UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO
ESCOLA DE INFORMÁTICA APLICADA
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO

RECONHECIMENTO DE ENTIDADES MENCIONADAS UTILIZANDO UMA ABORDAGEM DE *HIDDEN MARKOV MODEL*

PEDRO NUNO DE SOUZA MOURA

PROF. DR. ALEXANDRE ALBINO ANDREATTA

FEVEREIRO DE 2009

RECONHECIMENTO DE ENTIDADES MENCIONADAS UTILIZANDO UMA ABORDAGEM DE *HIDDEN MARKOV MODEL*

Projeto de Graduação apresentado à Escola de Informática Aplicada da Universidade Federal do Estado do Rio de Janeiro (UNIRIO) para obtenção do título de Bacharel em Sistemas de Informação.

PEDRO NUNO DE SOUZA MOURA

PROF. DR. ALEXANDRE ALBINO ANDREATTA

RECONHECIMENTO DE ENTIDADES MENCIONADAS UTILIZANDO UMA ABORDAGEM DE *HIDDEN MARKOV MODEL*

Aprovado em ____/_____/_____

BANCA EXAMINADORA

_____

Prof. Alexandre Albino Andreatta – D. Sc. (orientador)
Universidade Federal do Estado do Rio de Janeiro

_____

Profa. Adriana Cesário de Faria Alvim – D. Sc.
Universidade Federal do Estado do Rio de Janeiro

_____

Prof. Sean Wolfgand Matsui Siqueira – D. Sc.
Universidade Federal do Estado do Rio de Janeiro

Rio de Janeiro, _____de_____ de_____.

_____

Pedro Nuno de Souza Moura

DEDICATÓRIA

À minha avó Carmen, pelo amor e apoio concedidos e pela companhia ao longo de minhas horas de estudo.

AGRADECIMENTOS

A meus pais, Mário e Fernanda, ao meu irmão, Fernando Henrique, e à minha tia Conceição, pelo apoio e incentivo ao longo dos últimos quatro anos, me impelindo a superar desafios e alcançar os objetivos.

A meus amigos de faculdade André, George e Marlon, pelos momentos de estudo e descontração compartilhados desde o primeiro período.

Ao Prof. Alexandre Andreatta, orientador deste trabalho, pelos ensinamentos desde as primeiras aulas na faculdade, assim como os conselhos dados ao longo de minha formação e na confecção deste trabalho.

Aos Profs. Luiz Amâncio e Luiz Pedro Jutuca, do Departamento de Matemática e Estatística, pela confiança e oportunidade nos projetos de monitoria e iniciação científica.

À Profa. Simone, pelos incentivos e encaminhamentos para o início de uma trajetória acadêmica.

Aos professores e funcionários do Departamento de Informática Aplicada de forma geral.

Ao mestrando Eduardo Motta pelas conversas iniciais que originaram este trabalho.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE SIGLAS E ABREVIATURAS

DARPA	– *Defense Advanced Research Project Agency*
HAREM	– Avaliação de sistemas de Reconhecimento de Entidades Mencionadas
HMM	– *Hidden Markov Model*
LEARN	– Laboratório de Engenharia de Algoritmos e Redes Neurais
MET	– *Multilingual Entity Task*
MUC	– *Message Understanding Conference*
NER	– *Named Entity Recognition*
NOSC	– *Naval Ocean System Centre*
OCR	– *Optical Character Recognition*
POS	– *Part-of-Speech*
REM	– Reconhecimento de Entidades Mencionadas
REPENTINO	– REPositório para reconhecimento de ENTidades com NOme
SCFG	– *Stochastic Context-Free Grammar*
SGML	– *Standard Generalized Markup Language*
XML	– *eXtensible Markup Language*

LISTA DE SÍMBOLOS

A	– matriz de probabilidades de transição entre estados
$A[i, j]$	– probabilidade de transição do estado S_i para o estado S_j
$\overline{A}$	– re-estimação da matriz A
$\overline{A}^{(n)}$	– matriz $\overline{A}$ em relação à n-ésima seqüência de observações
a_{ij}	– probabilidade de transição do estado S_i para o estado S_j
B	– matriz de probabilidades de emissão de símbolos
$B[i, k]$	– probabilidade de emissão do símbolo v_k pelo estado S_i
$\overline{B}$	– re-estimação da matriz B
$\overline{B}^{(n)}$	– matriz $\overline{B}$ em relação à n-ésima seqüência de observações
$C\_START[i]$	– contagem do número de ocorrências em que o estado S_i é o estado inicial
$C\_TRANS[i, j]$	– contagem do número de ocorrências em que o estado S_i é seguido pelo estado S_j
$C[i]$	– contagem do número de ocorrências do estado S_i
$C[i, k]$	– contagem do número de ocorrências em que, estando-se no estado S_i, emitiu-se o símbolo v_k
$O[1 \ldots T]$	– seqüência de observações
$O[t]$	– observação emitida no instante t
$\mathbf{O}$	– conjunto de seqüências de observações
$O^{(n)}$	– n-ésima seqüência de observações
$O(f(n))$	– notação O-Grande que denota uma função assintoticamente dominada por $f(n)$
N	– número de estados
M	– número de símbolos no alfabeto
$P(X = x)$	– probabilidade de a variável aleatória discreta X assumir o valor x
$P(A \mid B)$	– probabilidade de o evento A ocorrer dado que B ocorreu

$P(O[1\ldots T]\mid\lambda)$ – probabilidade de a seqüência de observações ter sido gerada pelo modelo λ

$q[1\ldots T]$ – seqüência de estados

$q[t]$ – estado no instante de tempo t

r – número de seqüências de observações em um conjunto de treinamento

S – conjunto de estados

S_0 – estado inicial

S_i – um estado qualquer

$s[t]$ – estado no instante de tempo t

T – comprimento de uma seqüência de observações

t – instante no tempo

V – alfabeto de símbolos

v_k – um símbolo do alfabeto

$\alpha[t,i]$ – probabilidade do prefixo $O[1\ldots t]$ e do estado S_i ser o estado corrente no instante t

$\beta[t,i]$ – probabilidade do sufixo $O[t+1\ldots T]$, sendo S_i o estado corrente no instante t e o modelo λ

$\gamma[t,i]$ – probabilidade do estado S_i em um instante t, sabendo-se a seqüência de emissões e o modelo λ

$\delta[t,i]$ – maior probabilidade de uma seqüência de estados até o instante t, com $q[t]=i$

$\psi[t,i]$ – estado que maximiza $\delta[t,i]$, para cada t e i

$\xi[t,i,j]$ – probabilidade de se estar no estado S_i no instante t e em S_j no instante $t+1$

ϕ – ponderação utilizada na medida *F-Measure*

π – vetor de probabilidade de transição do estado inicial para os demais

$\overline{\pi}$ – re-estimação do vetor π

$\overline{\pi}^{(n)}$ – vetor $\overline{\pi}$ em relação à n-ésima seqüência de observações

λ – um modelo de *HMM*

$\overline{\lambda}$ – re-estimação do modelo λ

Δ – distância entre o modelo λ e sua re-estimação $\overline{\lambda}$

ε – valor positivo arbitrariamente pequeno

RESUMO

Com a disseminação do computador pessoal e a consolidação da internet, a disponibilidade de acesso a textos e documentos nas mais variadas formas tornou-se obíqua. A capacidade de manipulação destas informações textuais por seres humanos é muito menor do que a quantidade a que se tem alcance. Mostra-se, portanto, imperativo o estudo de mecanismos que facilitem e automatizem a obtenção de conhecimento a partir deste volume de informações. A atividade de Reconhecimento de Entidades Mencionadas – REM – visa à obtenção de entidades, tais como nomes de pessoas, organizações e locais, assim como expressões numéricas e temporais. Esta atividade envolve o uso de modelos baseados em regras ou, alternativamente, modelos estatísticos. Este trabalho apresenta uma formalização de um modelo estatístico, *Hidden Markov Model*, e a implementação deste modelo na atividade REM. Foram realizados testes objetivando a avaliação das implementações realizadas e os resultados foram analisados.

Palavras-chave: reconhecimento de entidades mencionadas, *hidden markov model*, mineração em textos.

ABSTRACT

With the spread of personal computers and the consolidation of the internet, the availability of access to texts and documents in varied forms has been made ubiquitous. The capacity of manipulation of this textual information by human beings is much less than the quantity one is able to reach. Therefore, it is showed imperative the importance of studying mechanisms that can facilitate and automate the knowledge acquisition from texts. The activity of Named Entity Recognition – NER – seeks the collection of entities, such as names of people, organization and places, as well as numerical and temporal expressions. This activity includes the use of models based on rules or, alternatively, statistical ones. This work presents a formalization of a statistical model, called Hidden Markov Model, and an implementation of this one in the NER activity. Tests were made aiming to evaluate the implementations done and the results were analyzed.

Keywords: named entity recognition, hidden markov model, text mining.

Capítulo 1 - Introdução

1.1. Motivação

Com o passar dos séculos, a informação foi assumindo cada vez mais um papel preponderante nas organizações e na sociedade como um todo. Nesse contexto, se encaixa a origem da informática, cuja denominação em português provém do francês *informatique* (contração dos sintagmas *information* e *automatique*). Percebe-se, pois, que, desde o início, a computação foi orientada ao processamento e à manipulação de dados, a fim de gerar informação.

Nas últimas décadas, com a disseminação do computador pessoal, a criação e o acesso a arquivos e documentos de cunho geral (isto é, que contêm algum tipo de informação), se tornaram maiores e mais fáceis. Acrescente-se a isso o surgimento da *Web* no início da década de 90, possibilitando a disponibilização e o acesso a dados das mais variadas formas. Ademais, a própria criação de dispositivos como *scanners* e de sistemas de reconhecimento de caracteres (sistemas *OCRs*) apresentaram um papel crucial na questão, na medida em que permitiram a digitalização de livros e documentos outrora unicamente tangíveis.

Neste sentido, ao se levar em consideração a limitada capacidade de processamento e armazenamento de informações do cérebro humano, percebe-se que a quantidade de informações a que se tem alcance é exponencialmente maior do que um indivíduo, ou mesmo um grupo de indivíduos, pode manipular. Faz-se, pois, imperioso, o estudo e a criação de dispositivos que facilitem e automatizem o processo de extração de informações e síntese de textos.

À primeira vista, mostra-se uma boa opção utilizar as técnicas preconizadas pela *Mineração de Dados*, que estuda a aplicação de técnicas para extração e inferência de conhecimento a partir de dados armazenados em repositórios. Contudo, após uma análise mais minuciosa, faz-se notória uma distinção crucial: os arquivos e documentos que se desejam tratar estão tipicamente em formatos semi-estruturados ou não-estruturados, ou seja, em formatações limitadas ou em texto livre, enquanto que a informação contida por um banco de dados segue uma estruturação, um formato rígido. Estima-se que 80% da informação de uma organização se encontre na

forma de textos desestruturados: relatórios, memorandos, e-mails, documentos gerais e etc (WEN, 2001 e KONCHADY, 2006).

Sob esse prisma, surge a *Mineração em Textos*, um desdobramento da própria Mineração de Dados, mas que conjuga forte interseção com as áreas de *Aprendizado de Máquina*, *Processamento de Linguagem Natural*, *Recuperação da Informação* e *Gestão do Conhecimento* (FELDMAN; SANGER, 2007). Além da obtenção de padrões, a Mineração em Textos tem também como objetivo extrair conhecimento e relações implícitas incutidas em documentos.

As técnicas de Mineração em Textos podem ser subdivididas em quatro categorias (PRADO; FERNEDA, 2008):

1) *Classificação*: consiste em alocar objetos em classes predefinidas;
2) *Análise de Relações*: estabelece conexões entre os conceitos presentes no texto;
3) *Extração da Informação*: ocupa-se na obtenção de dados e expressões relevantes; e
4) *Clusterização*: visa à descoberta de estruturas subjacentes ao conjunto de documentos.

1.2. Problema

Uma atividade pertencente à Extração da Informação é a denominada de *NER* (*Named Entity Recognition* – Reconhecimento de Entidades Mencionadas), surgida durante o sexto *MUC* (*Message Understanding Conference*), que eram conferências realizadas nos anos 1980 e 1990 pela *NOSC* (*Naval Ocean System Centre*) com o suporte da *DARPA* (*Defense Advanced Research Project Agency*), reunindo pesquisadores da área de Extração da Informação, com foco em textos não-estruturados.

A ênfase de tais conferências era inicialmente problemas relativos a documentos militares. Aos poucos, o enfoque migrou para temáticas civis. Ademais, antes de sistemas *NER* serem reconhecidos oficialmente em 1996, pesquisas eram conduzidas no sentido de extrair apenas nomes próprios a partir de textos (NADEAU, 2007).

