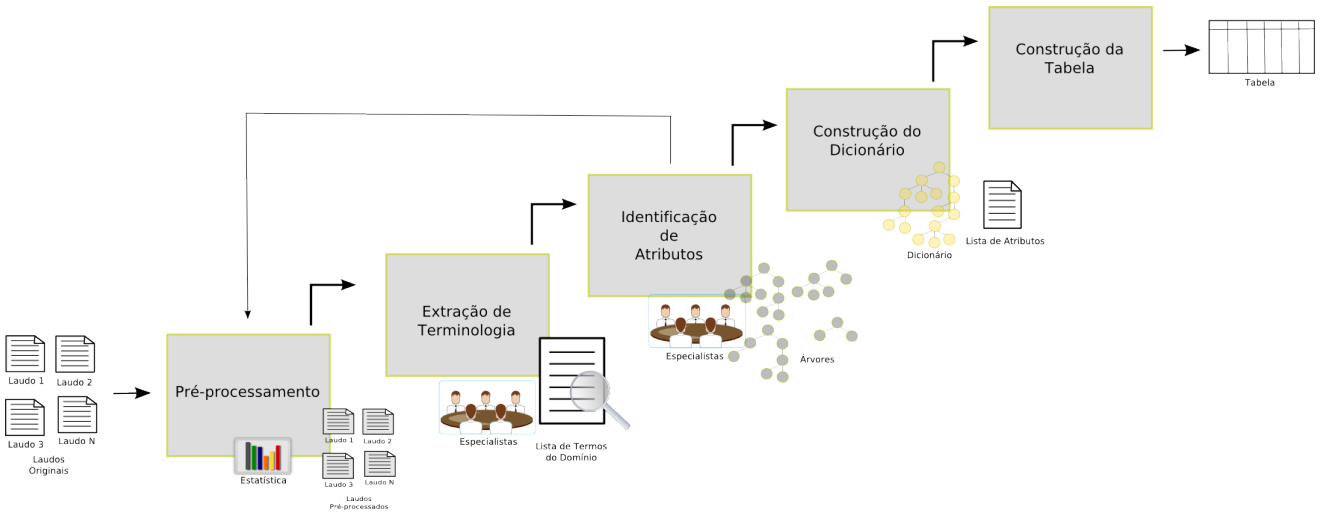

Figura 4.1: Metodologia desenvolvida

Na fase de pré-processamento, o conjunto de documentos é preparado para a próxima fase, a de extração de terminologia, por meio de tarefas tais como toquenização, segmentação, construção de frases únicas, remoção de *stopwords* e lematização. Na fase de extração de terminologia são geradas, a partir da coleção de documentos, listas de termos do domínio que possuem determinadas propriedades sintáticas. Os termos identificados são utilizados na terceira fase, na identificação de atributos, na qual, a partir de cada termo da lista de termos identificados, são geradas árvores a partir das quais são identificados os atributos que irão compor a tabela atributo-valor. Depois de identificados os atributos, os mesmos são mapeados em um dicionário, o qual é utilizado na última fase da metodologia, a de construção da tabela atributo-valor. Na construção da tabela são utilizados o conjunto de documentos, o dicionário e a lista de atributos identificados. A tabela é constituída por atributos binários, ou seja, o valor dos atributos é zero ou um; ou presente ou ausente. Portanto, durante o processamento do texto, se a seqüência de texto extraída de um documento (laudo) "casar" com uma seqüência inserida no dicionário, é mapeado no atributo correspondente da tabela o valor presente, caso contrário, é mapeado o valor ausente. A seguir é apresentada detalhadamente cada uma das fases da metodologia proposta.

4.2 Pré-processamento

A fase de pré-processamento consiste de sete etapas, as quais são aplicadas sobre o conjunto de laudos do domínio que está sendo trabalhado. Essas etapas correspondem a preparação do *corpus* por meio de tarefas como:

1. Segmentação do laudo;
2. Construção do Conjunto de Frases Únicas — CFU;
3. Remoção de *stopwords*;
4. Transformação para minúsculo;
5. Correção ortográfica;
6. Aplicação de substituições; e
7. Aplicação do lematizador.

4.2.1 Preparação do *Corpus*

Conforme mencionado no Capítulo 1, em muitos laudos do domínio médico, acontece que as informações são divididas, *i.e.*, segmentadas, dentro dos laudos. Por exemplo, o laudo da Figura 1.3 na página 15, de Andrologia, possui duas seções, uma na qual são mapeadas informações gerais do paciente e outra na qual é colocada a avaliação macroscópica do paciente. No laudo da Figura 1.4 na página 16, de Endoscopia Digestiva Alta, existe quatro divisões, uma na qual estão inseridas informações do esôfago, outra do estômago, outra do duodeno e outra de conclusões do exame. E, o laudo da Figura 1.5, na

página 17, de Colonoscopia, é dividido em duas seções, uma para informações gerais e outra para informações específicas do exame.

Conforme pode ser observado, o formato do laudo pode variar, mas, geralmente, existe uma certa segmentação nas informações, as quais podem ser observadas por meio dos termos indicativos (cabeçalhos) utilizados pelo médico durante o exame. Por exemplo, no laudo de EDA, ilustrado na Figura 1.4, podem ser observados quatro segmentos, conforme ilustrados na Figura 4.2.

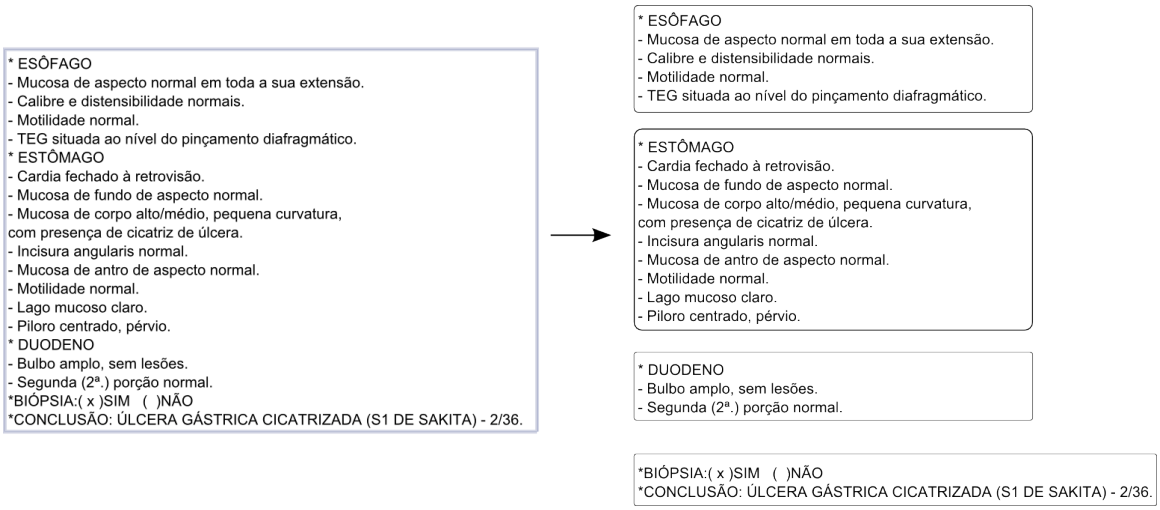

Figura 4.2: Divisão dos segmentos do laudo de Endoscopia Digestiva Alta

Neste trabalho adotamos a estratégia de processar cada segmento separadamente, conforme ilustrado na Figura 4.3. Deve ser observado que a tarefa

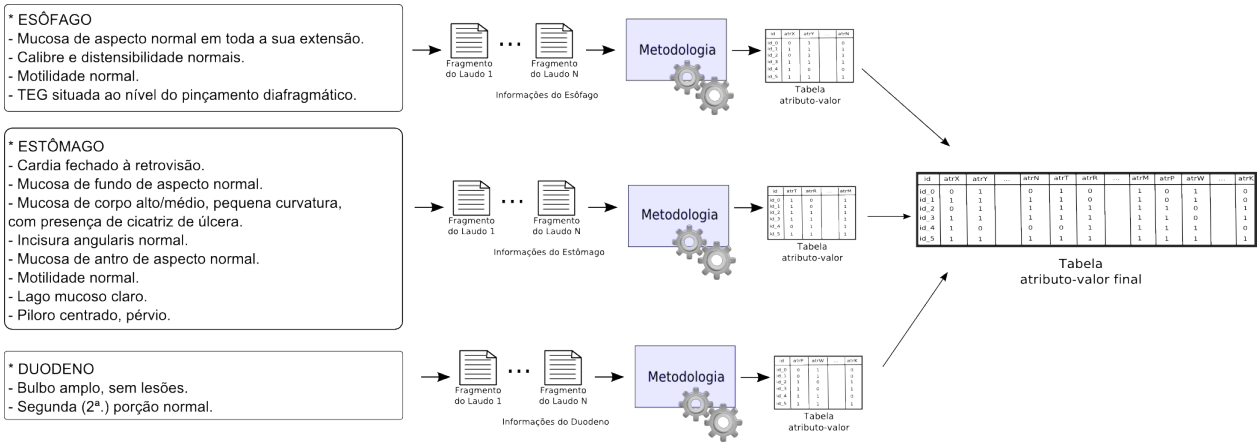

Figura 4.3: Exemplo de processamento dos segmentos do laudo

de divisão dos laudos em segmentos é específica para cada domínio do conjunto de laudos. Porém, após análise pelo especialista do domínio, essa tarefa pode ser realizada automaticamente. Após o processamento dos segmentos individuais, as tabelas geradas por cada segmento podem ser facilmente unidas, formando uma única tabela atributo-valor. Nas próximas seções, é considerado que os algoritmos estão sendo aplicados sobre um determinado segmento, caso o laudo possua vários segmentos.

4.2.2 Construção do Conjunto de Frases Únicas

A segunda etapa do pré-processamento tem por objetivo reunir em um único arquivo, denominado Conjunto de Frases Únicas, todas as frases diferentes de todos os documentos. Deve ser observado que as características do CFU são importantes para os bons resultados da metodologia proposta. Ele deve ser um conjunto com um número reduzido de frases. Isso se verifica no caso de laudos médicos, tratados este trabalho, mas também se verifica em documentos de outros domínios nos quais, também, tanto o vocabulário quanto a forma de expressar fatos são bastante controlados.

A Figura 4.4 ilustra o processo realizado para a geração do CFU. Nesse processo, a primeira tarefa a ser realizada é a toquenização, descrita na Seção 3.2, na qual é aplicado sobre o conjunto de documentos o algoritmo de toquenização.