A atividade de *NER* caracteriza-se por conter três subtarefas: o reconhecimento de nomes de entidades, expressões temporais e expressões numéricas. Essas expressões podem ser identificadores únicos de entidades (organizações, pessoas e localizações), tempo (datas, horas) ou quantidades (valores monetários e porcentagens) (GRISHMAN; SUNDHEIM, 1996). Por identificador único entende-se uma expressão que possui um único significado nos diversos contextos em que existe, não possuindo sentido em qualquer outro contexto em que apareça.

Apesar de a definição preconizar expressões sem ambigüidade em relação ao contexto (identificador único), desde o início, pesquisadores têm levado em consideração o problema com ambigüidade, particularmente pelo seu desafio e dificuldade. Além disso, as expressões mais pesquisadas e que possuem maior ênfase são os nomes de entidades (organizações, pessoas e localizações), que freqüentemente apresentam ambigüidades.

A identificação e o reconhecimento das entidades são simbolizados através de etiquetas (anotações) afixadas no próprio texto de origem, tal qual demonstrado em (GRISHMAN; SUNDHEIM, 1996), em que *ENAMEX* designa nomes de entidades e *NUMEX* referencia expressões numéricas:

Mr. <ENAMEX TYPE="PERSON"> Dooner </ENAMEX> met with <ENAMEX TYPE="PERSON"> Martin Puris </ENAMEX>, president and chief executive officer of <ENAMEX TYPE="ORGANIZATION"> Ammirati & Puris </ENAMEX>, about <ENAMEX TYPE="ORGANIZATION"> McCann </ENAMEX>'s acquiring the agency with billing of <NUMEX TYPE="MONEY"> $400 million </NUMEX>, but nothing materialized.

Há tipicamente duas abordagens para *NER*: aquela baseada em gramáticas e outra baseada em modelos estatísticos. A primeira exige o envolvimento de um número relativamente grande de lingüistas experientes, implicando em alto custo. Essa vertente geralmente obtém resultados melhores do que a de modelos estatísticos, mas exige a disponibilidade de textos a serem utilizados no treinamento e, dependendo do tipo de aprendizado empregado, algumas intervenções no processo. Contudo, considerando-se a relação custo-benefício, esta vertente é mais atrativa de ser explorada.

Dentro da abordagem estatística, as técnicas de aprendizado se subdividem em três: aprendizado supervisionado, semi-supervisionado e não-supervisionado. A

técnica dominante para abordar o problema em questão é a de aprendizado supervisionado (NADEAU, 2007). Nesta categoria, estão incluídas as abordagens de *Hidden Markov Models* (*HMM*), Árvores de Decisão, *Support Vector Machine* e *Maximum Entropy Model* (WEN, 2001).

1.3. Objetivos do Trabalho

O objetivo geral deste trabalho é realizar um estudo do problema de *NER* (*Named Entity Recognition*), empregando uma abordagem de *Hidden Markov Models*.

Os objetivos específicos são:

- Apresentar uma formalização, adaptada de (RABINER, 1989), da abordagem *HMM*;
- Implementar um sistema baseado em *HMM* para *NER*; e
- Testar a implementação e analisar os resultados.

Para o teste da implementação, são utilizadas versões já etiquetadas morfossintaticamente dos textos presentes no *HAREM* – Avaliação de sistemas de Reconhecimento de Entidades Mencionadas (SANTOS et al., 2006).

O diagrama da Figura 1.1 resume a metodologia de *NER* adotada. Em um primeiro momento, um texto é analisado morfossintaticamente (*Part-of-Speech Tagging*), em que cada palavra (*token*) é associada a uma dada classe gramatical: substantivo, adjetivo, verbo, advérbio e etc. O processo de etiquetação consiste em colocar, após cada palavra, a sua respectiva classificação, conforme ilustrado no Anexo B.

Após isso, são obtidas as entidades mencionadas do texto, com base na análise morfossintática estabelecida na etapa anterior. Por fim, são apresentadas tais entidades extraídas.

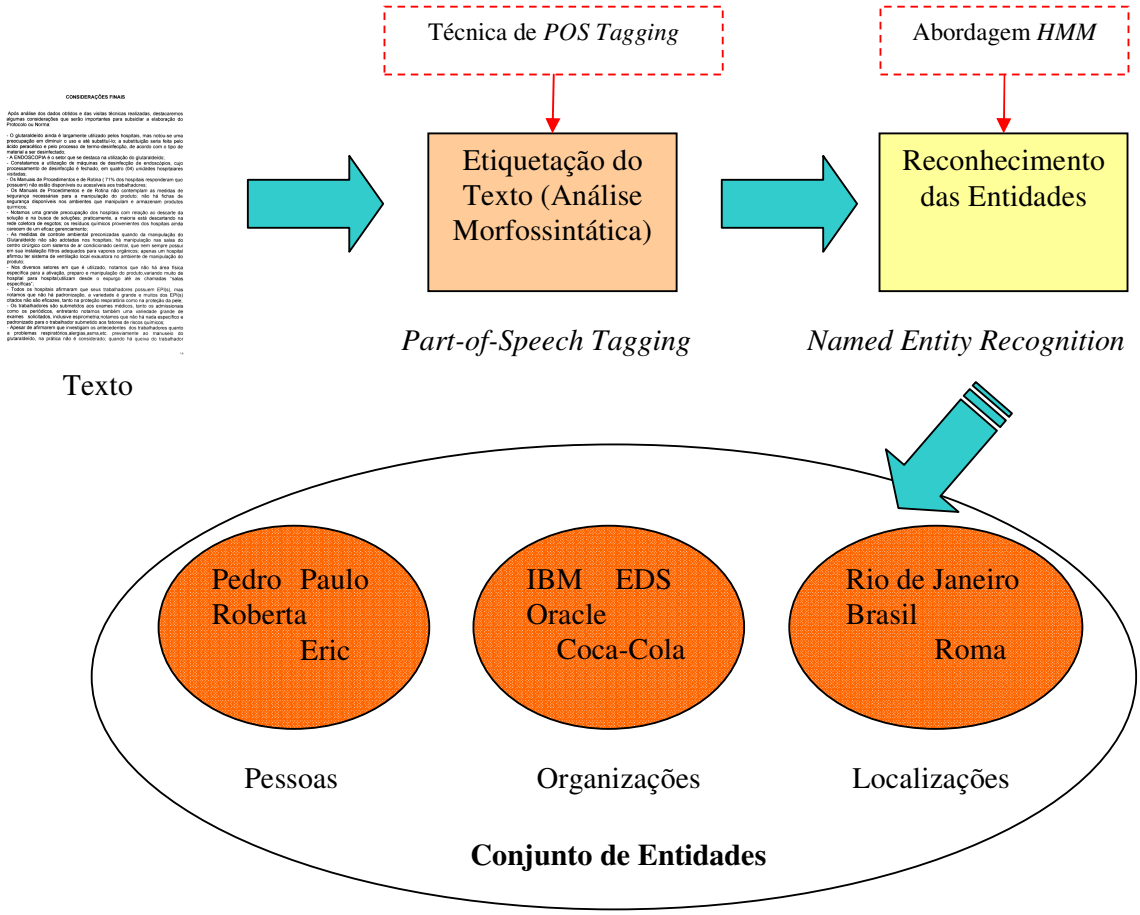

Figura 1.1 – Resumo do Processo de Extração e Classificação.

1.4. Organização do Trabalho

No capítulo 2, apresenta-se uma breve introdução aos conceitos de Mineração em Textos, Extração da Informação e as atividades de extração e classificação de entidades. No capítulo 2, apresentam-se também técnicas comumente utilizadas na atividade de *NER*, que são tipicamente subdivididas em sistemas baseados em regras ou sistemas baseados em aprendizado. O capítulo 3 é dedicado à apresentação do modelo de markov oculto (*HMM*) e dos algoritmos utilizados para resolver os três problemas centrais endereçados pelo modelo.

No capítulo 4, são descritos os experimentos computacionais realizados, os resultados alcançados e a análise dos dados obtidos. Finalmente, no capítulo 5, são apresentados os comentários finais e a conclusão do trabalho.

Capítulo 2 – Mineração em Textos

2.1. Conceituação

A Mineração em Textos pode ser definida como um processo intensivo de descoberta de conhecimento em que um usuário interage com uma coleção de documentos através do tempo, utilizando um conjunto de ferramentas analíticas (FELDMAN; SANGER, 2007). A área de Mineração de Textos inclui a ação de explorar documentos, retirando fragmentos e inferindo conhecimento previamente desconhecido, mas não se restringe a isto. Inclui toda a estrutura, desde a operação mais elementar de manipulação do texto às ferramentas visuais que darão a percepção da informação ao usuário.

Uma arquitetura para um sistema de mineração de textos genérico, no sentido de que não é orientado a nenhum domínio específico, é exibida na Figura 2.1:

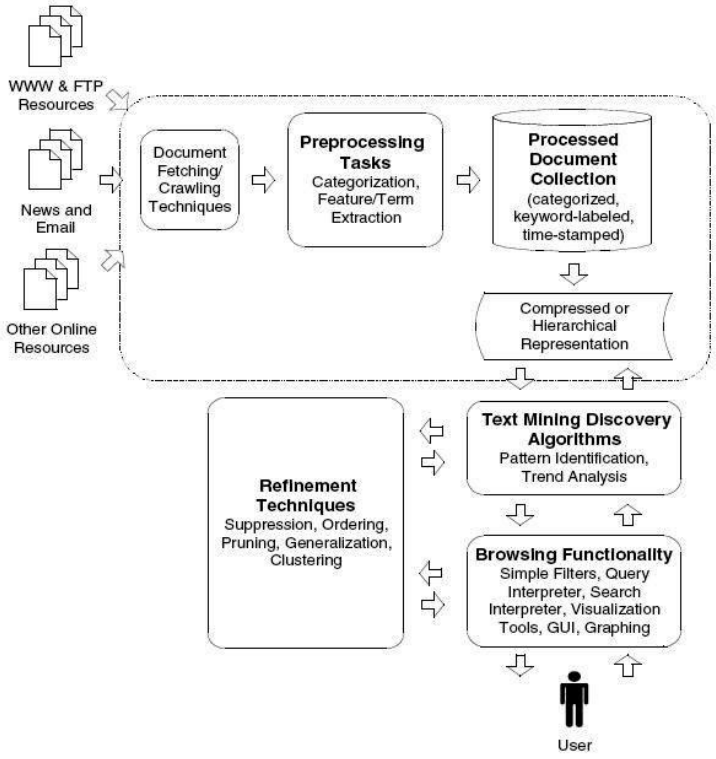

Figura 2.1: Arquitetura de um sistema genérico de Mineração em Textos (FELDMAN; SANGER, 2007).

A primeira etapa é a *Coleta dos Documentos* (*Document Fetching*). O sistema recebe como entrada um conjunto de documentos sem qualquer tipo de tratamento inicial. O sistema pode obtê-los através da *Web* ou explicitamente pelo usuário. Nesse sentido, para obtenção de textos na *Web* são utilizados *crawlers*, mecanismos que percorrem a internet sistematicamente, a fim de obter possíveis fontes de textos.

Os documentos que servem como entrada ao sistema são textos desestruturados ou semi-estruturados. Nesse sentido, cabe fazer uma distinção entre esses dois tipos. Os primeiros não possuem qualquer tipo de estruturação, tais como elementos que indiquem a tipografia ou o leiaute, sendo também denominados de documentos livres de formato. Os textos semi-estruturados possuem elementos de formatação mais consistentes, possibilitando a inferência de metadados, isto é, dados que descrevam os dados contidos no texto (FELDMAN; SANGER, 2007). É o caso de páginas *HTML* e documentos *PDF*.

Cada documento passa por um filtro que o converte em texto simples (*plain text*), sem seguir qualquer linguagem de marcação ou formato proprietário (KONCHADY, 2006). Após isso, os textos são indexados, consistindo em um conjunto de palavras que os representam.

A etapa seguinte corresponde ao *Pré-processamento* (*Preprocessing Tasks*). Esta compreende as rotinas e os métodos utilizados na preparação dos dados, com o objetivo de obter uma estruturação a partir dos textos. O resultado do pré-processamento é um documento dentro de um modelo padronizado de representação. (FELDMAN; SANGER, 2007). Após essa etapa, os documentos são guardados em um repositório (*Processed Document Collection*).

Algumas possibilidades de pré-processamento são *Categorização de Textos*, *Extração da Informação*, *Sumarização* e *Clusterização*. Cada uma é indicada para problemas específicos, devendo-se proceder primeiro à identificação dos objetivos da mineração em textos a ser aplicada (GOMES, 2006).