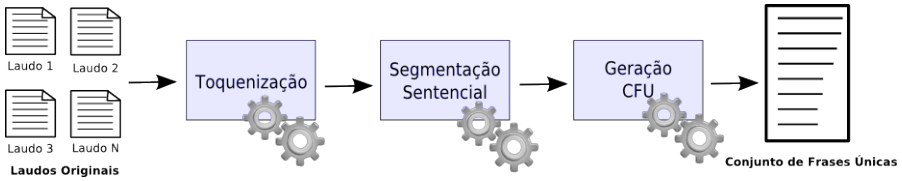

Figura 4.4: Exemplo de geração do Conjunto de Frases Únicas — CFU

Após a aplicação do algoritmo de toquenização, é aplicado o segmentador sentencial automático sobre o conjunto de documentos. A segmentação sentencial é o processo de segmentar um texto em unidades menores, que podem ser sentenças, parágrafos e até mesmo tópicos. A granularidade da segmentação depende exclusivamente da aplicação a que se destina a segmentação sentencial. Neste trabalho foi utilizado o segmentador SENTER, descrito na Seção 3.3 e a segmentação sentencial realizada sobre o conjunto de documentos resulta em setenças (frases) as quais são utilizadas na construção do CFU.

No Algoritmo 1 é apresentado o processo de geração do CFU.

Nesse algoritmo, primeiramente é realizada a toquenização e depois a segmentação sentencial do documento. No próximo passo é criado o conjunto de frases únicas CFU e é computada a freqüência dessas frases. Após a execução do algorítmo, essa informação é armazenada em um arquivo de frases únicas no formato XML. Na Figura 4.5 é ilustrado o fragmento de um arquivo de frases únicas, onde a *tag fraseUnica* armazena uma frase única, sua respectiva freqüência e se a frase foi processada ou não, caso esteja sendo realizada uma análise manual juntamente com o especialista.

São também extraídas e armazenadas em um arquivo XML várias informações estatísticas das frases que fazem parte do CFU, tais como número de frases, número de palavras, média de palavras por frase, tamanho da maior frase, maiores frases, tamanho da menor frase e menores frases. Na Figura 4.6 é ilustrado um arquivo XML com essas informações. Ambos arquivos XML, *i.e.*, o arquivo contendo o CFU e o arquivo contendo as estatísticas, são utilizados em etapas posteriores da aplicação da metodologia. É importante observar que a cada processamento que é realizado sobre o conjunto de documentos, tais como a aplicação de padronização, a correção ortográfica e a lematização,

Algoritmo 1 Construção do Conjunto de Frases Únicas

Entrada: Conjunto de Documentos $\{d_1, d_2, \cdots, d_n\}$
Saída: Conjunto de Frases Únicas e freqüências

```
CFU ← ∅
D ← {d_1, d_2, ···, d_n}
for all d_i ∈ D do
  d'_i ← aplicaToquenizador(d_i);
  d''_i ← aplicaSegmentador(d'_i);
  for all frase_j ∈ d''_i do
    if frase_j ∈ CFU then
      contador[frase_j] = contador[frase_j] + 1
    else
      CFU = CFU + {frase_j}
      contador[frase_j] = 1
    end if
  end for
end for
return CFU, contador
```


```
- <root>
  - <fraseUnica>
      <frequencia>100</frequencia>
      <frase>- Motilidade normal .</frase>
      <processada>NAO</processada>
    </fraseUnica>
  - <fraseUnica>
      <frequencia>99</frequencia>
      <frase>- Calibre e distensibilidade normais .</frase>
      <processada>NAO</processada>
    </fraseUnica>
  - <fraseUnica>
      <frequencia>86</frequencia>
    - <frase>
        - TEG situada ao nível do pinçamento diafragmático .
      </frase>
      <processada>NAO</processada>
    </fraseUnica>
      ...
```


Figura 4.5: Exemplo do arquivo do Conjunto de Frases Únicas

é gerado um novo CFU e as respectivas estatísticas. Os CFU anteriores são armazenados, uma vez que podem ser consultados posteriormente.

4.2.3 Remoção de *Stopwords*

Essa tarefa tem por objetivo remover do conjunto de documentos palavras que não são de interesse para análise do texto, ou seja, são palavras consideradas não relevantes tais como conjunções, artigos e preposições. A seguir é apre-

```
- <estatistica>
    <numeroDocumentos>100</numeroDocumentos>
    <numeroTotalFrases>398</numeroTotalFrases>
    <numeroFrases>33</numeroFrases>
    <numeroPalavras>314</numeroPalavras>
    <mediaPalavrasPorFrase>9</mediaPalavrasPorFrase>
    <tamanhoMaiorFrase>25</tamanhoMaiorFrase>
  - <maiorFrase>
      Observa-se à partir de terço médio presença de cordões
      varicoso , médio e grosso calibre , cor azul , sinal da cor
      vermelha presente- Calibre e distensibilidade normais .
    </maiorFrase>
    <tamanhoMenorFrase>2</tamanhoMenorFrase>
    <menorFrase>- Motilidade normal .</menorFrase>
  </estatistica>
```


Figura 4.6: Exemplo do arquivo de estatística do CFU

sentado um conjunto de frases, com as *stopwords* sublinhadas, e o respectivo conjunto no qual as *stopwords* foram removidas.

Frases com *stopwords*

> Mucosa em toda a sua extensão com presença de pontos e placas esbranquiçadas.
> Mucosa de terço distal com presença de uma erosão linear.
> Calibre e distensibilidade normais

Frases sem *stopwords*

> Mucosa toda extensão com presença pontos placas esbranquiçadas.
> Mucosa terço distal com presença uma erosão linear.
> Calibre distensibilidade normais.

A lista de *stopwords* (*stoplist*) pode ser incrementada por meio de interações com especialistas do domínio, nas quais, além das *stopwords* usuais, outras *stopwords* do domínio podem ser acrescentadas pelo especialista. As *stopwords* são armazenadas em um arquivo XML. Na Figura 4.7 é apresentado o fragmento de um arquivo de *stopwords*.

O processo de remoção de *stopwords* é aplicado para cada documento do conjunto de documentos, no qual são removidas as *stopwords* presentes na *stoplist*.

4.2.4 Transformação para Minúsculo

Nessa etapa, as informações presentes em todos documentos do conjunto são tranformadas para minúsculo.

```
− <root number="216">
    <stopword>uma</stopword>
    <stopword>de</stopword>
    <stopword>o</stopword>
    <stopword>a</stopword>
    <stopword>que</stopword>
    <stopword>e</stopword>
    <stopword>do</stopword>
    <stopword>da</stopword>
    <stopword>em</stopword>
    <stopword>um</stopword>
    <stopword>para</stopword>
    • • •
```

Figura 4.7: Exemplo do arquivo de *stopwords*

4.2.5 Correção Ortográfica

Conforme mencionado, muitas vezes durante a digitação das informações que o médico coletou do paciente para os laudos, são inseridas palavras com erros de ortografia. Sendo assim, foi desenvolvido um módulo que auxilia na correção dos termos que apresentam possíveis erros de ortografia. Duas maneiras de aplicar esse módulo de correção são possíveis, uma manual e outra semi-automática. Neste último caso são identificados automaticamente possíveis termos mapeados de maneira incorreta e é necessário uma interação do usuário para validar os termos que estão realmente com erros de ortografia.

No modo manual, deve ser analisado manualmente o CFU com o intuíto de identificar termos ortograficamente errados. Os termos identificados devem ser inseridos no módulo de correção juntamente com o termo correto. Por exemplo, se forem identificados os termos `aproximandamente` e `aproxmadmente` como termos errados, os mesmos devem ser inseridos no sistema junto com o termo correto `aproximadamente`, de modo que posteriormente sejam alterados os termos incorretos pelo termo correto.

Com o objetivo de auxiliar o usuário no processo de correção ortográfica do conjunto de documentos, neste trabalho foi implementado um algoritmo que auxilia nessa tarefa. Esse algoritmo utiliza uma idéia simples baseada no cálculo de distância entre palavras, e tem por objetivo identificar os possíveis termos que deverão ser corrigidos no conjunto de documentos. É importante ressaltar que o algoritmo não faz a correção automática mas sim gera um lista de possíveis correções.

A idéia de usar o cálculo de distância entre palavras foi introduzida em um algoritmo proposto por Damerau (1964), que difundiu a idéia de distância mínima de edição (*minimum edit*). Basicamente, o conceito de distância mínima de edição quantifica a idéia de uma seqüência de caracteres estar próxima a outra, pela contagem do número de operações de caracteres (como inserção, deleção e substituição) que são realizadas para transformar uma *string* em outra. Usando essa métrica, os melhores candidatos para a palavra correta são aqueles que apresentam a distância mínima de edição. No algoritmo implementado neste trabalho, no cálculo da distância entre palavras foi utilizada

a medida de Levenshtein (Levenshtein, 1966)[1], a qual é uma medida de similaridade entre duas *strings* S e T. A distância é o número de remoção, inserção e substituição necessária para transformar S em T.

O algoritmo implementado não faz uso de nenhum conhecimento adicional, mas apenas das informações presentes no conjunto de documentos. Para exemplificar, a seguir é apresentado um exemplo de sua aplicação. O algoritmo primeiramente gera, a partir do conjunto de documentos, uma lista de unigramas com suas respectivas freqüências. Nessa lista, cada unigrama também possui um marcador, o qual indica se o unigrama já foi processado durante a aplicação do algoritmo. Para exemplificar, considere a lista de unigramas e as respectivas freqüência no conjunto de documentos apresentada na Tabela 4.1.

Palavra	Freqüência	Processada
aproximandamente	2	não
aproximadamente	172	não
aproximadmaente	1	não
terço	274	não
terso	2	não
deterço	1	não

Tabela 4.1: Lista de unigramas

Primeiramente é obtido o primeiro termo da lista de unigramas, nesse caso `aproximandamente`, e ele é marcado como processado. Depois é criada uma lista auxiliar — LA e inserido o unigrama `aproximandamente` nessa lista. A partir desse ponto o algoritmo realiza o cálculo da distância da palavra `aproximandamente` com todas as palavras restantes da lista de unigramas. As palavras que possuem distância ≤ 2 são marcadas como processadas na lista de unigramas e inseridas na LA. Em seguida, depois de completamente processada a lista de unigramas, o algoritmo identifica a palavra que possuir maior freqüência na LA e coloca como sugestão de palavra correta, conforme ilustrado na Tabela 4.2.

Palavra	Freqüência	Sugestão Correta
aproximandamente	2	
aproximadamente	172	X
aproximadmaente	1	

Tabela 4.2: Lista auxiliar

As palavras restantes da LA são colocadas como possíveis palavras incorretas. Após gerar a primeira lista a partir da palavra `aproximandamente`, a lista de unigramas vai estar com a configuração ilustrada na Tabela 4.3. Nesse ponto, o algoritmo continua a execução para a próxima palavra da lista de unigramas ainda não processada, neste caso `terço`, e realiza o mesmo procedimento novamente.

No final do processamento tem-se a lista de sugestões de correção na qual cada sugestão possui uma ou mais possíveis palavras incorretas que deverão ser analisadas. A complexidade do algoritmo implementado é $O(n^2)$.

Finalmente, a lista de sugestões com as possíveis palavras incorretas correpondentes é armazenada em um arquivo do tipo XML. Na Figura 4.8 é ilustrado

[1] `http://www.merriampark.com/ld.htm`

Palavra	Freqüência	Processado
aproximandamente	2	sim
aproximadamente	172	sim
aproximadmaente	1	sim
terço	274	não
terso	2	não
deterço	1	não

Tabela 4.3: Lista de unigramas após processar a palavra `aproximandamente`

um trecho desse arquivo, no qual a *tag termoSugerido* armazena a sugestão de palavra correta e a *tag* listaTermos armazena a lista de possíveis palavras erradas que deverão ser substituídas pela palavra correta.

```
- <sugestao>
    <termoSugerido>aproximadamente</termoSugerido>
  - <listaTermos number="2">
      <termo>aproximandamente</termo>
      <termo>aproximadmaente</termo>
    </listaTermos>
  </sugestao>
- <sugestao>
    <termoSugerido>cm</termoSugerido>
  - <listaTermos number="2">
      <termo>cmv</termo>
      <termo>cmm</termo>
    </listaTermos>
  </sugestao>
- <sugestao>
    <termoSugerido>confluentes</termoSugerido>
  - <listaTermos number="1">
      <termo>cofluentes</termo>
    </listaTermos>
  </sugestao>
```

Figura 4.8: Exemplo do arquivo de correção de termos incorretos

Após construída essa lista, o usuário deve analisar, por meio de uma interface do sistema que implementa a metodologia proposta, e selecionar quais sugestões geradas pelo algoritmo estão corretas. Depois dessa seleção, o algoritmo de correção é executado, o qual processa todos os documentos do conjunto de documentos e aplica as correções sugeridas.

4.2.6 Normalização

A normalização parte do fato de que um conceito pode ter várias representações. A idéia é evitar que muitas formas de uma palavra seja associada com o mesmo conceito. Por exemplo, o conceito de "livro" é "objeto físico constituído de um número de páginas colocadas juntas" e pode ser representado por "livro" e "livros". O processo de normalização propõe que essas duas formas devem ser agrupadas em apenas uma que tem o mesmo significado. Alguns

tipos de normalização são o tratamento de sinônimos e a lematização. A seguir são descritas sucintamente essas abordagens e como elas são aplicadas neste trabalho.

4.2.6.1 Aplicação de Substituições

No domínio médico, é freqüente a utilização de sinônimos na descrição de informações semelhantes presentes nos laudos médicos ou a presença de frases que expressam dois eventos em uma mesma frase. Isso faz com que a normalização das informações contidas nos laudos seja necessária. Neste trabalho, o tratamento de sinônimos e de frases que expressam mais de um evento, tarefa que denominamos de aplicação de substituições e deve ser realizada manualmente pelo especialista. Após a obtenção do conjunto de frases únicas CFU é possível para o especialista identificar, manualmente, parte das informações que poderão ser substituídas. Na Tabela 4.4 são apresentados dois exemplos de substituições, definidas pelo especialista, que podem ser realizadas.

Antes da substituição	**Depois da substituição**
Coloração esbranquiçada	Anormal
Calibre distensibilidade normais	Calibre normal Distensibilidade normal

Tabela 4.4: Exemplos de substituições.

O termo `coloração esbranquiçada`, utilizado para descrever a característica de um tecido biológico, é sinônimo de que o tecido está `anormal`. A frase `calibre distensibilidades normais`, representa dois eventos diferentes, ou seja, essa frase indica que `calibre` está `normal` (`calibre normal`) e `distensibilidade` está `normal` (`distensibilidade normal`). Portanto, no arquivo de substituições deve ser mapeada a transformação dessa frase de modo a aparecer, depois da substituição, apenas um evento em cada frase. A construção do arquivo de substituição é realizada à medida que informações que podem ser substituídas são identificadas, o que pode acontecer em qualquer fase da metodologia. Desse modo, o arquivo de substituições permite que se trabalhe com um vocabulário controlado e também auxilia na padronização de frases com mais de um evento.

No mapeamento das substituições, sugeridas pelo especialista, é gerado um arquivo no formato XML no qual são armazenadas as substituições que devem ser realizadas. Diversos tipos de substituições podem ser mapeadas. As principais substituições identificadas são:

- Substituição de palavras;
- Substituição de frases;
- Substituição por expressão regular.

Na Figura 4.9 é ilustrado o trecho de um arquivo XML no qual estão mapeadas algumas substituições realizadas pelo especialista. Nesse arquivo, a *tag sequenciaOriginal* indica a seqüência que deverá ser substituida e na lista de substituições é armazenada a lista de substituições, definida pela *tag sequenciaNova*.

```
- <root number="3">
  - <substituicao identificador="calibre">
      <sequenciaOriginal>calibre distensibilidade normais .</sequenciaOriginal>
    - <listaSubstituicao number="2">
        <sequenciaNova>calibre normal .</sequenciaNova>
        <sequenciaNova>distensibilidade normal .</sequenciaNova>
      </listaSubstituicao>
    </substituicao>
  - <substituicao identificador="junta">
      <sequenciaOriginal>(\d+)\s+,\s+(\d+)</sequenciaOriginal>
    - <listaSubstituicao number="1">
        <sequenciaNova>$1,$2</sequenciaNova>
      </listaSubstituicao>
    </substituicao>
  - <substituicao identificador="teg">
      <sequenciaOriginal>teg situada nível pinçamento diafragmático</sequenciaOriginal>
    - <listaSubstituicao number="1">
        <sequenciaNova>teg normal</sequenciaNova>
      </listaSubstituicao>
    </substituicao>
  </root>
```

Figura 4.9: Exemplo do arquivo de substituição

Neste trabalho, é utilizado o módulo Text::RewriteRules, descrito na Seção 3.4. Assim, para aplicar as substituições, primeiramente foi desenvolvido um algoritmo para converter o arquivo XML gerado para o arquivo do Perl no formato utilizado pelo módulo Text::RewriteRules.

A seguir é ilustrado o arquivo gerado automaticamente a partir do XML apresentado na Figura 4.9.

```
use Text::RewriteRules;

RULES calibre
calibre distensibilidade normais.==>calibre normal.
                                    distensibilidade normal.
ENDRULES

RULES junta

(\d+)\s+,\s+(\d+)==>$1,$2

ENDRULES

RULES teg
teg situada nível pinçamento diafragmático==>teg normal
ENDRULES
1;
```

Depois de gerado automaticamente o arquivo de regras a ser utilizado pelo módulo Text::RewriteRules, o mesmo é utilizado no processamento do conjunto de documentos, conforme é ilustrado na Figura 4.10.

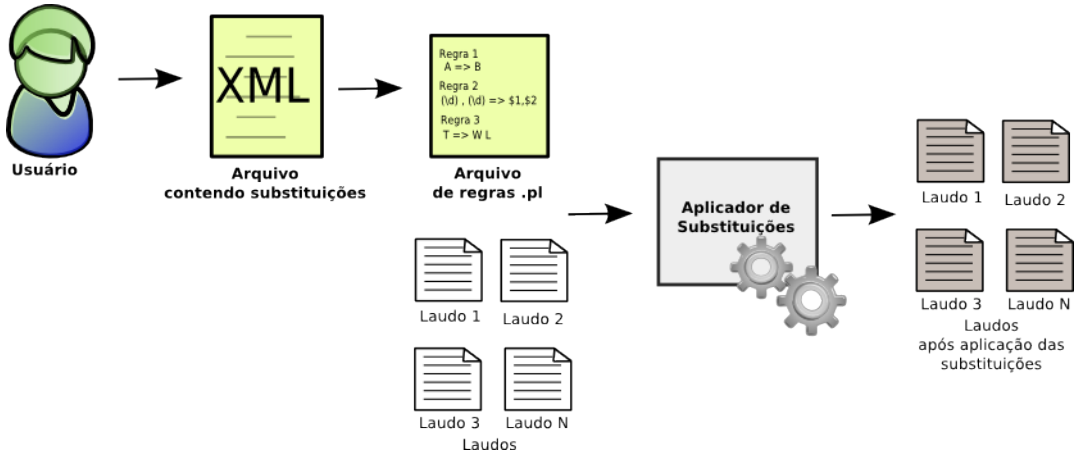

Figura 4.10: Processo de aplicação das substituições

O Algoritmo 2 apresenta o processo realizado para aplicar as substituições no conjunto de documentos. A variável $CDSA$ corresponde a Conjunto de Documentos com Subsituições Aplicadas.

Algoritmo 2 Aplicação de substituições

Entrada: Conjunto de Documentos $\{d_1, d_2, \cdots, d_n\}$ e arquivo de regras de substituição

Saída: Conjunto de Documentos com Substituições Aplicadas

LR $\leftarrow$ Lista de regras de substituições
$D \leftarrow \{d_1, d_2, \cdots, d_n\}$
$CDSA \leftarrow \emptyset$
for all $d_i \in D$ **do**
  $d'_i \leftarrow$ [ ]
  for all frase$_k \in d_i$ **do**
    for all identificador$_t \in$ LR **do**
      S = executa_Substituição(identificador$_t$, frase$_k$)
      d'_i = append(d'_i, S)
    end for
  end for
  $CDSA$ = $CDSA$ + { d'_i}
end for
return $CDSA$

Nesse algoritmo, para cada linha do documento é chamada a função definida pelo identificador. Conforme mencionado, caso alguma regra tenha o antecedente atendido, é retornada a seqüência de substituição, ou seja, o consequente da regra, a qual é substituída na linha do documento.

4.2.6.2 Aplicação de Lematização

Esta tarefa tem por objetivo aplicar a lematização sobre o conjunto de documentos que está sendo processado. Uma das muitas formas de representar uma palavra é substituir ela pela sua forma primitiva. A lematização é o processo de transformar verbos para sua forma no infinitivo e substantivos e adjetivos para o masculino singular. Neste trabalho é utilizado o TREETAGGER, descrito na Seção 3.7. No Algoritmo 3 é apresentado o processo utilizado para realizar a lematização do conjunto de documentos. A variável CDL corresponde a Conjunto de Documentos Lematizados.

Algoritmo 3 Lematização

Entrada: Conjunto de Documentos $\{d_1, d_2, \cdots, d_n\}$
Saída: Conjunto de Documentos Lematizados
  $D \leftarrow \{d_1, d_2, \cdots, d_n\}$
  $CDL \leftarrow \emptyset$
  for all $d_i \in D$ **do**
    d'_i = aplicaLematizador(d_i)
    CDL = CDL + { d'_i }
  end for
  return CDL

4.3 Extração de Terminologia

Depois de realizar o pré-processamento no conjunto de documentos, o mesmo está pronto para ser processado na segunda fase, a de extração de terminologia — Figura 4.1 na página 34. Nesse trabalho, a segunda fase tem por objetivo extrair termos do domínio que são frequentemente utilizados pelo profissional da saúde, geralmente um médico, no mapeamento de informações nos laudos. Neste caso, não estamos interessados em extrair todas as unidades terminológicas do domínio que está sendo abordado, mas nosso interesse é extrair as unidades terminológicas que indicam informações que estão sendo coletadas do paciente durante o exame e que encontram-se no laudo.