As *Operações de Mineração* (*Text Mining Discovery Algorithms*) formam o componente central do sistema de mineração de textos. Dentre as principais operações estão a descoberta de padrões, análise de tendências e algoritmos incrementais de descoberta do conhecimento (FELDMAN; SANGER, 2007). Tais

algoritmos são assim chamados na medida em que são aplicados a cada ciclo de interação com o usuário.

A *Camada de Apresentação* (*Browsing Functionality*) inclui a interface com o usuário e as funcionalidades para realização de consulta e exibição de resultados. A informação desejada pelo usuário é exposta de maneira inteligível e passível de interação, podendo-se gerar gráficos e realizar análises. Pode incluir um interpretador para uma linguagem de consulta (FELDMAN; SANGER, 2007). É um processo interativo, em que o usuário visualiza a resposta das consultas realizadas e submete novas requisições.

As *Técnicas de Refinamento* (*Refinement Techniques*) incluem métodos de filtragem de informação redundante e junção de dados relacionados (FELDMAN; SANGER, 2007). Envolvem também ordenação e supressão de alguns dados. É também denominada de *Pós-processamento.*

2.2. Extração da Informação

A *Extração da Informação* é uma disciplina que já existia antes da Mineração em Textos, embora atualmente seja visualizada como uma das técnicas componentes desta. É uma das alternativas para a etapa de pré-processamento de um texto, obtendo entidades, eventos e relacionamento relevantes de um texto desestruturado (FELDMAN; SANGER, 2007). O objetivo é transformar um texto desestruturado em uma forma que possa ser carregada em uma tabela de banco de dados (KONCHADY, 2006), se utilizando de uma série de filtros para obter a informação.

Não há qualquer preocupação em obter um entendimento total do texto ou realizar um *parser* completo, mas sim o reconhecimento de padrões lingüísticos e sua extração (JACKSON; MOULINIER, 2002). Métodos de reconhecimento de padrões e algoritmos de aprendizado de máquina são suficientes para essa atividade (KONCHADY, 2006).

Há quatro tipos de elementos básicos que podem ser extraídos de textos (FELDMAN; SANGER, 2007):

1) *Entidades*: São os conceitos-chave cujas instâncias devem ser extraídas, como por exemplos nomes de pessoas, organizações, locais, carros, drogas, táxons em biologia, etc.
2) *Atributos*: São propriedades das entidades extraídas. Exemplos são a idade de uma pessoa, a cor de um carro, o nicho de mercado de uma organização, o efeito de uma droga, etc.
3) *Fatos*: São os relacionamentos que existem entre entidades. Por exemplo, a posse de um carro por uma pessoa, o relacionamento de emprego entre uma pessoa e uma empresa, etc.
4) *Eventos*: São atividades ou ocorrências em um instante do tempo envolvendo entidades de interesse. Por exemplo, a aquisição de uma organização por outra, a realização de uma patente por uma empresa, o lançamento de um produto, etc.

A Extração da Informação é uma das técnicas preponderantes correntemente utilizadas nas operações de pré-processamento e sem a qual sistemas de Mineração em Textos teriam capacidades muito mais limitadas de descoberta da informação (FELDMAN; SANGER, 2007).

Cabe fazer uma distinção entre as áreas de *Recuperação da Informação* e Extração da Informação. A primeira se ocupa da estrutura de armazenamento e recuperação de documentos relevantes em uma coleção, retornando documentos que se enquadrem em uma dada consulta feita pelo usuário, mas ainda exigindo deste a leitura do texto para obter a informação desejada (KONCHADY, 2006). A segunda já retorna ao usuário a informação desejada em uma forma estruturada (FELDMAN; SANGER, 2007).

A área começou a ganhar relevância nas conferências *MUC*, que eram conferências realizadas pela *NOSC*, objetivando incentivar pesquisas em análise automática de mensagens militares contendo informações textuais. No *MUC-2*, foram definidos os parâmetros de avaliação de sistemas de Extração da Informação, seguindo aquilo que era preconizado pela área de Recuperação da Informação: Abrangência (*Recall*) e Precisão (*Precision*). No *MUC-6*, a atividade de Reconhecimento de Entidades Mencionadas foi definida, abrangendo a identificação

de nomes de pessoas, organizações e localizações. Marcações *SGML - Standard Generalized Markup Language* – foram utilizadas para denotar a classificação de uma entidade. Tal atividade foi posteriormente aprimorada de modo a englobar também tempo, valores monetários e porcentagem. Também foi idealizada a atividade de co-referência, que consiste em indicar qual elemento é substituído por um termo anafórico/catafórico (pronome relativo, pronome demonstrativo e etc.) ou a qual entidade determinada expressão se refere.

Além das conferências *MUC*, cabe mencionar as conferências *MET* – *Multilingual Entity Task* – as primeiras conferências multilíngües de *NER*, que ocorreram entre 1996 e 1998 (CARDOSO, 2006), abordando as línguas inglesa, espanhola, chinesa e japonesa.

A arquitetura de um sistema de Extração da Informação pode ser dividida em cinco etapas (KONCHADY, 2006), conforme ilustrado na Figura 2.2. As primeiras duas etapas são gerais e independentes de domínio, ao passo que as três posteriores são específicas do domínio de aplicação.

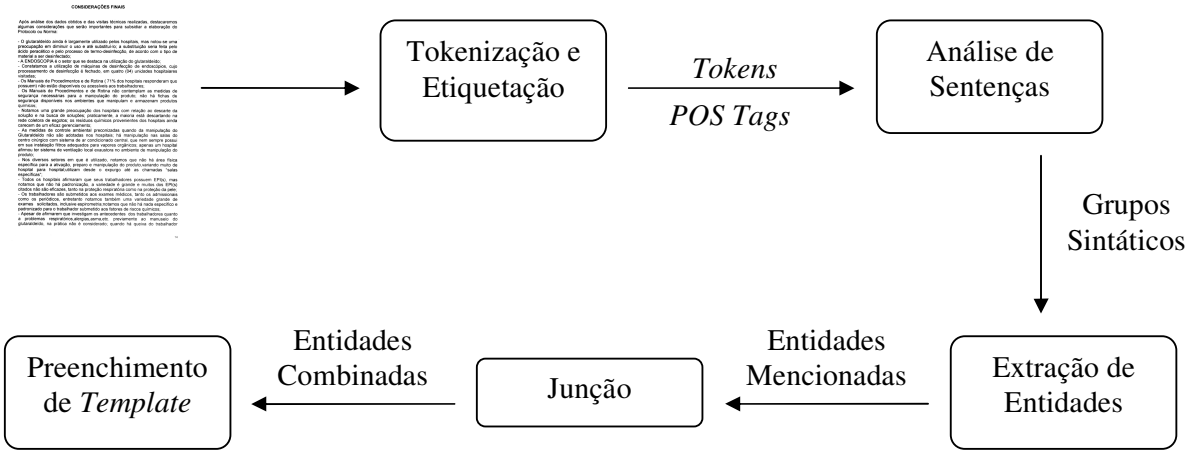

Figura 2.2: Arquitetura de um sistema de Extração da Informação (adaptada de KONCHADY, 2006).

A primeira etapa (*Tokenização e Etiquetação*) consiste na partição do texto em capítulos, ou seções, ou parágrafos, ou sentenças, ou palavras ou até mesmo em sílabas ou fonemas (FELDMAN; SANGER, 2007). Normalmente, o texto é

particionado em sentenças e estas em *tokens*, sendo o processo denominado de *tokenization*. Um *token* é uma unidade de texto, podendo consistir até de mais de uma palavra. Neste trabalho, considera-se que um *token* corresponde a uma palavra, um número, um caractere de pontuação ou um caractere delimitador.

Além disso, na primeira etapa também é realizada a etiquetação morfossintática (*POS Tagging*), em que cada *token* é associado a uma determinada classe gramatical, de acordo com o papel assumido na sentença sendo processada. Etiquetas morfossintáticas fornecem alguma informação quanto ao conteúdo semântico de um *token* (FELDMAN; SANGER, 2007). O menor conjunto de classes morfossintáticas no português corresponde a: artigo, substantivo, adjetivo, numeral, pronome, verbo, advérbio, conjunção, interjeição e preposição, além de uma classe para denotar caracteres de pontuação (KONCHADY, 2006).

Abordagens à atividade de *POS Tagging* seguem aquelas empregadas em Reconhecimento de Entidades Mencionadas e serão apresentadas na seção 2.4.

Ocasionalmente, a etiquetação morfossintática pode buscar informações sobre os radicais (*stems* ou *lemmas*) dos *tokens*, em um processo denominado de *lemmatization* ou *stemming* (FELDMAN; SANGER, 2007). Este procedimento é utilizado para converter palavras derivadas à sua forma base, facilitando, assim, o processo de classificação gramatical.

A etapa de *Análise de Sentenças* consiste na criação de grupos sintáticos, obtendo sujeito, predicado, objeto direto/indireto e etc. Um *parser* superficial (*shallow parsing*) é suficiente para essa tarefa (KONCHADY, 2006). Em vez de prover uma análise completa da sentença inteira, produz apenas partes que são fáceis e não-ambíguas. Seu uso se justifica pelo fato de ser rápido e robusto, se em comparação com um *parser* completo, sendo suficiente para as atividades de Extração da Informação (FELDMAN; SANGER, 2007).

A *Extração de Entidades* busca extrair as entidades mencionadas específicas do domínio de aplicação do sistema. Por exemplo, em um domínio farmacêutico, possíveis entidades envolveriam drogas, patentes e nomes de empresas. As informações obtidas são posteriormente utilizadas para preencher os campos de um dado *template* pré-concebido para o domínio (KONCHADY, 2006).

A etapa de *Junção* realiza a atividade de co-referência, estabelecendo um elo entre todas as referências para uma mesma entidade, já que esta pode ser identificada de diversas formas em um texto. Por exemplo, a empresa *Microsoft* pode ser referenciada como *MS* apenas, ou ainda como a *dona do Windows* ou a *empresa de Bill Gates*, além de quaisquer outros pronomes que a ela se refiram. As classes gramaticais que podem exercer o papel de elemento anafórico (refere-se a um antecedente) ou catafórico (refere-se a uma expressão posterior) são: pronome relativo, pronome demonstrativo, pronome pessoal do caso reto e pronome pessoal oblíquo.

Essa etapa provê uma forma canônica de representação de uma mesma entidade à qual todas as referências estarão conectadas (KONCHADY, 2006).

A última etapa – *Preenchimento de Template* – se concentra na combinação das informações coletadas nos componentes anteriores e no preenchimento de um *template* específico do domínio, descrevendo o relacionamento entre as entidades (FELDMAN; SANGER, 2007). Normalmente, se resume em uma estrutura tabular a ser preenchida. Detalhes e restrições do domínio são aplicados neste momento.

Como a arquitetura segue uma abordagem seqüencial entre as etapas, qualquer erro cometido no início se propaga para as etapas posteriores, levando a um desempenho ruim do sistema (KONCHADY, 2006).

2.3. Reconhecimento de Entidades Mencionadas

A atividade de Reconhecimento de Entidades Mencionadas é o cerne da Extração da Informação. De maneira geral, as classes comumente utilizadas são pessoa, local, organização, expressões de tempo e expressões numéricas. Entretanto, em sistemas orientados a um domínio de aplicação específico, outros conceitos podem ser extraídos.

NER é fracamente dependente do domínio de aplicação, no sentido de que se pode ou não degradar a performance de um sistema ao se mudar o domínio dos textos analisados. Isso dependerá do nível de generalização que o sistema obtiver e da similaridade dos dois domínios (FELDMAN; SANGER, 2007).

Entidades mencionadas possuem forte correlação com nomes próprios, ou seja, substantivos que comecem com letras maiúsculas. Entretanto, isso não significa que, de fato, seja uma entidade, haja vista que as palavras que iniciam frases possuem a primeira letra maiúscula (JACKSON; MOULINIER, 2002).

Algumas abordagens são baseadas no uso de um ***gazetteer***, ou seja, uma lista ou base de entidades contendo nomes de pessoas, organizações, localizações, moedas ou outras (MIKHEEV; MOENS; GROVER, 1999). Um *gazetteer* normalmente facilita bastante o trabalho classificação de entidades, mas possui alguns problemas: dificuldade de manter a lista atualizada e o tratamento para palavras que apareçam em mais de uma categoria, por exemplo, *Ford*, pessoa e *Ford*, organização (KONCHADY, 2006).

A atividade de extração e classificação de entidades pode ser empregada visando vários objetivos, como por exemplo:

1) Monitoramento da freqüência e rastreamento da ocorrência de determinadas entidades ou eventos em notícias (KONCHADY, 2006);
2) Obtenção de sentenças que contenham determinadas entidades a partir de consultas feitas pelo usuário (KONCHADY, 2006);
3) Recuperação de informação semântica, retornando, por exemplo, uma lista de elementos quando a consulta é feita por uma categoria de entidade (NADEAU, 2007);
4) Auxílio em mecanismos de tradução automática, para evitar a tradução de palavras que, de fato, correspondam a um nome próprio (WEN, 2001); e
5) Preenchimento de uma ontologia a partir das entidades obtidas de documentos de um dado domínio (CIMIANO, 2006).

2.4. Abordagens ao Reconhecimento de Entidades Mencionadas

2.4.1. Modelos Baseados em Regras

Estes modelos são obtidos através de uma análise da língua sendo manipulada. Em outras palavras, são vistas as propriedades das palavras, tais como classificações morfológicas, sintáticas e características ortográficas, sendo criadas

regras heurísticas, comumente denotadas através de expressões regulares. Caso uma palavra se encaixe na regra delineada, obtém-se imediatamente a sua classificação. Exemplos de regras seriam:

- "*Se uma palavra é precedida pela preposição 'em' e começa com letra maiúscula, então é um Local"*; e
- "*Se uma palavra é precedida por um pronome pessoal de tratamento e começa com letra maiúscula, então é uma Pessoa"*.

Abordagens iniciais a *NER* eram baseadas em regras, ao passo que recentemente vem migrando para modelos estocásticos. Ainda assim, quando conjuntos de treinamento não estão disponíveis, modelos baseados em regras continuam sendo a abordagem preferida (NADEAU, 2007).

Normalmente, possuem grande grau de acerto, mas são orientadas ao domínio para o qual foram construídas, não podendo ser aproveitadas para outros contextos (WEN, 2001). Além disso, envolvem um grande esforço humano na sua confecção.

2.4.2. Modelos Estatísticos

São aqueles que, a partir de exemplos de treinamento, inferem automaticamente características e parâmetros utilizados na extração e classificação das entidades. Devem ser capazes de classificar entidades antes desconhecidas. Passam, portanto, por uma etapa de treinamento ou aprendizado.

Possuem um menor esforço de confecção quando comparados à abordagem baseada em regras. Além disso, podem ser utilizados em outros contextos ou domínios, exigindo apenas pequenas modificações no código, sendo também menos dependentes da língua a que são aplicados (WEN, 2001).

Subdividem-se em três tipos, de acordo com a quantidade de intervenção necessária: aprendizado supervisionado, semi-supervisionado e não-supervisionado.

I) Aprendizado Supervisionado

O aprendizado supervisionado é aquele que se utiliza de um conjunto de exemplo de entidades mencionadas em um texto previamente etiquetado, a fim de

obter uma capacidade de generalização. Deve haver tantos dados quantos forem necessários para exemplificar o uso de todas as etiquetas (KONCHADY, 2006). Tipicamente, exigem um grande esforço na obtenção de tal conjunto de exemplo, o que muitas vezes acabe sendo um fator proibitivo para seu uso.

A performance dessa abordagem depende do vocabulário de transferência, que é a proporção de palavras, sem repetição, que aparecem tanto no conjunto de treinamento quanto no de avaliação. Além disso, essa proporção corresponde a um bom indicador para a medida de avaliação de Abrangência (*Recall*) (NADEAU, 2007).

Bikel (BIKEL et al., 1997) descreve um sistema que utiliza *HMM* baseado em propriedades léxicas (por exemplo, se a primeira letra de uma palavra é maiúscula, se é toda maiúscula, se contém dígitos e etc.) para realizar a atividade de *NER*. O treinamento é realizado a partir do conjunto de textos anotados e utilizado no *MUC-6*.

Wen (WEN, 2001) apresenta um sistema que utiliza *HMM* em que os estados correspondem às classes de entidades mencionadas e as observações correspondem às palavras. O número de emissões do *HMM* corresponde ao número de palavras diferentes encontradas nos dados de treinamento. Técnicas de suavização são utilizadas para ajustar a probabilidade de determinados estados e emissões. A ênfase do sistema residiu na utilização de uma técnica para tratar palavras desconhecidas, ou seja, que não tenham aparecido no conjunto de treinamento.

II) Aprendizado Semi-Supervisionado

Esta abordagem, também denominada de fracamente supervisionada, envolve alguma intervenção. O treinamento normalmente utiliza uma mistura de texto etiquetado com texto não-etiquetado (KONCHADY, 2006). A principal corrente envolve o emprego de uma técnica denominada *bootstrapping*. Essa consiste no fornecimento, pelo usuário, de um pequeno conjunto de palavras ou regras que sirvam como os exemplos iniciais (FELDMAN; SANGER, 2007). Tais exemplos também são chamados de sementes (*seeds*).

A partir disso, o sistema tenta generalizar, buscando sentenças em algum outro sistema ou repositório (tipicamente, na Internet) que contenham as entidades e tentando identificar padrões no contexto entre estas. Após isso, tenta, então, obter novos nomes com os contextos identificados, reiniciando o processo.

Nadeau (NADEAU, 2007) apresentou um sistema baseado nessa abordagem e obteve resultados comparáveis aos de sistemas cuja ênfase está sobre o aprendizado supervisionado. Essencialmente, a única interação exigida é a escolha e o fornecimento das sementes iniciais.

III) Aprendizado Não-Supervisionado

No aprendizado não-supervisionado, não é exigido qualquer tipo de interação no sentido de marcação manual do texto, o que acaba sendo uma opção de menos esforço quando comparada a uma abordagem de aprendizado supervisionado. Assim sendo, o treinamento é realizado a partir de um conjunto de dados não-etiquetado. Normalmente, em contextos ambíguos, essa abordagem não possui um bom desempenho (KONCHADY, 2006).

Algoritmos baseados nessa abordagem procuram por estruturas freqüentes e padrões no conjunto de dados, sendo normalmente utilizados para análise exploratória (CIMIANO, 2006). Costumam envolver o uso de classificadores, que separam os dados em partições, isto é, conjuntos de classes distintas. Um exemplo é a abordagem de clusterização, que separa os dados em grupos cujos elementos são muito próximos quanto a uma dada medida adotada.

Em *NER*, pode-se utilizar clusterização para obter as entidades mencionadas a partir de grupos estabelecidos quanto à similaridade de contexto. Pode-se também utilizar padrões léxicos ou estatísticas obtidas a partir dos textos (NADEAU, 2007).

Nadeau (NADEAU, 2007) descreve um sistema que se utiliza de uma observação de que entidades mencionadas geralmente aparecem de maneira sincronizada em notícias, enquanto nomes comuns não possuem essa característica. Foi então explorado o fato de que há uma forte correlação entre um nome ser entidade mencionada e aparecer pontual e simultaneamente em diversos jornais. O

sistema trabalha de maneira não-supervisionada ao fazer a exploração das diversas fontes de notícias e achar as interseções capazes de serem entidades.

2.4.3. Modelos Híbridos

Um modelo híbrido corresponde a uma combinação da abordagem baseada em regras com algum modelo estatístico, na tentativa de conjugar as vantagens de ambos e suprimir as deficiências. Um tal modelo tenta se utilizar da grande precisão de modelos baseados em regras, enquanto procura reduzir (normalmente de forma drástica) a sua quantidade intrínseca de esforço manual, pelo emprego de estatísticas obtidas a partir do conjunto de treinamento (FELDMAN; SANGER, 2007). A Figura 2.3 apresenta a estrutura de um modelo híbrido, em que primeiramente há uma etapa de aplicação de regras e posteriormente é feita a análise estatística:

Figura 2.3: Modelo Híbrido (MEDEIROS, 2008).

Um exemplo de modelo híbrido é uma *Gramática Livre de Contexto Estocástica* (*Stochastic Context-Free Grammar – SCFG*), que corresponde a uma Gramática Livre de Contexto da Hierarquia de Chomsky modificada para que às regras sejam associadas probabilidades de aplicação. Assim sendo, as regras são obtidas manualmente, enquanto as probabilidades são obtidas através de treinamento a partir de um conjunto de dados. As regras devem ser simples, para limitar a quantidade de trabalho manual, e o tamanho do conjunto de treinamento é, normalmente, bem menor do que o utilizado para sistemas puramente estatísticos (FELDMAN; SANGER, 2007).

Capítulo 3 – *Hidden Markov Models*

Este capítulo tem como objetivo realizar uma formalização da teoria de *Hidden Markov Models*, acrescentando aspectos referentes a seu uso em processamento da linguagem natural. Está baseado, principalmente, no texto de Lawrence Rabiner (RABINER, 1989).

Os **Modelos Markovianos** modelam processos que tipicamente emitem sinais observáveis. Tais sinais podem ser discretos (símbolos de um dado alfabeto, palavras de um texto, números inteiros, etc) ou contínuos (fala, temperatura, ondas, etc). Sob determinadas condições, pode ser que os sinais sofram algum tipo de corrupção, isto é, atuação de um ruído, podendo ser caracterizados como puros ou corrompidos. Pode-se, ainda, levar em consideração a variação no tempo: estacionários ou não-estacionários.

Originalmente estudados por Andrei A. Markov na modelagem de seqüências de letras na literatura russa (KONCHADY, 2006), essa abordagem tem se tornado uma ferramenta estatística popular para tratamento de uma ampla classe de problemas, que inclui o mapeamento do genoma humano, o mapeamento de decisões de consumidores e o reconhecimento da fala (KONCHADY, 2006).

Uma cadeia de Markov corresponde a um sistema com um conjunto de N estados distintos (variáveis aleatórias), $S = \{S_1, S_2, \ldots, S_N\}$, e que, em um dado instante do tempo t (t assumindo valores discretos) está em um estado $s[t] \in S$. Uma tal cadeia é representada através de um grafo direcionado, no qual os vértices correspondem aos estados e as arestas à transição entre estes. Um exemplo de cadeia de markov é ilustrado na Figura 3.1.

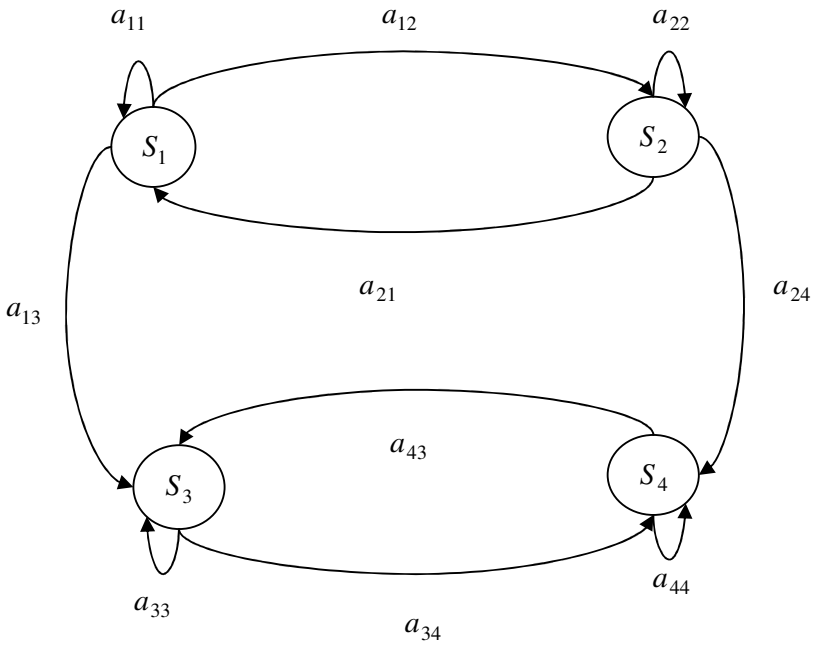

Figura 3.1 – Exemplo de cadeia de Markov.

A transição do estado S_i para o estado S_j ocorre sob uma probabilidade $a_{ij} = P(s[t] = S_j \mid s[t-1] = S_i)$. A ausência de arestas ligando dois estados indica probabilidade nula. Todas as transições a partir de um estado constituem uma medida de probabilidade e, portanto, $\sum_{j=1}^{N} a_{ij} = 1, \quad 1 \le i \le N$.

Uma cadeia de transições de estados é markoviana se a probabilidade de se estar em um determinado estado após uma seqüência de transições depende apenas do predecessor. Em suma:

$$P(s[t] = S_{i\_t} \mid s[t-1] = S_{i\_(t-1)}, s[t-2] = S_{i\_(t-2)}, \ldots, s[1] = S_{i\_1}) = P(s[t] = S_{i\_t} \mid s[t-1] = S_{i\_(t-1)}) \quad (3.1)$$

Diz-se também que tal cadeia é uma cadeia de primeira ordem, pois a dependência ocorre apenas do estado imediatamente anterior. Em uma cadeia de segunda ordem, a dependência ocorre em relação aos dois estados anteriores. Uma cadeia é de ordem n quando o estado corrente é dependente dos n estados anteriores. Um exemplo de aplicação de cadeias de Markov de ordens superiores a

um é a modelagem de emissão de seqüências de palavras, na medida em que a dependência entre estas em uma frase não ocorre apenas entre as adjacentes.