Para exemplificar o tipo de terminologia que estamos interessados considere os seguintes termos extraídos de um laudo de Endoscopia Digestiva Alta, especificamente do segmento de esôfago: `calibre`, `motilidade`, `terço distal`, `teg` e `mucosa`. Agora considere esses termos de acordo como aparecem freqüentemente nos laudos de esôfago:

> calibre normal.
> motilidade normal.
> terço distal com erosões lineares.
> teg situada nível pinçamento diafragmático.
> mucosa de aspecto normal em toda extensão.

Os termos sublinhados constituem nome de órgãos, localizações e características dessas localizações que são mapeadas pelo médico durante a realização do exame. A idéia de capturar apenas os termos acima citados está no fato

de que posteriormente, na nossa proposta, esses termos são utilizados para a geração de uma estrutura hierárquica (árvore) a partir da qual é possível identificar os possíveis relacionamentos (contexto) que esses termos possam ter. Nesse ponto, é necessário responder a seguinte pergunta: Como realizar a extração desses termos? Conforme mencionado, na área de extração de terminologia podem ser aplicadas três abordagens para a extração de unidades terminológicas: estatística, lingüística e híbrida.

Com base no conhecimento dessas técnicas para extração de terminologia foram analisadas algumas possibilidades que poderiam ser aplicadas para a extração de terminologia, no caso específico deste trabalho.

Para a identificação das unidades terminológicas que são utilizadas para gerar a estrutura hierárquica, neste trabalho é adotada uma abordagem híbrida seguida da aplicação de algumas heurísticas sobre o conjunto de termos identificados. O maior motivo para não realizar a extração de terminologia utilizando a abordagem estatística está no fato de que nesse trabalho é gerado um conjunto muito grande de termos e, desse modo, o custo de tempo de analisar esses termos seria muito alto. Outro motivo está no fato de que não dispomos de uma lista de referência, ou seja, uma lista com termos do domínio para auxiliar na identificação dos termos de interesse nos laudos. Deve ser observado que essa informação externa dos termos do domínio tratado, além de facilitar a tarefa de identificação de unidades terminológicas, permitem obter melhores resultados quando a metodologia é aplicada sem o auxílio do especialista para validar as unidades terminológicas encontradas utilizando a abordagem híbrida proposta neste trabalho. Na nossa proposta, isso é possível simplesmente não executando os passos que requerem a atença do especialista, ou seja, utilizando somente o conhecimento sintático extraído do conjunto de dados.

Conforme mencionado, na abordagem lingüística e híbrida, informações como classe gramatical, como nomes (N), adjetivos (ADJ) e verbos (VERB), são adicionadas no *corpus*. Um fator primordial para a escolha do método híbrido está no fato de que os termos de nosso interesse neste trabalho são, na maioria dos casos, pertencentes à uma determinada classe gramatical e, sobre esses termos, é calculada a freqüência de cada um. Depois de realizar a extração de termos usando a abordagem híbrida, são aplicadas algumas heurísticas que auxiliam na redução do número de termos que é utilizado na terceira fase da metodologia.

Com base nas considerações apresentadas, neste trabalho é proposta a seguinte metodologia para a extração da terminologia de interesse, ilustrada na Figura 4.11.

Essa metodologia é realizada em três etapas. Na primeira etapa é realizada a etiquetação do conjunto de documentos. Na segunda é realizada a extração de termos sobre o conjunto de documentos etiquetados, utilizando a abordagem híbrida. Na terceira etapa, são aplicadas algumas heurísticas sobre a lista de termos extraídos na segunda etapa com o objetivo de selecionar o subconjunto de unidades terminológicas a serem utilizadas posteriormente. A seguir é apresentado detalhadamente o processo de extração de unidades terminológicas proposto neste trabalho.

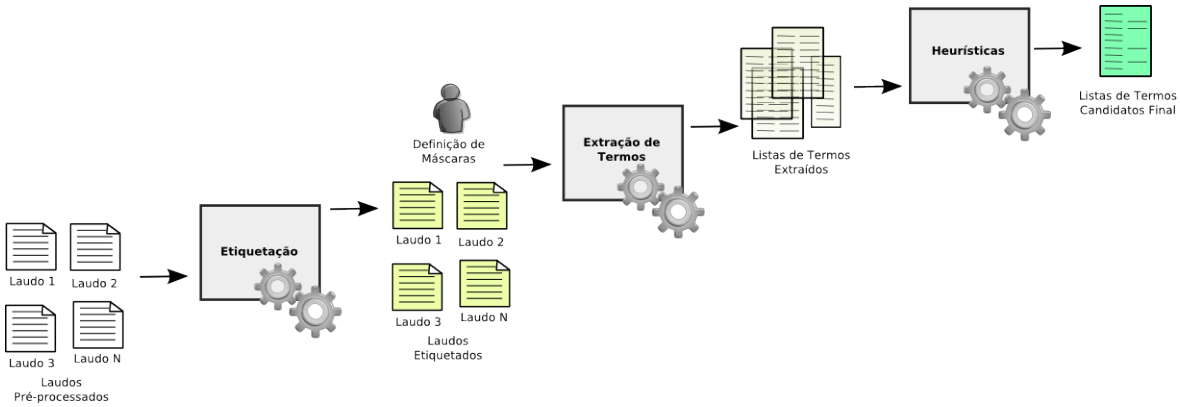

Figura 4.11: Metodologia para extração de termos

4.3.1 Etiquetação dos Documentos

A primeira tarefa que deve ser realizada no processo de extração de terminologia usando a abordagem híbrida é a etiquetação dos documentos a partir dos quais serão extraídos os termos. Neste trabalho utilizamos o etiquetador MXPOST descrito na Seção 3.5. Um problema identificado utilizando esse etiquetador, é que ele foi treinado com textos livres da língua portuguesa do Brasil. Assim, para o domínio específico da medicina o mesmo não é totalmente adequado pois coloca algumas etiquetas de maneira incorreta. Vale observar que o treinamento de um etiquetador é uma tarefa extremamente demorada que requer muito trabalho manual, o qual envolve uma grande força tarefa de pessoas envolvidas nas diferentes etapas do treinamento do etiquetador. Assim, para usar um etiquetador em domínios mais específicos, o mais aconselhável é realizar um *tunning* do etiquetador já treinado para conseguir atender o domínio específico. Com esse fim, neste trabalho foi adotado o seguinte procedimento. Primeiramente é aplicado o etiquetador MXPOST sobre o CFU gerado na primeira fase e definido pelo usuário, o qual pode ser o CFU padronizado e corrigido ou o CFU padronizado, corrigido e lematizado. Como resultado é gerado o CFU etiquetado.

Figura 4.12: Geração e análise do CFU etiquetado

Após gerar o CFU etiquetado é realizada uma análise manual do CFU etiquetado, com o intuíto de identificar as etiquetas incorretas, conforme mostra a Figura 4.12. É importante observar que para realizar essa análise manual não é necessário um especialista do domínio, qualquer pessoa munida de um

dicionário[2] pode realizar as correções. Quando um termo com uma etiqueta incorreta é identificado, o termo é adicionado em um arquivo, no formato XML, de correção de etiquetas, junto com a etiqueta correta. Depois que o CFU etiquetado for analisado por completo, o algoritmo de correção de etiquetas deve ser aplicado, conforme mostra a Figura 4.13, o qual irá retornar um novo CFU com as etiquetas corrigidas.

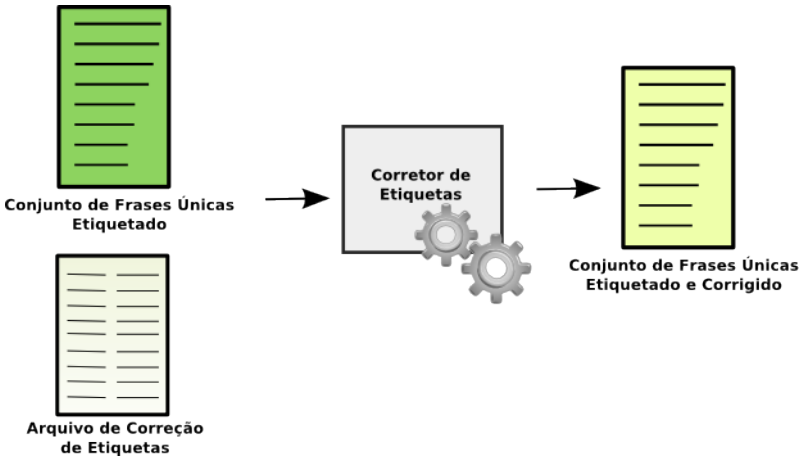

Figura 4.13: Aplicação do algoritmo de correção de etiquetas

Na Figura 4.14 é ilustrado o trecho de um arquivo de correção de etiquetas. Por exemplo, a *tag termos* refere-se ao termo analisado e a *tag etiqueta* corresponde a etiqueta correta a ser associada a esse termo. Depois de construído

```
− <root numero="17">
  − <correcao>
      <termo>calibre</termo>
      <etiqueta>N</etiqueta>
    </correcao>
  − <correcao>
      <termo>teg</termo>
      <etiqueta>N</etiqueta>
    </correcao>
  − <correcao>
      <termo>situado</termo>
      <etiqueta>VERB</etiqueta>
    </correcao>
  − <correcao>
      <termo>diafragmático</termo>
      <etiqueta>ADJ</etiqueta>
    </correcao>
    • • •
```

Figura 4.14: Exemplo do arquivo de correção de etiquetas

[2]Na Web existem alguns dicionário online, tais como o da Academia Brasileira de Letras, no endereço (http://www.academia.org.br/) e o Priberam, no endereço (http://www.priberam.pt/). Outro excelente dicionário para português disponível na internet é o Aulete Digital, disponível no endereço (www.auletedigital.com.br).

o arquivo de correção de etiquetas contendo os termos incorretamente etiquetados pelo MXPOST, o mesmo é utilizado quando for etiquetado o conjunto de documentos, conforme Figura 4.15. Nesse processo, primeiro é realizada a etiquetação do conjunto de laudos utilizando o etiquetador MXPOST e, depois de etiquetar os laudos, é aplicado o corretor de etiquetas sobre o conjunto de laudos, usando o arquivo de correção de etiquetas. O corretor implementado realiza o processamento no conjunto de documentos e os termos que tiveram uma nova etiqueta definida no arquivo de correção de etiquetas têm a etiqueta alterada. Como resultado desse processo tem-se o conjunto de laudos etiquetados com as etiquetas corretas, a partir dos quais pode ser realizada a extração de termos utilizando a abordagem hibrída.