No contexto da lingüística computacional, diz-se que uma cadeia de markov de primeira ordem corresponde a um **bigrama** e uma de segunda ordem a um **trigrama**. De forma geral, um modelo **n-grama** usa $(n-1)$ etiquetas anteriores para definir a próxima (SCHÜTZE; MANNING, 1999). O termo n-grama também é utilizado para denotar as n últimas etiquetas processadas.

O modelo markoviano apresentado é um modelo visível (*Visible Markov Model*), onde cada estado corresponde a um evento observável. Entretanto, modelos visíveis têm limitações para a modelagem de problemas, não sendo adequados para muitos processos mais complexos (RABINER, 1989). Tais problemas podem ser abordados por *modelos markovianos ocultos* (*Hidden Markov Models - HMM*), em que não se têm informações diretas sobre os estados, dispondo-se apenas de acesso às emissões por eles realizadas, como ilustrado na Figura 3.2. Uma aresta partindo de um estado S_i para uma emissão observável v_k representa uma probabilidade não-nula de S_i emitir v_k.

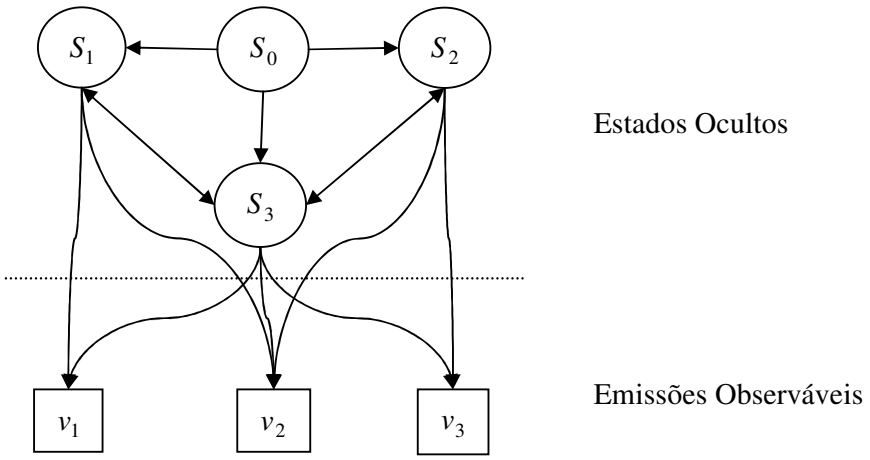

Figura 3.2 – Exemplo de *Hidden Markov Model*.

Formalmente, a definição de um *HMM* compreende:

1) Um conjunto de N estados ocultos $S = \{S_1, S_2, \ldots, S_N\}$ e um estado inicial S_0;
2) Um alfabeto de M símbolos $V = \{v_1, v_2, \ldots, v_M\}$;
3) Uma matriz A de probabilidade de transição entre estados, em que $A[i, j] = P(q[t+1] = j \mid q[t] = i)$, $1 \le i, j \le N$, onde $q[t] = i$ significa que o estado corrente no instante t é S_i;
4) Uma matriz B de probabilidade de emissões observáveis, onde $B[i, k]$ é a probabilidade $P(O[t] = k \mid q[t] = i)$ de o símbolo $v_k \in V$ ter sido observado a partir do estado $S_i \in S$ no instante t; e
5) Um vetor π tal que $\pi[i] = P(q[1] = i)$, $1 \le i \le N$, representando a probabilidade de transição do estado inicial S_0 para os demais.

Um *HMM* é definido pela tripla $\lambda = (A, B, \pi)$. Uma seqüência de observações $O[1 \ldots T]$ é realizada a partir das emissões produzidas por um percurso no grafo de transição de estados, começando sempre pelo estado inicial S_0, onde cada emissão é realizada ao final de cada transição.

Os três problemas centrais da teoria de *HMM*, que estão diretamente relacionados ao seu uso são os seguintes:

1) Problema de Avaliação. Dado o modelo $\lambda = (A, B, \pi)$, obter a probabilidade $P(O[1 \ldots T] \mid \lambda)$ de a seqüência de observações $O[1 \ldots T]$ ter sido gerada pelo modelo λ. Quando a seqüência de observações é obtida a partir de um exemplar da realidade modelada, o valor $P(O[1 \ldots T] \mid \lambda)$ representa a qualidade do modelo λ para a seqüência de observações.

2) Problema da Seqüência de Estados Ótima. Dada uma seqüência de observações $O[1 \ldots T]$ e um modelo λ, selecionar uma seqüência de

estados $q[1\ldots T]$ que tem a maior probabilidade de ter gerado as emissões.

3) Problema de Estimação dos Parâmetros. Como ajustar os parâmetros do modelo $\lambda = (A, B, \pi)$ para maximizar $P(O[1\ldots T] \mid \lambda)$? É o problema de treinamento, em que se deseja estimar as probabilidades iniciais, de transição e de emissão de maneira a maximizar a probabilidade das seqüências de observações de apresentadas, denominadas observações de treinamento.

3.1. Solução para o Problema de Avaliação

A probabilidade $P(O[1\ldots T] \mid \lambda)$, onde $O[1\ldots T]$ é uma seqüência de observações, corresponde à soma de probabilidades de todas as seqüências de estados de comprimento T capazes de gerar $O[1\ldots T]$.

Existem N^T seqüências de estados de comprimento T. Um algoritmo de obtenção de $P(O[1\ldots T] \mid \lambda)$, baseado na enumeração sistemática das seqüências, tem complexidade exponencial em relação ao comprimento T da seqüência. Em vez de investigar o espaço de seqüências de estados de comprimento T e avaliar $P(O \mid \lambda)$, o que levaria a uma complexidade de pior caso $O(TN^T)$, adota-se uma estratégia polinomial em T baseada em programação dinâmica.

Considere o prefixo $O[1\ldots t]$ de comprimento t da seqüência $O[1\ldots T]$, $t \leq T$. Denota-se por $\alpha[t,i]$ a probabilidade do prefixo $O[1\ldots t]$ ter ocorrido e do estado S_i ser o estado corrente no instante t, dado o modelo λ, ou seja

$$\alpha[t,i] = P(O[1\ldots t], q[t] = i \mid \lambda) \tag{3.2}$$

Ora, $P(O[1\ldots T] \mid \lambda) = \sum_{i=1}^{N} \alpha[T,i]$, e $\alpha[t,i]$ pode ser obtido recursivamente por

(1) $\alpha[1,i] = \pi[i]B[i,O[1]]$, para $i = 1,2,\ldots,N$; e

(2) $\alpha[t,i] = B[i, O[t]] \sum_{j=1}^{N} \alpha[t-1, j] A[j, i]$.

O algoritmo de avaliação de $P(O[1\ldots T] \mid \lambda)$, utilizando Programação Dinâmica tabular, baseado nos prefixos de $O[1\ldots T]$ é dado abaixo:

Algoritmo avaliaPrForward ($O[1\ldots T]$) //retorna $P(O[1\ldots T] \mid \lambda)$

```
//Etapa de Inicialização
para i ← 1 até N faça
    α[1,i] ← π[i] * B[i,O[1]]
fim-para
//Etapa de Recursão
para t ← 1 até T-1 faça
        para i ← 1 até N faça
                acumulado ← 0
                para j ← 1 até N faça
                        acumulado ← acumulado + α[t, j] * A[j,i]
                fim-para
                α[t+1,i] ← acumulado * B[i,O[t+1]]
        fim-para
fim-para
//Etapa de Finalização
resultado ← 0
para i ← 1 até N faça
    resultado ← resultado + α[T,i]
fim-para
retorne resultado
```

fim-algoritmo

A Figura 3.3 ilustra como os valores de $\alpha[t, j]$, $1 \le j \le N$, são somados para gerar $\alpha[t+1, i]$.

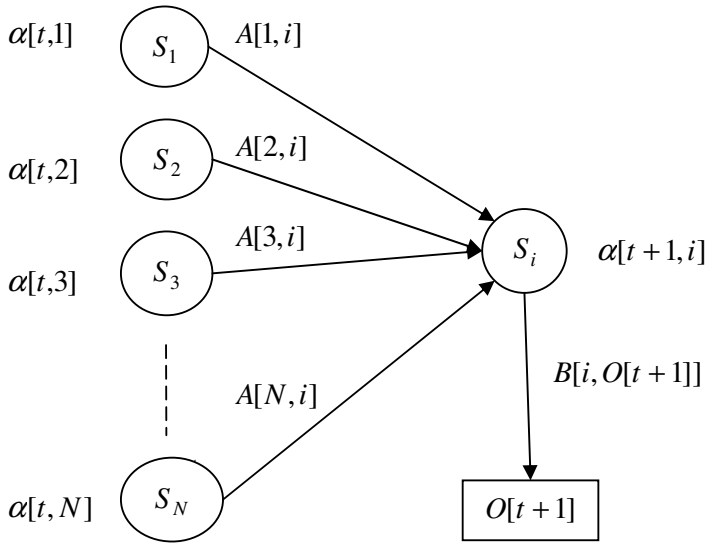

Figura 3.3 – Cálculo de $\alpha[t+1,i]$ em função dos $\alpha[t, j],\ 1 \le j \le N$.

Em função dos três loops aninhados da etapa de recursão, a complexidade do algoritmo é claramente $O(N^2T)$.

Uma outra forma de avaliação de $P(O[1\ldots T]\,|\,\lambda)$ é baseada em sufixos da seqüência $O[1\ldots T]$, isto é, $O[t+1\ldots T]$, $t \ge 0$. Indica-se por $\beta[t,i]$ a probabilidade de ocorrência do sufixo, sendo S_i o estado corrente no instante t e o modelo λ, ou seja

$$\beta[t,i] = P(O[t+1\ldots T]\,|\,q[t]=i,\lambda) \tag{3.3}$$

Logo, tem-se que $P(O[1\ldots T]\,|\,\lambda) = \sum_{i=1}^{N} \pi[i]B[i,O[1]]\beta[1,i]$, e a variável $\beta[t,i]$ é obtida recursivamente por

(1) $\beta[T,i]=1$, para $i=1,2,\ldots,N$; e

(2) $\beta[t,i] = \sum_{j=1}^{N} A[i,\ j]B[\,j,O[t+1]]\beta[t+1,\ j]$.

O algoritmo de avaliação de $P(O[1\ldots T]\,|\,\lambda)$, utilizando Programação Dinâmica tabular, baseado nos sufixos de $O[1\ldots T]$ é dado abaixo:

Algoritmo avaliaPrBackward ($O[1\ldots T]$) //retorna $P(O[1\ldots T] \mid \lambda)$

//Etapa de Inicialização

para i ← 1 **até** N **faça**

$\beta[T, i] \leftarrow 1$

fim-para

//Etapa de Recursão

para t ← T-1 até 1 **faça**

para i ← 1 **até** N **faça**

$\beta[t, i] \leftarrow 0$

para j ← 1 **até** N **faça**

$\beta[t, i] \leftarrow \beta[t, i] + A[i, j] * B[j, O[t+1]] * \beta[t+1, j]$

fim-para

fim-para

fim-para

//Etapa de Finalização

resultado ← 0

para i ← 1 **até** N **faça**

resultado ← resultado + $\pi[i] * B[i, O[1]] * \beta[1, i]$

fim-para

retorne resultado

fim-algoritmo

A Figura 3.4 ilustra o funcionamento do algoritmo:

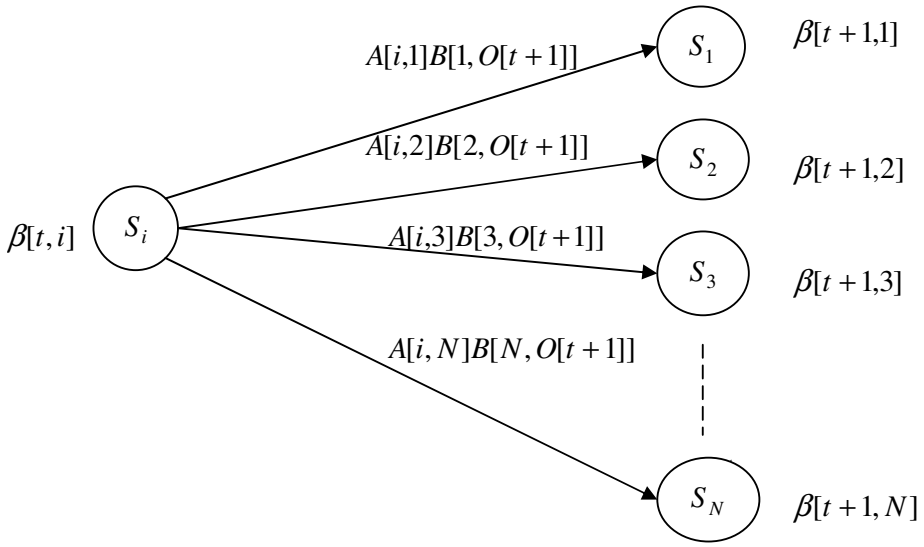

Figura 3.4 – Cálculo de $\beta[t,i]$ em função dos $\beta[t+1, j],\ 1 \le j \le N$.

A complexidade do algoritmo é $O(N^2T)$, em função dos três loops aninhados. Possui, pois, a mesma complexidade que o algoritmo *avaliaPrForward*. Portanto, pode-se optar livremente por um ou por outro para a resolução do problema de avaliação. Ademais, ambos os algoritmos assumem um papel importante na solução do problema de estimação dos parâmetros do modelo, como será visto adiante.

3.2. Solução para o Problema da Seqüência de Estados Ótima

Diversos critérios de ótimo podem ser usados para a obtenção da seqüência de estados a partir das observações $O[1\ldots T]$. Examinam-se dois critérios:

1) Determinar o estado ótimo para cada observação $O[i]$, $1 \le i \le T$; e
2) Determinar uma seqüência de estados que maximize a probabilidade das observações $O[1\ldots T]$ como um todo.

O critério de ótimo utilizado no algoritmo de *Viterbi*, no qual os experimentos realizados foram baseados, é o segundo.

Em relação ao primeiro critério, a variável:

$$\gamma[t,i] = P(q[t] = i \mid O[1\ldots T], \lambda) \tag{3.4}$$

fornece a probabilidade de se estar no estado S_i em um instante t, sabendo-se a seqüência de emissões e o modelo. Essa variável pode ser expressa em termos das variáveis α e β anteriormente definidas, já que, a partir do modelo λ, a primeira fornece a probabilidade de ocorrência das observações $O[1\ldots t]$ conjugada com a ocorrência do estado corrente S_i no instante t ($q[t] = i$) e a segunda fornece a probabilidade de ocorrência das observações $O[t+1\ldots T]$ dado que o estado corrente em t foi S_i ($q[t] = i$).

$$\gamma[t,i] = \frac{\alpha[t,i]\beta[t,i]}{P(O[1\ldots T] \mid \lambda)} = \frac{\alpha[t,i]\beta[t,i]}{\sum_{i=1}^{N} \alpha[t,i]\beta[t,i]} \tag{3.5}$$

Dessa forma, para uma seqüência de emissões de tamanho T, o estado $q[t]$ mais provável em cada instante t é dado por:

$$q[t] = \arg\max_{1 \le i \le N}(\gamma[t,i]), \quad 1 \le t \le T \tag{3.6}$$

onde *arg* é uma função que retorna o índice do estado que maximiza γ no instante t. Portanto, pode-se obter, a partir da expressão 3.6, uma seqüência de estados em que cada estado $q[t]$ individualmente é mais provável de ter gerado a observação $O[t]$.

A complexidade de um algoritmo baseado na expressão 3.6 para resolver o problema é $O(N^2T)$, pois: (1) o cálculo das variáveis α e β exige tempo $O(N^2T)$, conforme visto na seção 3.1; (2) o cálculo de $P(O[1\ldots t] \mid \lambda) = \sum_{i=1}^{N} \alpha[t,i]\beta[t,i]$ exige tempo $O(N)$, uma vez resolvido o item 1; (3) o cálculo de todos os $\gamma[t,i]$, $1 \le i \le N$ *e* $1 \le t \le T$ toma $O(NT)$, uma vez resolvidos os itens 1 e 2; e, finalmente, a obtenção de cada $q[t]$ exige a comparação dos valores de $\gamma[t,i]$ para todos os N estados, tomando tempo $O(N)$ para cada um, e $O(NT)$ para todos os valores de t. Logo, a complexidade do algoritmo é dominada pelo cálculo das variáveis α e β, tomando, portanto, tempo $O(N^2T)$.

Este tipo de critério de maximização pode levar a resultados inconsistentes. Isto pode ocorrer quando o modelo λ apresenta probabilidades de transição $A[i, j] = 0$, para algum par i e j, e a maximização local em cada t obtém $q[t] = i$ e $q[t+1] = j$. O segundo critério de otimização não apresenta este tipo de inconsistência.

Para obter uma seqüência de estados que maximize a probabilidade da seqüência de observações $O[1\ldots T]$, adota-se uma estratégia de Programação Dinâmica.

A Programação Dinâmica é uma estratégia ascendente (*bottom-up*) que resolve problemas através da decomposição em subproblemas e combinação das soluções destes. É aplicado em problemas de otimização combinatória, isto é, em que se deseja maximizar ou minimizar uma solução que consiste em uma combinação de elementos (CORMEN et al., 2005).

Essa estratégia pode ser aplicada em problemas de otimização que possuam duas características:

1) Subestrutura ótima: uma solução ótima para o problema contém soluções ótimas para todos os seus subproblemas; e
2) Existência de um número polinomial (em relação ao tamanho da entrada do problema) de subproblemas sobrepostos (subproblemas que aparecem repetidamente na decomposição de um problema).

A maximização da seqüência (caminho) de estados como um todo é sempre consistente e corresponde a maximizar $P(q[1\ldots T] \mid O[1\ldots T], \lambda)$. Como

$$P(q[1\ldots T] \mid O[1\ldots T], \lambda) = \frac{P(q[1\ldots T], O[1\ldots T] \mid \lambda)}{P(O[1\ldots T] \mid \lambda)} \tag{3.7}$$

e $O[1\ldots T]$ é constante, tem-se que maximizar $P(q[1\ldots T] \mid O[1\ldots T], \lambda)$ é equivalente a maximizar $P(q[1\ldots T], O[1\ldots T] \mid \lambda)$.

Considere a variável

$$\delta[t, i] = \max_{q[1\ldots(t-1)]} P(q[1\ldots t-1], q[t] = i, O[1\ldots t] \mid \lambda) \tag{3.8}$$

que fornece a maior probabilidade de uma seqüência de estados $S_{q[1]}, S_{q[2]}, \ldots, S_{q[t-1]}, S_i$, ocorrer concomitantemente com as observações $O[1 \ldots t]$, até o instante t, com $q[t] = i$. Recursivamente, tem-se:

$$\delta[t+1, i] = (\max_{1 \le j \le N} \delta[t, j]\, A[j, i]) B[i, O[t+1]] \qquad (3.9)$$

O seguinte algoritmo, desenvolvido por Andrew James Viterbi (VITERBI, 1967) em um contexto de correções de erros para ruídos de comunicações digitais, calcula a seqüência de estados ótima a partir de uma seqüência de observações:

Algoritmo Viterbi ($O[1 \ldots T]$)

```
//Etapa de Inicialização
para i ← 1 até N faça
    δ[1, i] ← π[i] * B[i, O[1]]
    ψ[1, i] ← −1
fim-para
//Etapa de Recursão
para t ← 2 até T faça
    para i ← 1 até N faça
        maior ← δ[t − 1, 1] * A[1, i]
        arg_maior ← 1
        para j ← 2 até N faça
            aux ← δ[t − 1, j] * A[j, i]
            se aux > maior então
                maior ← aux
                arg_maior ← j
            fim-se
        fim-para
        δ[t, i] ← maior * B[i, O[t]]
        ψ[t, i] ← arg_maior
    fim-para
fim-para
```

//Etapa de Finalização

maior $\leftarrow \delta[T,1]$

arg_maior $\leftarrow 1$

para i $\leftarrow$ 2 **até** N **faça**

aux $\leftarrow \delta[T,i]$

se aux > maior **então**

maior $\leftarrow$ aux

arg_maior $\leftarrow$ i

fim-se

fim-para

$P^* \leftarrow$ maior

$q^*[T] \leftarrow$ arg_maior

//Etapa de recuperação da seqüência de estados

para t $\leftarrow$ T-1 até 1 **faça**

$q^*[t] \leftarrow \psi[t+1, q^*[t+1]]$

fim-para

fim-algoritmo

A matriz $\psi[i,t]$ armazena o estado que maximiza a expressão 3.9, para cada i e t.

A complexidade do algoritmo é $O(N^2T)$ pelos três laços no cálculo dos valores de $\delta[2\ldots T,1\ldots N]$ na etapa de recursão.

Para um modelo de ordem dois, o algoritmo de *Viterbi* exige algumas modificações, já que deve levar em conta os dois estados anteriores (THEDE; HARPER, 1999).

3.3. Solução para o Problema de Estimação dos Parâmetros

O Problema 3 corresponde a determinar os parâmetros do modelo estocástico $\lambda = (A, B, \pi)$ que maximizem a probabilidade de emissões já observadas, isto é:

obtenha λ tal que $P(O[1\ldots T]\,|\,\lambda)$ seja máximo. Não existe método analítico de resolvê-lo, mas há um método iterativo (que é, a rigor, um método de Estimação da Máxima Verossimilhança), que maximiza localmente a probabilidade, consistindo em uma Busca Local pelo espaço de soluções.

O algoritmo de *Baum-Welch* utiliza a variável $\xi[t,i,j]$, que fornece a probabilidade de se estar no estado S_i no instante t e em S_j no instante $t+1$, dados o modelo λ e a seqüência de emissões $O[1\ldots T]$, conforme ilustrado na Figura 3.5:

$$\xi[t,i,j] = P(q[t]=i, q[t+1]=j \,|\, O[1\ldots T], \lambda) \tag{3.10}$$

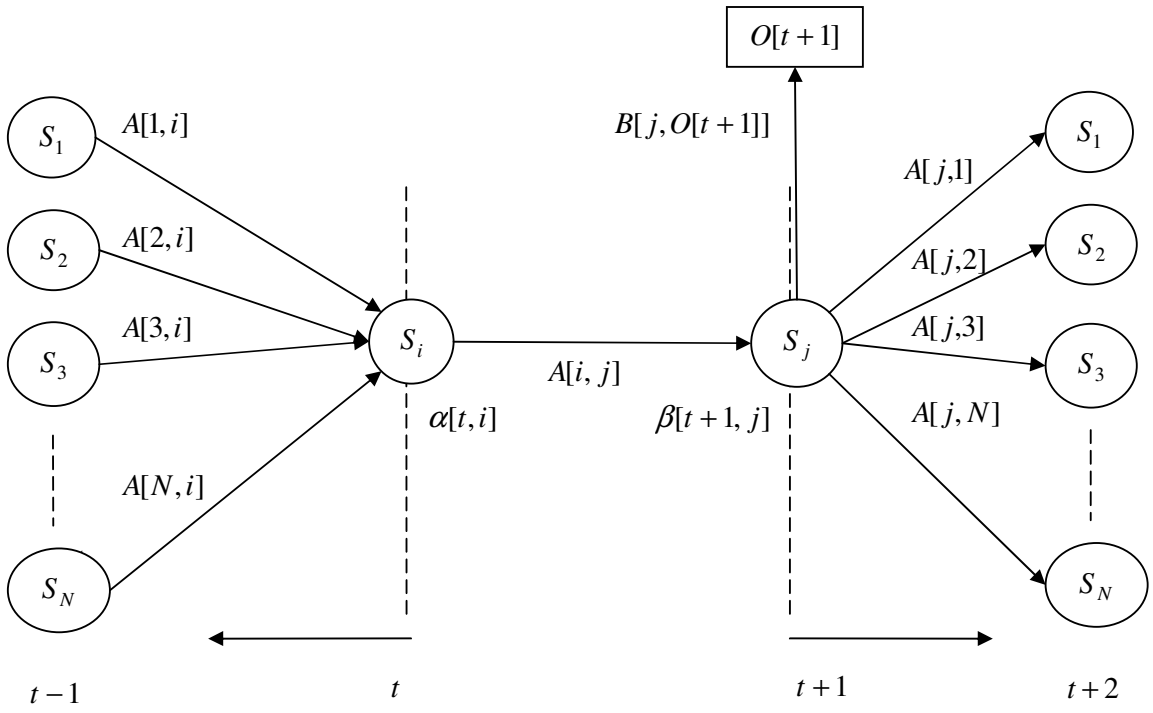

Figura 3.5 – Ilustração esquemática do cálculo da variável $\xi[t,i,j]$.

A variável pode ser escrita em função de α e β, da seguinte forma:

$$\xi[t,i,j] = \frac{\alpha[t,i]A[i,j]B[j,O[t+1]]\beta[t+1,j]}{P(O[1\ldots T]\,|\,\lambda)} \tag{3.11}$$

O cálculo da variável $\xi[1\ldots T,1\ldots N,1\ldots N]$ pressupõe o cálculo das variáveis α e β, que exige tempo $O(N^2T)$ (ver seção 3.1). O denominador pode ser calculado a partir das variáveis α ou β, em $O(N^2T)$ operações (ver seção 3.1). Logo, a complexidade corresponde a $O(N^2T)$.