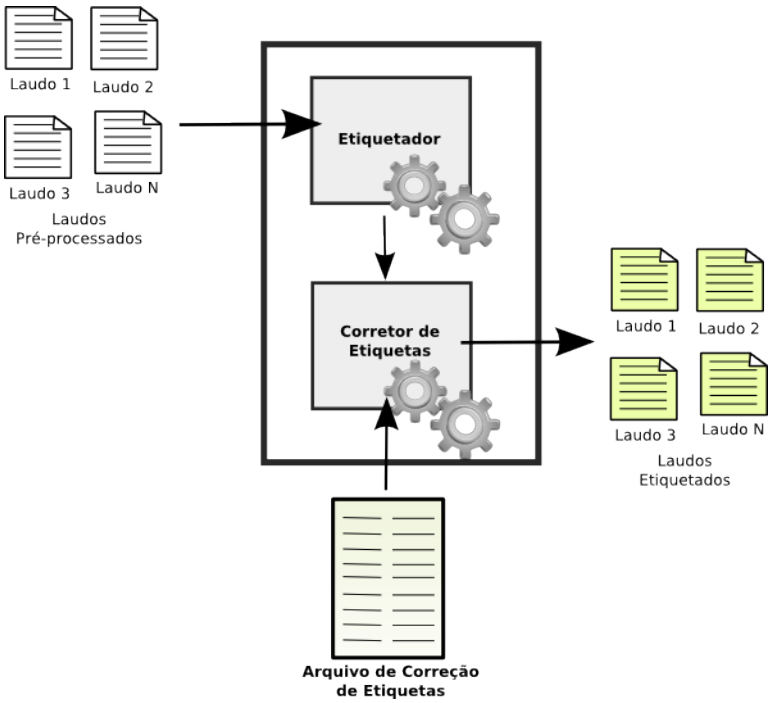

Figura 4.15: Processo de etiquetação do conjunto de documentos

4.3.2 Definição de Máscaras

A definição das máscaras que serão utilizadas para a extração de termos não é única para todos os domínios, ou seja, algumas máscaras são mais representativas em alguns domínios do que em outros. De acordo com a literatura, o ideal é realizar uma análise do *corpus* com o intuíto de identificar os termos mais freqüentes (Teline et al., 2003). Neste trabalho, a metodologia segue o princípio de que as informações são mapeadas como local ou característica de uma parte do corpo e informações associadas à esse local ou característica, conforme exemplo na Figura 4.16, no qual constam frases extraídas de um laudo de EDA, especificamente do segmento do esôfago. Nessa figura, os termos sublinhados, correspondem a locais e determinadas características do segmento do esôfago que são mapeadas pelo médico no momento do exame do paciente.

Na Figura 4.17 é ilustrado o arquivo da Figura 4.16 etiquetado. Observe

mucosa aspecto normal toda extensão .
calibre normal .
distensibilidade normal .
motilidade normal .
teg situada nível pinçamento diafragmático .

Figura 4.16: Exemplos de frases extraídas de um laudo

mucosa_N aspecto_N normal_ADJ toda_ADJ extensão_N ._.
calibre_N normal_ADJ ._.
distensibilidade_N normal_ADJ ._.
motilidade_N normal_ADJ ._.
teg_N situada_VERB nível_N pinçamento_N diafragmático_ADJ ._.

Figura 4.17: Exemplos de frases etiquetadas extraídas de um laudo

que os termos nos quais estamos interessados (sublinhados) aparece com a etiqueta N.

Na metodologia proposta, embora possam ser definidos diversos tipos de máscaras, notou-se que duas máscaras são freqüentemente utilizadas no mapeamento das informações. Essas duas máscaras, ilustradas na Tabela 4.5, foram definidas como padrão.

nome (N)
nome (N) nome (N)

Tabela 4.5: Tabela de máscaras padrão

4.3.3 Extração de Termos dos Documentos

Depois de definidas as máscaras mais apropriadas para o conjunto de documentos utilizados, é realizada a extração de termos, conforme ilustrado na Figura 4.18. Para realizar a extração de termos, o algortimo recebe o conjunto de documentos etiquetados e o conjunto de máscaras definido pelo usuário. O resultado são listas de termos, cada uma correspondente a uma determinada máscara que foi definida — duas listas no caso de utilizar as máscaras padrão. No Algoritmo 4 é apresentado o processo de extração de termos.

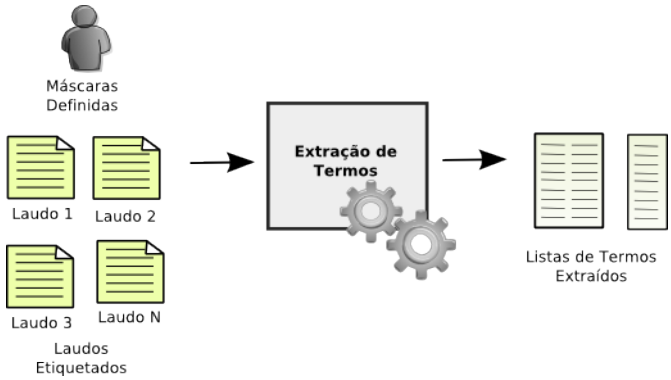

Figura 4.18: Processo de extração de termos

Algoritmo 4 Extração de termos

Entrada: Conjunto de Documentos $\{d_1, d_2, \cdots, d_n\}$ e Lista de Máscaras
Saída: Listas de Termos (LT) correspondentes às máscaras definidas e freqüências

```
LM ← Lista_de_Mascaras
LT ← ∅
D ← {d_1, d_2, ···, d_n}
for all d_i ∈ D do
  for all frase_k ∈ d_i do
    for all m_r ∈ LM do
      while termo_k = match_Padrão(m_r, frase_k) do
        if termo_k ∈ LT then
          contador[termo_k] = contador[termo_k] + 1;
        else
          LT = append(LT, termo_k)
          contador[termo_k] = 1
        end if
      end while
    end for
  end for
end for
return LT, contador
```

Após a construção da lista de termos e as respectivas freqüências pelo Algoritmo 4, ela é dividida em diversas listas de termos, cada lista correspondente a uma determinada máscara que foi definida. Por exemplo, se foram definidas as máscaras N, N N e N ADJ, serão criadas três listas de termos com as freqüências correspondentes, ou seja, uma lista com os termos que casaram com a máscara N, outra lista com termos que casaram com a máscara N N e uma lista de termos que casaram com a máscara N ADJ. Para ilustrar, na Tabela 4.6 são apresentadas as listas de termos geradas a partir das máscaras N (unigramas) e NN (bigramas), juntamente com as respectivas freqüências extraídas. Essas duas listas foram geradas a partir de um conjunto de 100 laudos de Endoscopia Digestiva Alta, especificamente do esôfago.

Nessa Tabela 4.6 é possível observar que muitos termos possuem baixa

N	Freqüência		N N	Freqüência
calibre	100		calibre distensibilidade	100
pinçamento	100		nível pinçamento	86
motilidade	100		mucosa aspecto	81
distensibilidade	100		mucosa terço	14
teg	99		presença erosões	10
mucosa	97		cm ads	4
nível	86		presença pontos	2
extensão	82		mucosa tipo	2
aspecto	81		presença cordões	1
terço	19		gástrica extendento	1
presença	13		erosões terço	1
erosões	11		área estenose	1
ads	4		sinal cor	1
coloração	4			
cm	4			
cor	3			
tipo	2			
pontos	2			
gástrica	2			
biópsia	2			
barrett	2			
cordões	2			
monilíase	2			
extendento	1			
área	1			
estenose	1			
circunferência	1			
lesão	1			
placas	1			
resíduos	1			
sinal	1			

Tabela 4.6: Lista de termos gerados com máscaras `N` e `N N`

freqüência. Além disso, é possível observar que alguns termos que aparecem na lista de unigramas também fazem parte da lista de bigramas. Nesse contexto, é bom lembrar que os termos utilizados pelos médicos para apresentar um determinado conceito são quase sempre os mesmos, e são aqueles que tem um freqüência maior.

4.3.4 Aplicação de Heurísticas

Com o objetivo de encontrar as unidades terminológicas mais apropriadas do domínio, neste trabalho são propostas algumas heurísticas com o intuíto de realizar uma filtragem sobre as listas de termos que foram geradas. Essas heurísticas trabalham sobre uma lista qualquer de unigrama e uma lista qualquer de bigramas e levam em consideração a freqüência dos termos presentes nas duas listas. Caso fossem utilizadas listas maiores que bigramas, por exemplo, trigramas, essas heurísticas deveriam ser re-definidas para levar em conta esse fato. Como mencionado, no domínio considerado, o uso de bigramas para encontrar as unidades terminológicas mostrou-se adequado e é o método padrão atualmente implementado.

As heurísticas implementadas são aplicadas em duas etapas. Na primeira etapa são analisados os termos da lista de unigramas que fazem parte de termos bigramas na lista de bigramas. Nessa etapa é gerada uma lista inicial de termos candidatos. Na segunda etapa é analisada a lista de candidatos com o intuíto de remover termos unigramas que fazem parte de ter-

mos bigramas. Por exemplo, na lista de candidatos pode existir o unigrama `distensibilidade` e o bigrama `calibre distensibilidade`. Dependendo das freqüências desses termos, o termo `distensibilidade` pode ser removido, pois o bigrama `calibre distensibilidade` é mais adequado. Após aplicadas essas heurísticas é analisada a lista de candidatos e removidos todos os termos que possuem uma freqüência menor do que um limiar definido.

Para exemplificar a aplicação das heurísticas propostas, considere as listas de unigramas e bigramas de termos apresentadas na Tabela 4.6. Na aplicação das heurísticas são utilizados dois parâmetros definidos pelo usuário: Alfa e Theta, os quais são utilizados na identificação das unidades terminológicas. No exemplo vamos utilizar o valor de Alfa=95%, o valor de Theta=10% e será apresentado em detalhes o processamento dos seguintes unigramas: `calibre`, `pinçamento`, `motilidade`, `distensibilidade`, `teg` e `mucosa`.

Inicialmente é criada uma Lista de Termos Candidatos —LTC vazia. Depois é inicializado o processameto das listas de unigramas e bigramas. Primeiramente é analisado o termo `calibre`. A partir desse termo, é verificado o número de vezes que ele aparece como primeiro termo na lista de bigramas. Nesse caso, podemos ter três situações: o termo `calibre` pode não aparecer, pode aparecer uma vez ou pode aparecer mais de uma vez. O parâmetro Alfa é utilizado nos dois últimos casos.

No caso do termo `calibre` podemos observar que ele aparece uma vez na lista de bigramas. Portanto, é realiza a seguinte verificação, conforme apresentado no Algoritmo 5, no qual a variável Unigrama corresponde a `calibre` e a variável Bigrama corresponde a `calibre distensibilidade`.

Algoritmo 5 Aplicação de heurísticas - parte 1

Entrada: Unigrama e Bigrama, Alfa
Saída: Termo selecionado para a lista de candidatos

```
Limiar ← (freqüência(Unigrama) * Alfa)/100
if freqüência(Bigrama) ≥ Limiar then
  return Bigrama
else
  return Unigrama
end if
```

Se a freqüência do bigrama, nesse caso, `calibre distensibilidade` for maior ou igual ao limiar calculado, então é escolhido o bigrama para ser inserido na LTC, senão é escolhido o unigrama `calibre`. Em outras palavras, a heurística proposta dá preferência a possíveis unidades terminológicas bigramas. Neste caso é inserido o termo `calibre distensibilidade` na LTC, já que o limiar é igual a 95%.

O próximo termo a ser processado é o termo `pinçamento`. A partir desse termo é verificado se existe algum bigrama cujo primeiro termo é `pinçamento`. Nesse caso não existe, portanto o termo `pinçamento` é inserido na LTC. O termo `motilidade` também é inserido na LTC, uma vez que não existe nenhum bigrama cujo primeiro termo é `motilidade`. Da mesma maneira acontece com os termos `distensibilidade` e `teg`.