Pode-se estabelecer a seguinte relação entre a variável $\xi[t,i,j] = P(q[t] = i, q[t+1] = j \mid O[1\ldots T], \lambda)$ definida acima e a variável $\gamma[t,i] = P(q[t] = i \mid O[1\ldots T], \lambda)$, estabelecida na seção 3.2:

$$\gamma[t,i] = \sum_{j=1}^{N} \xi[t,i,j] \tag{3.12}$$

Ao se somar $\gamma[t,i]$ através do tempo t, $1 \le t \le T-1$, obtém-se o **número esperado de transições** feitas a partir do estado S_i, supondo a seqüência de observações em $O[1\ldots T]$. Analogamente, efetuando-se a soma de $\xi[t,i,j]$, com $1 \le t \le T-1$, obtém-se uma medida representativa do número esperado de transições do estado S_i para o estado S_j.

Utilizando-se as conclusões acima podem ser obtidos novos parâmetros para um modelo λ, descrito através das seguintes expressões:

$$\text{número esperado de vezes no estado } S_i \text{ em } (t=1) \equiv \bar{\pi}[i] = \gamma[1,i] \tag{3.13}$$

$$\bar{A}[i,j] = \frac{\text{número esperado de transições de } S_i \text{ para } S_j}{\text{número esperado de transições a partir do estado } S_i} = \frac{\sum_{t=1}^{T-1} \xi[t,i,j]}{\sum_{t=1}^{T-1} \gamma[t,i]} \tag{3.14}$$

$$\bar{B}[i,k] = \frac{\text{número esperado de vezes em que, no estado } S_i, \text{ observa - se o símbolo } v_k \in V}{\text{número esperado de vezes no estado } S_i} = \frac{\sum_{t=1}^{T} \gamma[t,i] * isTrue(O[t] = k)}{\sum_{t=1}^{T} \gamma[t,i]} \tag{3.15}$$

onde *isTrue()* é uma função booleana que retorna 1 se a expressão entre parênteses é verdadeira e 0 em caso contrário.

Para o caso em que houver mais de uma seqüência de observações, isto é, $\mathbf{O} = O^{(1)}, O^{(2)}, \ldots, O^{(r)}$, utilizam-se, normalmente, as seguintes expressões adicionais:

$$\bar{\pi}[i] = \frac{1}{r} \sum_{n=1}^{r} \bar{\pi}^{(n)}[i] \tag{3.16}$$

$$\bar{A}[i, j] = \frac{1}{r} \sum_{n=1}^{r} \bar{A}^{(n)}[i, j] \tag{3.17}$$

$$\bar{B}[i, k] = \frac{1}{r} \sum_{n=1}^{r} \bar{B}^{(n)}[i, k] \tag{3.18}$$

em que:

r - número de seqüências de observações utilizadas no treinamento; e

$\bar{\lambda}^{(n)} = (\bar{\pi}^{(n)}, \bar{A}^{(n)}, \bar{B}^{(n)})$ - é o modelo obtido através da aplicação das fórmulas 3.13 a 3.15 para a n-ésima seqüência de observações.

Cada seqüência de observações contribui para a estimação do novo modelo $\bar{\lambda}$.

Com base nas expressões acima, é definido o seguinte algoritmo para estimação dos parâmetros de um modelo:

Algoritmo Baum-Welch ($\mathbf{O} = O^{(1)}, O^{(2)}, \ldots, O^{(r)}$, máximoIterações)

$\lambda \leftarrow$ Estime_Lambda( ); // possivelmente assumindo valores aleatórios para π, A e B.

loop

$\bar{\lambda} \leftarrow 0$ //O acumulador é zerado

para i ← 1 **até** r **faça**

$O[1 \ldots T] \leftarrow O^{(i)}[1 \ldots T]$;

α ← CalculeAlfa(λ, O[1...T]);

β ← CalculeBeta(λ, O[1...T]);

P ← avaliaPrObs(α, O[1...T]);

ξ ← CalculeXi(λ, α, β, O[1...T], P);

$\gamma \leftarrow$ CalculeGamma(ξ);

$\overline{\lambda}^{(i)} \leftarrow$ Calcule_Lambda(O[1...T], γ, ξ); //através das expressões 3.13-15

$\overline{A} \leftarrow \overline{A} + \overline{A}^{(i)}$; $\overline{B} \leftarrow \overline{B} + \overline{B}^{(i)}$; $\overline{\pi} \leftarrow \overline{\pi} + \overline{\pi}^{(i)}$; //acumulação das expressões 3.15-17

fim-para

$\overline{\lambda} \leftarrow \left(\frac{\overline{A}}{r}, \frac{\overline{B}}{r}, \frac{\overline{\pi}}{r} \right)$ //cada elemento de $\overline{A}, \overline{B}, \text{e } \overline{\pi}$ é dividido por r

$\Delta \leftarrow$ distancia($\lambda, \overline{\lambda}$)

se ($\Delta \leq \varepsilon \vee$ númeroIterações $\geq$ máximoIterações) **saia do loop**

$\lambda \leftarrow \overline{\lambda}$;

númeroIterações++;

fim-loop

fim-algoritmo

O critério de parada é alcançar um número máximo de iterações predefinido (em função de limitações de tempo, principalmente) ou quando houver convergência local para um modelo, no sentido em que se considera que o novo modelo recém calculado e o anterior estão suficientemente próximos. A função de distância pode ser uma média das distâncias de todos os elementos de um modelo para o outro, ou ser a maior destas distâncias, por exemplo. O valor de ε pode variar de aplicação para aplicação.

$\overline{\lambda} = (\overline{A}, \overline{B}, \overline{\pi})$ é o novo modelo obtido após uma iteração do loop principal: se o modelo, inicial ou da iteração anterior, λ define um ponto crítico da função de verossimilhança, então, necessariamente, $\Delta \leq \varepsilon$; caso contrário, o modelo $\overline{\lambda}$ é mais provável que o modelo λ, ou seja, $P(\mathbf{O} \mid \overline{\lambda}) > P(\mathbf{O} \mid \lambda)$, e o novo modelo possui maior probabilidade de ter gerado as seqüências de treinamento apresentadas.

O algoritmo descrito acima, pelo fato de ser uma Busca Local, só garante levar a máximos locais, pois normalmente existem diversos máximos locais no espaço de busca (FERREIRA, 2005).

Uma forma de tentar fugir da armadilha da convergência para máximos locais de baixa qualidade é começar a busca em regiões heuristicamente mais promissoras, no sentido de estarem mais próximas do máximo global. Isto é possível através da estimação de bons valores para os parâmetros, em vez de apenas defini-los aleatoriamente, mas isto exige um conhecimento aprofundado sobre o domínio da aplicação. Na prática, percebe-se que boas estimativas inicias para a matriz B assumem um papel importante, ao passo que estimativas aleatórias para A e π são normalmente satisfatórias (SCHÜTZE; MANNING, 1999).

Na análise da complexidade de uma iteração do algoritmo de *Baum-Welch*, considera-se que cada uma das r observações toma tempo $O(T)$. As funções CalculeAlfa( ), CalculeBeta( ), avaliaPrObs( ), CalculeXi( ) e CalculeGamma( ) possuem, em conjunto, complexidade $O(N^2T)$, conforme seções anteriores.

Por sua vez, na operação Cacule_Lambda( ), o cálculo da expressão 3.13 toma $O(N)$, o da expressão 3.14 toma $O(N^2T)$ e o da 3.15 leva $O(NMT)$, sendo M o número de símbolos do alfabeto. Como tais passos são realizados para cada seqüência de observações, conclui-se que a complexidade de uma iteração corresponde a $O(rNT(N+M))$, em que r é o número de seqüências de observações utilizadas.

O algoritmo de *Baum-Welch* é classificado também como sendo uma abordagem de aprendizado não-supervisionado, na medida em que o treinamento é feito a partir de um conjunto de dados não-etiquetado, isto é, em que não há um gabarito através do qual se possa aprender.

Outra abordagem à estimação dos parâmetros do modelo λ, quando se dispõe de um conjunto de dados etiquetado corretamente, é a estimação a partir de contagem das freqüências relativas de cada estado e de cada emissão, tal como sugerido em (SCHÜTZE; MANNING, 1999). De fato, o que se faz é obter a estimativa de máxima verossimilhança. As fórmulas utilizadas são as seguintes:

$$\pi[i] = \frac{C\_START[i]}{NUM\_SENTENCAS} \tag{3.19}$$

$$A[i,j] = \frac{C\_TRANS[i,j]}{C[i]} \tag{3.20}$$

$$B[i,k] = \frac{C[i,k]}{C[i]} \tag{3.21}$$

em que:

$C\_START[i]$ - número de ocorrências (contagem) em que o estado S_i é o estado inicial;

$NUM\_SENTENCAS$ - número de sentenças;

$C\_TRANS[i, j]$ - número de ocorrências em que o estado S_i é seguido pelo estado S_j;

$C[i]$ - número de ocorrências do estado S_i; e

$C[i,k]$ - número de ocorrências em que, estando-se no estado S_i, emitiu-se o símbolo v_k.

Como tais contagens são obtidas a partir de um conjunto de treinamento manualmente etiquetado, esta abordagem é classificada como sendo de aprendizado supervisionado.

De maneira geral, Schütze e Manning (SCHÜTZE; MANNING, 1999) apontam para três possibilidades: (1) quando se dispõe de textos anotados corretamente e que são similares aos textos-alvo, aos quais o modelo será aplicado, deve-se optar pela abordagem supervisionada, procedendo-se à estimação através de contagem; (2) em casos em que não se dispõe de textos anotados ou quando estes forem muito diferentes dos textos-alvo, mas se detêm algumas informações léxicas, deve-se realizar apenas algumas iterações do algoritmo de *Baum-Welch*; e (3) quando não há textos anotados ou qualquer informação léxica, devem ser realizadas dez ou mais iterações de *Baum-Welch*, mas sem esperar uma grande performance do modelo obtido.

3.4. Considerações sobre Erros Numéricos

Nos algoritmos vistos nas seções anteriores, são realizadas repetidas operações com probabilidades, números entre 0 e 1, e que podem ocasionar problemas de *underflow* em função da representação computacional de reais como números de ponto flutuante com precisão limitada.

Para o algoritmo de *Viterbi*, que envolve apenas a multiplicação de probabilidades, pode-se utilizar a função logaritmo e sua inversa para reduzir o

impacto desse problema. Fez-se uso desse artifício na implementação do modelo de ordem dois.

Contudo, nos algoritmos *avaliaPrForward* e *avaliaPrBackward* (e, indiretamente, o algoritmo de *Baum-Welch*), são realizadas não apenas multiplicações, mas também adições entre probabilidades, o que limita a abrangência do uso da função logaritmo como recurso de redução de erros numéricos. Uma solução é utilizar fatores de escala, de tal forma que as probabilidades permaneçam dentro de um intervalo numérico pré-estabelecido. Confronte (RABINER, 1989), (SCHÜTZE; MANNING, 1999) e (CUNHA, 2002). Tais artifícios foram utilizados na implementação dos algoritmos *avaliaPrForward*, *avaliaPrBackward* e *Baum-Welch* tanto para o modelo simples quanto para o modelo de ordem dois.

Quando as probabilidades de algumas transições entre estados ou de emissões forem avaliadas como nulas, por não ocorrerem no conjunto de treinamento, deve-se proceder a técnicas de suavização (*smoothing*). Para modelos de ordem dois, um algoritmo comumente utilizado é o de *deleted interpolation*. Para maiores detalhes, ver (FELDMAN; SANGER, 2007) e (THEDE; HARPER, 1999).

Capítulo 4 – Experimentos Computacionais

4.1. *Corpora* Utilizados

Um *corpus* (corpo em latim), com plural *corpora*, é uma coleção de textos colocados juntos para se pesquisarem um ou mais aspectos da língua (BILISOLY, 2008). Foram utilizados dois *corpora* pertencentes à coleção dourada (*Golden Collection*) do *HAREM*, que é um evento de *NER* em língua portuguesa realizado em Portugal pela *Linguateca*, um centro de pesquisa do processamento do português (LINGUATECA). O objetivo desses eventos é estimular pesquisas nesse âmbito e promover competições de avaliação aos moldes dos *MUCs*.

Os textos utilizados no evento provêm de diversos países de língua portuguesa (Brasil, Portugal, Angola e etc.) e diversos gêneros (jornalístico, literário, técnico e etc.). A coleção dourada é um conjunto de textos manualmente anotados com as entidades mencionadas, sendo utilizada como conjunto de treinamento para os sistemas que participarem do evento.

Uma lista contendo as classes de entidades presentes nos textos está no Apêndice A. O *Corpus* 1 se refere à coleção dourada utilizada na primeira avaliação *HAREM*, ocorrida em 2005, ao passo que o *Corpus* 2 se refere à coleção dourada presente no evento *MiniHAREM*, ocorrido em 2006.