No processamento do termo `mucosa` é identificado que na lista de bigramas existem três bigramas cujos primeiros termos são `mucosa`. Nesse caso, é utilizado novamente o parâmetro Alfa e é utilizado o seguinte procedimento, conforme apresentado no Algoritmo 6.

Algoritmo 6 Aplicação de heurísticas - parte 2

Entrada: Unigrama, Lista Bigramas e Alfa
Saída: Termo(s) selecionado(s) para a lista de candidatos

```
LB ← Lista Bigramas
LTB ← ∅
Limiar ← (freqüência(Unigrama) * Alfa)/100
soma = soma(LB, Unigrama)
if soma ≥ Limiar then
  LTB ← extrai(LB, Unigrama)
  return LTB
else
  return Unigrama
end if
```

Nesse algoritmo são somadas as freqüências dos três bigramas que contem o unigrama `mucosa`, pela função "soma(LB, Unigrama)" e, se a soma for maior do que o limiar definido, são extraídos, por meio da função "extrai(LB, Unigrama)" os três bigramas da lista de bigramas e inseridos na LTC, senão é adicionado apenas o unigrama `mucosa`. Nesse caso são adicionados os três bigramas na LTC[3].

O procedimento realizado para o processamento dos termos exemplificados acima é realizado para todos os termos restantes da lista de unigramas. Na Tabela 4.7 é apresentado o resultado da primeira etapa da aplicação das heurísticas.

Na segunda etapa é realizado o processamento da LTC com o objetivo de identificar e remover unigramas que fazem parte de algum bigrama. Os unigramas removidos são os que apresentam a mesma freqüência dos bigramas. No exemplo considerado, após essa etapa, não participam da LTC da Tabela 4.7, os termos `distensibilidade`, `ads`, `aspecto`, `tipo`, `pontos` e `estenose`. A nova LTC é apresentada na Tabela 4.8.

Na terceira etapa é realizada a remoção de todos os termos da LTC da Tabela 4.8 que possuem freqüência abaixo do limiar Theta, conforme apresentado no Algoritmo 7.

Nessa etapa é utilizado o parâmetro Theta e o número de laudos, portanto, todos os termos que possuem freqüência abaixo do limiar definido pelo Theta são removidos da LTC. A lista final, após a aplicação das heurísticas, é ilustrada na Tabela 4.9.

Após a execução desse algoritmo, tem-se a lista final de termos a serem utilizados como unidades terminológicas. É possível observar que houve uma

[3]Observe que se o parâmetro Alfa for definido como 100%, então apenas se o brigrama ou a soma dos brigramas for igual à freqüência do unigrama, o bigrama ou os bigramas são inseridos na LTC. Senão é inserido o unigrama.

Termos	Freqüência
calibre distensibilidade	100
pinçamento	100
motilidade	100
distensibilidade	100
teg	99
mucosa aspecto	81
mucosa terço	14
mucosa tipo	2
nível pinçamento	86
extensão	82
aspecto	81
terço	19
presença erosões	10
presença pontos	2
presença cordões	1
erosões	11
ads	4
coloração	4
cm ads	4
cor	3
tipo	2
pontos	2
gástrica	2
biópsia	2
barrett	2
cordões	2
moniliase	2
extendento	1
área estenose	1
estenose	1
circunferência	1
lesão	1
placas	1
resíduos	1
sinal cor	1

Tabela 4.7: Lista de termos candidatos gerados após realizar a primeira etapa

Algoritmo 7 Filtragem de termos candidatos

Entrada: Lista de Termos Candidatos, Theta e número de laudos
Saída: Lista de Termos Candidatos após remoção de termos

```
LTC ← Lista_Termos_Candidatos
Limiar ← (número_Laudos * Theta)/100
for all termo_i ∈ LTC do
  if freqüência(termo_i) ≤ Limiar then
    remove(LTC, termo_i)
  end if
end for
return LTC
```

redução significativa do número de termos na lista final em relação ao número de termos das listas originais. Os termos da lista final são utilizados na próxima fase, a de identificação de atributos que serão mapeados na tabela atributo-valor.

Termos	Freqüência
calibre distensibilidade	100
pinçamento	100
motilidade	100
teg	99
mucosa aspecto	81
mucosa terço	14
mucosa tipo	2
nível pinçamento	86
extensão	82
terço	19
presença erosões	10
presença pontos	2
presença cordões	1
erosões	11
coloração	4
cm ads	4
cor	3
gástrica	2
biópsia	2
barrett	2
cordões	2
monilíase	2
extendento	1
área estenose	1
circunferência	1
lesão	1
placas	1
resíduos	1
sinal cor	1

Tabela 4.8: Lista de termos gerados após realizar a segunda etapa

Termos	Freqüência
calibre distensibilidade	100
pinçamento	100
motilidade	100
teg	99
mucosa aspecto	81
mucosa terço	14
nível pinçamento	86
extensão	82
terço	19
erosões	11

Tabela 4.9: Lista de termos gerados após realizar a terceira etapa

4.4 Identificação de Atributos

Nesta fase, a partir de cada termo identificado na fase de extração de terminologia, é construída uma estrutura hierárquica, representada por uma árvore, cuja raíz é representada por uma unidades terminológica, a partir da qual são identificados os atributos para compor a tabela atributo-valor que será gerada na próxima fase. A partir de cada árvore podem ser identificados zero ou mais atributos para compor a tabela. A identificação dos atributos é realizada em três etapas:

- Definição de termos que serão utilizados como raíz da árvore;
- Geração das árvores; e
- Identificação dos atributos a partir da árvore gerada.

A seguir e apresentada uma descrição detalhada do processo de identificação de atributos.

4.4.1 Definição de Termos Raíz

Depois de gerada a lista de termos na fase anterior de extração de terminologia, a identificação dos termos que serão utilizados para gerar as árvores pode ser realizada de duas maneiras, as quais podem ser automática ou manual. No modo automático, todos os termos são considerados como termos raíz, ou seja, a partir de cada termo é gerada uma árvore. No modo manual, uma análise, junto com o especialista, pode ser realizada sobre a lista de termos com o intuíto de identificar os termos que realmente serão utilizados no mapeamento de informações. Nessa análise podem ser identificados termos da língua geral e desse modo, o especialista pode auxiliar nessa tarefa indicando quais termos poderão ser utilizados para raíz das árvores. É importante ressaltar que nessa abordagem está sendo realizada uma análise do significado dos termos, ou seja, está sendo utilizada semântica na análise das informações.

Para exemplificar, considere a Tabela 4.9 com a lista de termos identificadas na fase anterior. A partir dessa lista os termos `pinçamento`, `nível pinçamento`, `extensão` e `erosões` podem ser removidos, uma vez fazem parte da língua geral ou não caracterizam localizações ou o nome de um componente de um órgão que está sendo analisado. A Tabela 4.10 apresenta a lista final que seria utilizada, após a remoção desses termos, na construção das árvores na próxima etapa.

Termos	Freqüência
calibre distensibilidade	100
motilidade	100
teg	99
mucosa aspecto	81
mucosa terço	14
terço	19

Tabela 4.10: Lista de termos final

É importante ressaltar que dependendo do termo, se for da língua geral, por exemplo, a remoção do termo pode ser realizada por um leigo.

4.4.2 Geração de Árvores

Depois de definidos os termos raíz das árvores é executado o algoritmo de geração de árvores, lembrando que para cada termo considerado como unidade terminológica é gerada uma árvore cuja raíz é definida pelo termo. Neste trabalho, as árvores são geradas a partir de listas de n-gramas, as quais são geradas a partir do conjunto de documentos, pela ferramenta NSP, descrita na Seção 3.6.

Para exemplificar a construção das árvores considere que o termo `lago` pertence ao conjunto de unidades terminológicas identificadas. Considere também que foram encontrados os seguintes n-gramas[4] no conjunto de do-

[4]Observe que as unidades terminológicas identificadas previamente podem ser de 2-grama. Nesse caso são pesquisadas as listas de n-gramas para $n > 3$

cumentos — Tabela 4.11.

1-**grama**	2-**grama**	3-**grama**	4-**grama**	5-**grama**
lago	lago mucoso	lago mucoso claro	lago mucoso tinto bile	—
		lago mucoso tinto		

Tabela 4.11: Lista de n-gramas.

O número de n-gramas a ser considerado na construção da árvore é determinado pelo valor da variável $NMAX$ definida pelo usuário. $NMAX$ indica até qual lista de n-gramas deve ser realizada a pesquisa no momento de gerar as árvores. Por exemplo, se $NMAX$ é 3, então para gerar a árvore será pesquisada até na lista de trigramas. O valor de $NMAX$ padrão é definido pelo número médio de palavras por frase do conjunto de documentos. No exemplo é considerado $NMAX$ = 5.

Primeiramente é definida a árvore cuja raíz é o termo `lago`, conforme Figura 4.19.

Figura 4.19: Termo raíz da árvore

No próximo passo é realizado uma busca nas listas de n-gramas ($n > 1$ no exemplo abordado), nas quais o termo `lago` é o primeiro elemento, com o objetivo de definir as sub-árvores do termo definido como raíz. Primeiramente é processada a lista de 2-gramas. Como há somente um elemento nessa lista, que tem como primeiro termo `lago`, então o "último grama" desse 2-grama (`mucoso` no exemplo considerado) é inserido na árvore como único filho da raíz `lago`, conforme ilustra Figura 4.20

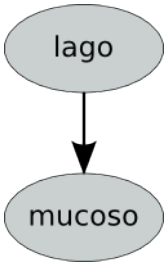

Figura 4.20: Filho do termo raíz

Após é analisada a lista de 3-grama correspondente ao 2-grama `lago mucoso`. Nesse caso tem-se dois 3-grama `lago mucoso claro` e `lago mucoso tinto`. Primeiramente é processado o 3-grama `lago mucoso claro`. Como $NMAX$ > 3, a lista 4-gramas é processada. Como não há nenhum 4-gramas que contem `lago mucoso claro`, o nó como o termo `claro` é um nó terminal da árvore. No caso do termo 3-grama `lago mucoso tinto`, considerando que existe somente um 4-grama `lago mucoso tinto bile` e não existe nenhum 5-grama que contém esse 4-grama, é inserido os termos `tinto bile` como nó terminal da árvore, conforme ilustrado na Figura 4.21.

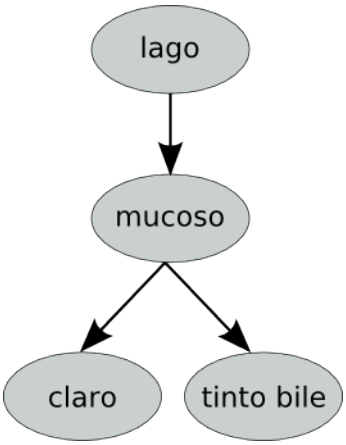

Figura 4.21: Filho do termo mucoso

Após geradas todas as árvores que tem como raíz as unidades terminológicas, elas são armazenadas em um arquivo no formato XML. Na Figura 4.22 é ilustrado o arquivo XML no qual está armazenada a estrutura de árvores correspondes às unidades terminológicas `lago` e `cardia`. Nessa árvore, a *tag nivel1* indica o primeiro nível da árvore, a *tag nivel2* indica o segundo nível e assim sucessivamente. Também são armazenadas as freqüências desses termos compostos de um nível para outro, de modo que possam ser utilizados no momento de identificação de atributos que irão compor a tabela atributo-valor.

A partir desse arquivo XML é possível gerar as figuras que representam as árvores que estão mapeadas nele. A geração das figuras das árvores é realizada utilizando a ferramenta GRAPHVIZ, a qual está descrita na Seção 3.8. As figuras das árvores são criadas individualmente, portanto, a partir das árvores apresentadas na Figura 4.22, serão geradas duas figuras, onde uma terá como raíz o termo `lago` e a outra terá como raíz o termo `cardia`. Nas Figuras 4.23 e 4.24 estão ilustradas, respectivamente, as árvores geradas a partir do termo `lago` e do termo `cardia`.

Foi implementado um procedimento que tem como entrada a descrição XML da árvore a ser desenhada e que gera o arquivo com os comandos correspondentes para a ferramenta GRAPHVIZ desenhar essa árvore. Na Figura 4.25 é ilustrado o arquivo de descrição da figura gerado a partir do termo `lago` e utilizado pelo GRAPHVIZ para desenhar a árvore final da Figura 4.23.

4.4.3 Definição de Atributos

Neste trabalho, a identificação dos atributos que participarão da tabela atributo-valor é realizada de maneira automática. Depois que foi gerada a árvore para cada termo raíz (unidade terminológica), é realizada uma busca nas árvores geradas, de modo a identificar o conjunto de atributos que fará parte da tabela atributo-valor. Neste trabalho foi definido que os atributos são binários, ou seja, o valor do atributo poderá receber apenas dois valores, os quais são, presente (1), ou ausente (0), tal que valor do $atributo = \{0, 1\}$.

A identificação dos atributos é realizada por meio de uma pesquisa automática em cada árvore, partindo do termo raíz. Nesse caso, o caminho a partir da raíz da árvore até um nó intermediário, ou mesmo, o nó folha que possuir

```
- <root number="2">
  - <nivel1>
    - <t1>
        <termo>lago</termo>
      - <nivel2 number="1">
        - <t2>
            <termo>mucoso</termo>
            <frequencia>109</frequencia>
          - <nivel3 number="2">
            - <t3>
                <termo>claro</termo>
                <frequencia>58</frequencia>
              </t3>
            - <t3>
                <termo>tinto bile</termo>
                <frequencia>51</frequencia>
              </t3>
            </nivel3>
          </t2>
        </nivel2>
      </t1>
    </nivel1>
  - <nivel1>
    - <t1>
        <termo>cardia</termo>
      - <nivel2 number="2">
        - <t2>
            <termo>fechado à retrovisão</termo>
            <frequencia>72</frequencia>
          </t2>
        - <t2>
            <termo>aberto à retrovisão</termo>
            <frequencia>13</frequencia>
          </t2>
        </nivel2>
      </t1>
    </nivel1>
  </root>
```


Figura 4.22: Arquivo correspondente a árvore gerada a partir dos termos `lago` e `cardia`

freqüência acima de um determinado limiar em relação ao número de laudos que foi utilizado para gerar a árvore, é mapeado como atributo. Esse limiar é definido pelo usuário e seu valor padrão é 5%. Considere a árvore ilustrada na Figura 4.26, gerada a partir do termo raíz `mucosa fundo`, a qual foi gerada utilizado 100 laudos do conjunto de laudos de Endoscopia Digestiva Alta.

Essa árvore informa que a seqüência `mucosa fundo aspecto normal` aparece 88 vezes no conjunto de 100 laudos, enquanto que a seqüência `mucosa corpo antro enantema difuso` aparece somente uma vez. Se for utilizado o limiar de 5%, isso indica que somente seqüências que aparecem no mínimo 5 vezes devem ser consideradas. Portanto, somente será identificado um atributo, o atributo `mucosa fundo aspecto normal`. Por outro lado, se a árvore for a ilustrada na Figura 4.23, gerada com os mesmos cem documentos, então com o limiar de 5%, serão identificados dois atributos: `lago mucoso claro` e `lago mucoso tinto bile`. Em outras palavras, nem todos ramos das árvo-

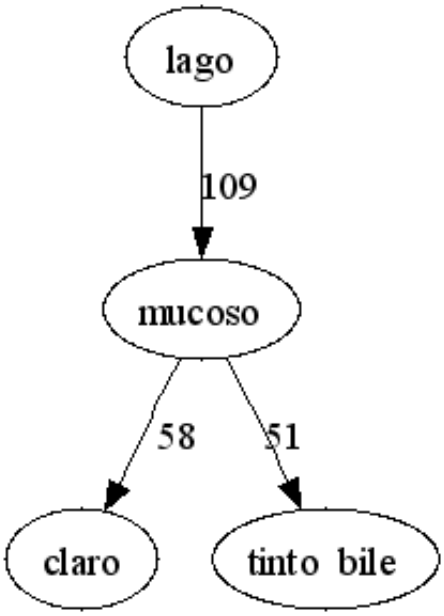

Figura 4.23: Árvore gerada a partir do termo `lago`

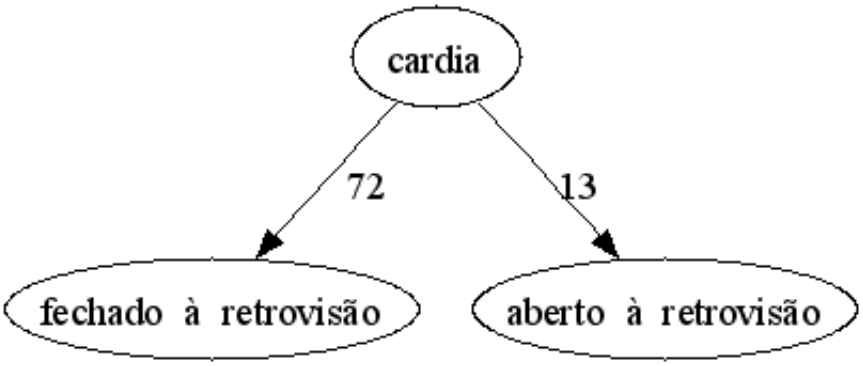

Figura 4.24: Árvore gerada a partir do termo `cardia`

```
digraph lago {
graph [charset=latin1];
"lago";
"lago" -> "mucoso" [label=109];
"mucoso" -> "claro" [label=58];
"mucoso" -> "tinto  bile" [label=51];
}
```

Figura 4.25: Arquivo com os comandos do GRAPHVIZ gerado a partir da descrição XML do termo `lago`

res, *i.e.*, seqüências de termos que aparecem nos laudos, são transformados em atributos. No final dessa pesquisa, todos ramos das árvores que possuem uma freqüência maior que um determinado limiar são mapeadas como atributos para compor a tabela atributo-valor[5].

Uma outra abordagem para a definição dos atributos é realizar uma análise

[5]Observe que um limiar de 0% faz com que sejam considerados todos os atributos na árvore, independente da sua freqüência no conjunto de documentos.

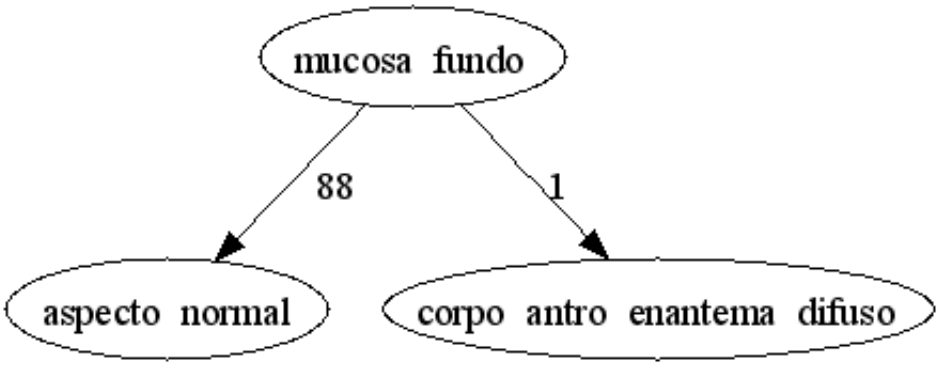

Figura 4.26: Árvore gerada a partir do termo `mucosa fundo`

manual das árvores geradas juntamente com o especialista do domínio. Nesse caso seria usado conhecimento semântico.

4.5 Construção do Dicionário

Depois que foram identificados os atributos, é necessário inseri-los em uma estrutura adequada que possibilite posteriormente a busca, no conjunto de documentos, por seqüências que "casem" com esses nomes de atributos. A esse estrutura denominamos *dicionário de conhecimento*. Neste trabalho foi adotada a árvore trie como a estrutura de dados para ser utilizada como dicionário de conhecimento. Essa estrutura é descrita na Seção 3.9. Depois de identificados os atributos, com base na análise das árvores geradas, os mesmos são inseridos na árvore trie. Na árvore trie são inseridas as seqüências identificadas nas árvores, por exemplo, se for identificado que a seqüência `lago mucoso claro` é um atributo, essa seqüência será inserida na árvore trie. Essa árvore é utilizada na última fase da metodologia, a de construção da tabela atributo-valor.

4.6 Construção da Tabela Atributo-valor

A tabela atributo-valor é a definida no Capítulo 1, Tabela 1.1, página 10, para exemplos não rotulados. Neste trabalho, cada exemplo d_i na Tabela 1.1, refere-se a uma laudo$_i$ do conjunto de laudos. O Algoritmo 8 descreve em alto nível a construção dessa tabela.

O preenchimento da tabela é realizado por meio de um ciclo de pesquisas entre o conjunto de documentos e o dicionário de conhecimento. Se uma determinada seqüência do laudo está presente no dicionário de conhecimento, é colocado no atributo correspondente da tabela o valor presente. Na Figura 4.27 é ilustrada um trecho de uma tabela atributo-valor na qual foram mapeados os três atributos identificados na Seção 4.4.3.

Nessa tabela são mapeados além do nome dos atributos, o nome do arquivo texto correspondente ao laudo que foi processado. Essa informação é importante, uma vez que laudos de determinado domínio podem ser segmentados,

Algoritmo 8 Geração da tabela atributo-valor

Entrada: Conjunto de Documentos $\{d_1, d_2, \cdots, d_n\}$ e Dicionário de Conhecimento DC

Saída: Tabela atributo-valor

```
D ← {d_1, d_2, ···, d_n}
for all d_i ∈ D do
  for all atributo_k ∈ DC do
    if encontrado(atributo_k, d_i) then
      tabela[i, atributo_k] = 1
    end if
  end for
end for
for i = 1 to n do
  for all atributo_k ∈ DC do
    if tabela[i, atributo_k] não definido then
      tabela[i, atributo_k] = 0
    end if
  end for
end for
return tabela
```

```
arquivo,lago_mucoso_tinto_bile,lago_mucoso_claro,mucosa_fundo_aspecto_normal
laudo4.txt.seg,0,1,1
laudo5.txt.seg,1,0,1
laudo6.txt.seg,1,0,1
laudo7.txt.seg,0,1,1
laudo8.txt.seg,0,1,1
laudo9.txt.seg,0,1,1
laudo10.txt.seg,0,1,1
laudo11.txt.seg,1,0,1
laudo12.txt.seg,0,1,1
laudo13.txt.seg,0,1,1
laudo14.txt.seg,1,0,1
laudo15.txt.seg,1,0,1
laudo16.txt.seg,0,1,1
laudo17.txt.seg,0,1,1
laudo18.txt.seg,1,0,1
laudo19.txt.seg,0,1,1
laudo20.txt.seg,1,0,1
laudo21.txt.seg,0,1,1
laudo22.txt.seg,0,1,1
laudo23.txt.seg,1,0,0
```

Figura 4.27: Exemplo da tabela atributo-valor

conforme foi apresentado na Seção 4.2.1 . Assim, com o nome do arquivo mapeado, posteriromente é possível, se for necessário, juntar as tabelas criadas para cada segmento do laudo em uma única tabela atributo-valor.

Na geração da tabela atributo-valor também é gerado um arquivo, o qual denominados de *garbage*, no qual são armazenadas as frases dos documentos que não sofreram nenhum processamento e as frases que foram processadas parcialmente. É bom ressaltar que essa é uma informação importante, uma vez que indica a porcentagem de frases que estão sendo processadas completamente, parcialmente ou não estão sofrendo nenhum processamento, em relação ao número total de frases do conjunto de documentos.

4.7 Considerações Finais

Neste capítulo foi descrita a metodologia de transformação de informações não estruturadas em informações estruturadas representadas em uma tabela atributo-valor. Nessa metodologia não é realizada uma análise aprofundada do texto como a realizada quando aplicando métodos da área de processamento de língua natural, na qual também é realizada a análise semântica. Neste trabalho é realizada uma análise sintática parcial, usando técnicas da área de extração de terminologia. A partir dessa análise, são identificados termos de interesse que são utilizados para a geração de estruturas hierárquicas (árvores) as quais são utilizadas para a identificação de atributos que irão compor a tabela atributo-valor. Depois de identificados os atributos é construído um dicionário de conhecimento, o qual é utilizado no momento do mapeamento das informações na tabela atributo-valor. No final da aplicação da metodologia tem-se a tabela atributo-valor preenchida a qual pode ser utilizada para a aplicação de algoritmos de mineração de textos.

Quanto a parte semântica, a metodologia proposta permite incorporar semântica, mas ela deve ser incorporada manualmente com a participação do especialista. Assim, como o especialista não participa do processo, tem-se um processo parcialmente sintático para escolher os atributos da tabela atributo-valor. No outro extremo, com a participação do especialista, tem-se um processo no qual o aspecto semântico também é considerado na metodologia proposta.

A metodologia foi implementada em um ambiente computacional, denominado de TP-DISCOVER, que oferece uma interface amigável de interação com o usuário. O sistema TP-DISCOVER está descrito em detalhes em Honorato & Monard (2008). Com o TP-DISCOVER, a metodologia foi avaliada utilizando diversos conjuntos de documentos que consistem de laudos de Endoscopia Digestiva Alta. Levando em consideração o aspecto puramente sintático (automático) e o aspecto semântico que requer a intervenção do(s) especialista(s), foi decidido avaliar a metodologia considerando somente o aspecto sintático. A idéia consiste em aplicar todas as fases da metodologia e analisar os atributos encontrados variando alguns dos parâmetros de execução do TP-DISCOVER. Após, os resultados são analisados qualitativamente e quantitativamente. A análise qualitativa é realizada conjuntamente com o especialista do domínio, com o objetivo de avaliar a qualidade dos atributos construídos automaticamente. Nessa avaiação, o especialista também pode observar se a sua intervenção durante a aplicação poderia ter corrigido esses atributos incorretos ou atributos faltantes. Para realizar essa tarefa, o arquivo de *garbage* que contem as frases que não foram processadas, ou que foram processadas parcialmente, fornece informações importantes para auxiliar a avaliação do especialista.

Os resultados obtidos da avaliação da metodologia devem sem consultadas em Honorato (2008).

Referências Bibliográficas

Almeida, G. M. D. B., de Oliveira, L. H. M., & Aluísio, S. M. (2006). A terminologia na era da informática. *Ciência e Cultura*, 58(2):42–45. Citado na página 19.

Bagot, R. E. (1999). Extracció de terminologia: elements per a la construcció d'un SEACUSE (Sistema d'Extracció Automática de Candidats a Unitats de Significació Especialitzada. Tese de Doutorado, Universidade Pompeu Fabra. Citado na página 19.

Bagot, R. E., Cabré, M. T., & Vivaldi, J. (1998). Sistemes d'extracció automàtica de (candidats a) termes: estat de la qüestió. 22:1–66. Papers de Institut Universitari de Lingüística Aplicada, Universitat Pompeu Fabra, Sèrie Informes. Citado na página 19.

Banerjee, S. & Pedersen, T. (2003). The design, implementation, and use of the Ngram Statistic Package. In *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*, pag. 370–381, Cidade do México, México. Citado na página 26.

Chang, C.-H., Kayed, M., Girgis, M. R., & Shaalan, K. F. (2006). A survey of web information extraction systems. *IEEE Transactions on Knowledge and Data Engineering*, 18(10):1411–1428. Citado na página 11.

Chen, H., Fuller, S. S., Friedman, C., & Hersh, W. (2005). *Medical Informatics: Knowledge Management and Data Mining in Biomedicine (Operations Research/Computer Science Interfaces)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA. Citado na página 17.

Church, K. W. & Hanks, P. (1989). Word association norms, mutual information, and lexicography. In *Proceedings of the 27th. Annual Meeting of the Association for Computational Linguistics*, pag. 76–83, Vancouver, Canadá. Citado na página 20.

Cimino, J. J. & Shortliffe, E. H. (2006). *Biomedical Informatics: Computer Applications in Health Care and Biomedicine (Health Informatics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA. Citado nas páginas 12, 13, e 16.

Damerau, F. J. (1964). A technique for computer detection and correction of spelling errors. *Commun. ACM*, 7(3):171–176. Citado na página 40.

Dubuc, R. (1999). *Manual de Terminologia*. União Latina/RIL Editores, Santiago, Chile. Citado na página 19.

Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74. Citado na página 20.

Frantzi, K. & Ananiadou, S. (1997). Automatic term recognition using contextual cues. In *Proceedings of 3rd DELOS Workshop*, pag. 25–32, Zurique, Suíca. Citado na página 18.

Gansner, E. R. & North, S. C. (2000). An open graph visualization system and its applications to software engineering. *Software — Practice and Experience*, 30(11):1203–1233. Citado na página 29.

Honorato, D. D. F. (2008). Metodologia de transformação de laudos médicos não estruturados e estruturados em uma representação atributo-valor. Dissertação de Mestrado (a ser defendida), ICMC-USP. Citado nas páginas 17 e 65.

Honorato, D. D. F., Lee, H. D., Monard, M. C., Wu, F. C., Machado, R. B., Neto, A. P., & Ferrero, C. A. (2005). Uma metodologia para auxiliar no processo de construção de bases de dados. In *Anais do V Encontro Nacional de Inteligência, XXV Congresso da Sociedade Brasileira de Computação*, pag. 593–601, Rio Grande do Sul, Brasil. Citado na página 17.

Honorato, D. D. F. & Monard, M. C. (2008). Descrição do ambiente computacional TP-DISCOVER para mapear informações não estruturadas em uma tabela atributo-valor. Relatório técnico 318, ICMC-USP. Citado nas páginas 17 e 65.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. Relatório técnico 8. Citado na página 41.

Pantel, P. & Lin, D. (2001). A statistical corpus-based term extractor. In *Proceedings of the 14th Biennial Conference of the Canadian Society on Computational Studies of Intelligence: Advances in Artificial Intelligence*, pag. 36–46, Londres, Reino Unido. Citado na página 19.

Pardo, T. A. S. (2006). Senter: Um segmentador sentencial automático para o português do brasil. Relatório técnico, ICMC-USP, São Carlos, SP. Citado na página 23.

Patry, A. & Langlais, P. (2005). Corpus-based terminology extraction. In *7th International Conference on Terminology and Knowledge Engineering*, pag. 313–321, Copenhague, Dinamarca. Citado na página 20.

Paulo, J. L., Correia, M., Mamede, N. J., & Hagège, C. (2002). Using morphological, syntactical, and statistical information for automatic term acquisition. In *PorTAL '02: Proceedings of the Third International Conference on Advances in Natural Language Processing*, pag. 219–228, Londres, Reino Unido. Citado na página 19.

Pavel, S. & Nolet, D. (2002). *Handbook of terminology*. Minister of Public Works and Government Services Canada, Québec, Canadá. `http://www.bureaudelatraduction.gc.ca/pwgsc_internet/en/publications/documents/handbook.pdf`. Citado na página 18.

Pazienza, M. T., Pennacchiotti, M., & Zanzotto, F. M. (2005). Terminology extraction: An analysis of linguistic and statistical approaches. In *Proceedings of the Nemis 2004 Final Conference*, volume 185 of *Studies in Fuzziness and Soft Computing*, pag. 255–279, Atenas, Grécia. Citado na página 20.

Ratnaparkhi, A. (1996). A maximum entropy model for part-of-speech tagging. In Brill, E. & Church, K., editors, *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pag. 133–142. Nova Jérsei, EUA. Citado na página 25.

Sager, J. C., Dungworth, & F., M. P. (1980). English Special Languages - Principles and Practice in Science and Technology. *Oscar Brandstetter Verlag KG*. Citado na página 18.

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, Manchester, UK. Citado na página 28.

Schwartz, R., Christiansen, T., & Pyle, L. W. (1997). *Learning Perl*. California. Citado na página 22.

Sebastiani, F. (2002). Machine learning in automated text categorisation. *ACM Computing Surveys*, 34(1):1–47. Citado na página 11.

Smadja, F. (1993). Retrieving collocations from text: Xtract. *Comput. Linguist.*, 19(1):143–177. Citado na página 20.

Teline, M. F., de B. Almeida, G. M., & Aluísio, S. M. (2003). Extração manual e automática de terminologia: comparando abordagens e critérios. In *Proceedings of the 16th Brazilian Symposium on Computer Graphics and Image Processing*, volume 1, pag. 1–12, São Carlos, Brasil. Citado na página 50.

Weiss, S. M., Indurkhya, N., & Zhang, T. u. (2005). *Text mining : predictive methods for analyzing unstructured information;*. Springer, New York, NY. Citado na página 10.

Ziviani, N. (2007). *Projeto de Algoritmos com implementações em Java e C++*. Thomson, São Paulo, SP. Citado na página 32.