Como algumas classes de entidades possuíam baixa freqüência de ocorrência e a fim de não se aumentar o tamanho dos modelos propostos, realizou-se um mapeamento entre estas, conforme descrito no Apêndice A. O Quadro 4.1 descreve numericamente os *corpora*.

Quadro 4.1 – Descrição dos *corpora*.

	***Corpus* 1**	***Corpus* 2**
Sentenças	4759	3390
Tokens	98810	66757
Não-Entidades Mencionadas	88622	59417
Entidades Mencionadas	10188	7340
Proporção (Não-EM / EM)	8,70	8,09
Quantitativo por Classe de Entidade:		
PESSOA	2048	1619
LOCAL	2003	1384
TEMPO	764	613
VALOR	944	598
OUTRA	4429	3126

Os *corpora* foram analisados morfossintaticamente (*POS Tagging*) por um etiquetador desenvolvido no Laboratório de Engenharia de Algoritmos e Redes Neurais (LEARN[1]) da Pontifícia Universidade Católica do Rio de Janeiro – PUC-Rio. As etiquetas utilizadas estão descritas no Anexo A. Como foram levadas em consideração muitas classificações gramaticais específicas, o número de possíveis etiquetas é grande. Dessa forma, o mapeamento contido no Apêndice B é feito, visando reduzir o número de etiquetas e, conseqüentemente, reduzir também o número de emissões do modelo markoviano utilizado.

O Quadro 4.2 demonstra as freqüências absoluta e relativa das anotações gramaticais, após o mapeamento, nos dois textos.

[1] Consultar site: http://agogo.learn.fplf.org.br/index.php/Welcome_to_LEARN_homepage%21

Quadro 4.2 – Freqüências das etiquetas morfossintáticas nos *corpora*.

	***Corpus* 1**	**Freq. Relativa (%)**	***Corpus* 2**	**Freq. Relativa (%)**
ADJ	3858	3,90	3106	4,65
ADV	4191	4,24	2363	3,54
ART	11389	11,53	7648	11,46
CUR	16	0,02	13	0,02
IN	128	0,13	70	0,10
KC	3039	3,08	1912	2,86
KS	1469	1,49	916	1,37
N	17695	17,91	12373	18,53
NPROP	7942	8,04	5771	8,64
NUM	788	0,80	529	0,79
PCP	1569	1,59	1143	1,71
PDEN	555	0,56	370	0,55
PREP	13666	13,83	9415	14,10
PRONOME	7559	7,65	4722	7,07
VERBO	11437	11,57	7041	10,55
PONTUACAO	11284	11,42	7453	11,16
DELIMITADOR	2225	2,25	1912	2,86
Total:	98810	100,00	66757	100,00

Durante o processo de anotação morfossintática, contrações (fusão de uma preposição com um artigo) e combinações (encadeamento de uma preposição com um artigo) foram desfeitas, tais como: “da” expandido para “de a” e “aos” expandido para “a os”.

Um fragmento ilustrativo do *corpus* 2, referente à coleção dourada do *MiniHAREM*, segue no Anexo B.

4.2. Modelagem Empregada

A idéia da modelagem de um sistema estocástico para *NER* é assumir que se detinham as sentenças com as entidades etiquetadas corretamente, mas que, após essas serem transmitidas por um dado canal com ruído, ficaram apenas anotadas morfossintaticamente. O modelo permite que se tente desfazer o processo, isto é, dada uma sentença anotada gramaticalmente, busca-se remover otimamente o ruído e restaurar as anotações *NER*. Essa abordagem é descrita em (NUGUES, 2006) em um contexto de marcação gramatical de frases a partir de suas palavras. A Figura 4.1 ilustra a abordagem.

Foram desenvolvidos dois modelos de *HMM*: um simples, com estados representativos apenas das entidades consideradas, e outro de ordem dois.

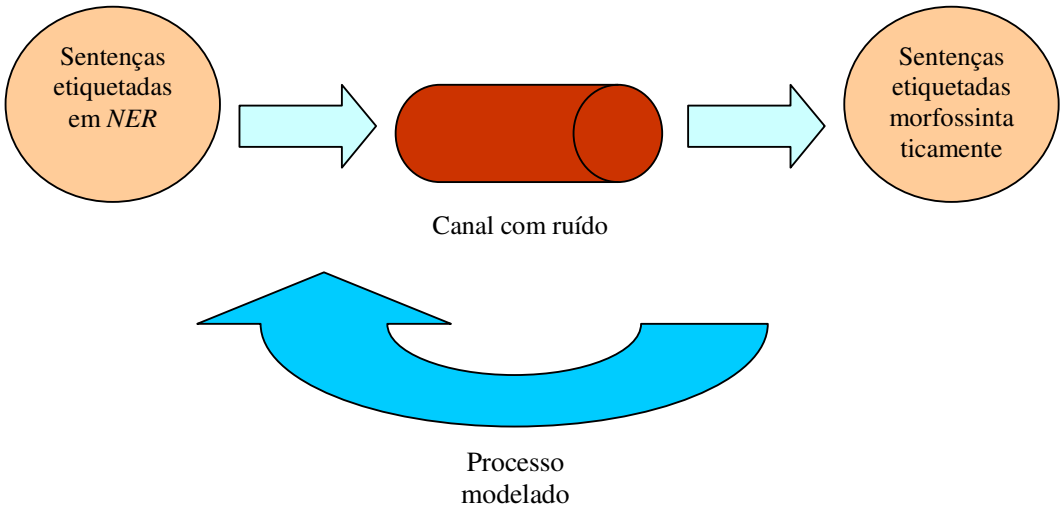

Figura 4.1 – Modelagem adotada.

4.2.1. Modelo Simples

O modelo simples de *HMM* idealizado corresponde à situação em que as observações são as etiquetas morfossintáticas, ao passo que os estados escondidos, sobre os quais se deseja obter informação, são as classes de entidades mencionadas.

Além destes estados, há dois estados adicionais, utilizados para marcar o início e o fim de cada sentença analisada: *START* e *END*. O grafo de transição de estados é formado por uma clique envolvendo o conjunto de estados $S = \{START, END, PESSOA, LOCAL, TEMPO, VALOR, OUTRA, O\}$, um arco ligando *START* a cada estado em S e um arco ligando cada estado em S ao estado final *END*. A Figura 4.2 ilustra a topologia, com a omissão de vários arcos, para maior clareza.

Para avaliação de um dado texto, a partir de sua anotação morfossintática, aplica-se o algoritmo de *Viterbi* para a obtenção da seqüência de estados mais provável para cada sentença. Obtém-se, assim, para cada uma, a extração e classificação de seus *tokens* em entidades mencionadas.

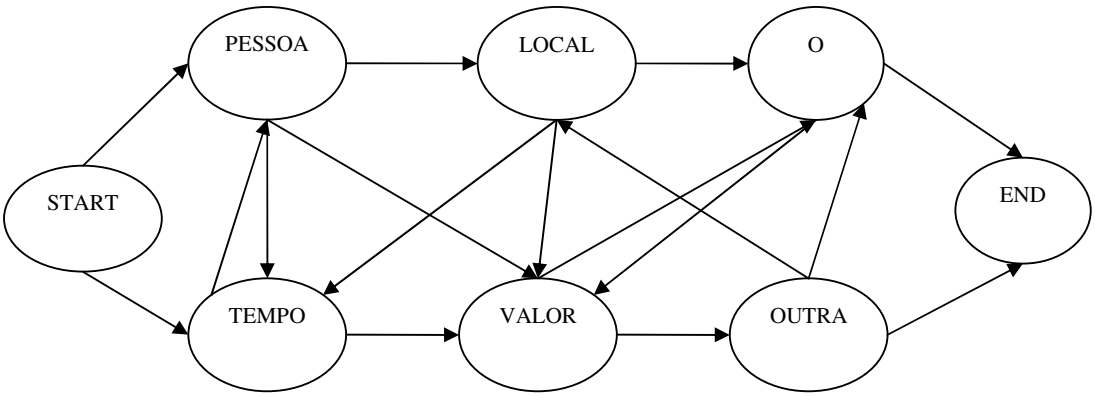

Figura 4.2 – Topologia do modelo simples concebido.

4.2.2. HMM de Ordem Dois

Como esperado, o modelo de primeira ordem não obteve resultados muito promissores. Visando a obter melhores resultados, utilizou-se também um modelo de ordem dois em que, por extensão da expressão 3.1, a probabilidade de se estar em um estado depende dos dois imediatamente anteriores, ou seja:

$$P(s[t] = S_{i\_t} \mid s[t-1] = S_{i\_(t-1)}, s[t-2] = S_{i\_(t-2)}, \ldots, s[1] = S_{i\_1}) = P(s[t] = S_{i\_t} \mid s[t-1] = S_{i\_(t-1)}, s[t-2] = S_{i\_(t-2)}) \quad (4.1)$$

Foram implementados dois tipos de modelos de ordem dois: um real, aumentando-se a dimensão das matrizes A e B, conforme descrito em (THEDE; HARPER, 1999) e outro que faz uma simulação através das etiquetas. Nos testes realizados, o que obteve o melhor desempenho foi o segundo. Um dos problemas enfrentados na implementação do primeiro foi lidar com os mecanismos de interpolação nas matrizes obtidas, já que ocorre o problema de esparsidade, em que grande parte dos valores são nulos, devido à não-ocorrência de muitos trigramas no conjunto de treinamento. Foi utilizado o método de *deleted interpolation*, estimando através das probabilidades unigrama, bigrama e trigrama (THEDE; HARPER, 1999), mas, aparentemente, esse não funcionou. Um estudo mais aprofundado do tema talvez possa solucionar tal problema e obter resultados mais promissores.

Assim sendo, o modelo implementado faz uma simulação de um modelo de ordem dois simplesmente se utilizando do nome das etiquetas, de modo a refletir o estado anterior. Mais especificamente, um estado cujo nome é PESSOA-LOCAL indica que o estado atual é um LOCAL e o anterior é uma PESSOA. Para as etiquetas morfossintáticas, o mesmo procedimento é adotado, de forma que PREP-VERBO indica que a observação atual é um VERBO e o anterior é uma PREP (Preposição).

Para este modelo, não foi necessário utilizar suavização nas probabilidades de transição e de emissão, mas fez-se uso do algoritmo de *Viterbi* com logaritmo das probabilidades, conforme descrito na seção 3.4, para reduzir/eliminar a influência da propagação de erros numéricos de ponto flutuante.

Além disso, especificamente neste modelo utilizou-se um *gazetteer* como classificador auxiliar, a fim de aumentar a probabilidade de classificação correta. Utilizou-se o único *gazetteer* disponível em língua portuguesa, *REPENTINO* – REPositório para reconhecimento de ENTidades com NOme (SARMENTO; PINTO; CABRAL, 2006) e (REPENTINO). Este *gazetteer* é um repositório público que contém aproximadamente 450 mil entidades nomeadas, isto é, em que são ignoradas quaisquer informações sobre o contexto de onde essas foram retiradas. As entidades estão separadas por classes e, dentro destas, por subcategorias específicas, em uma estrutura de árvore, conforme exemplificado no Anexo C.

Antes de ser utilizado, o *gazetteer* passou por um processo de "limpeza", em que foi formatado para facilitar o seu processamento. Todas as categorias, excetuando-se SER, LOC e ORG foram eliminadas, além da eliminação de grande parte das *tags XML*.

Toda vez que um *token* com uma etiqueta NPROP, que corresponde a um nome próprio, é encontrado no *corpus* de avaliação, adotou-se a seguinte heurística: Busca-se no *gazetteer* uma ocorrência do *token* priorizando primeiramente a (1) Lista de Pessoas; depois (2) Lista de Locais; e finalmente (3) Lista de Organizações.

Assim, para cada sentença processada, procura-se no *gazetteer* pelos *tokens* que correspondam a nomes próprios. Caso sejam encontrados, obtêm-se suas classificações. Após isso, as etiquetas morfossintáticas referentes a essa sentença são submetidas ao modelo de ordem dois e sua saída obtida. Para os *tokens* em que se obteve sucesso no *gazetteer*, a classificação fornecida pelo *HMM* é ignorada. A arquitetura segue descrita na Figura 4.3.

Para realizar a comparação de cadeias de caracteres, utilizou-se a função *strstr(string1, string2)* da biblioteca *cstring*, que retorna um ponteiro para a primeira ocorrência de *string2* em *string1* (*cplusplus.com*). Seu uso se justifica pelo fato de ter obtido um melhor tempo de execução do que a implementação realizada para os algoritmos *Knuth-Morris-Pratt* e *Boyer-Moore* (ver CORMEN et al., 2005 e SEDGEWICK, 1984).

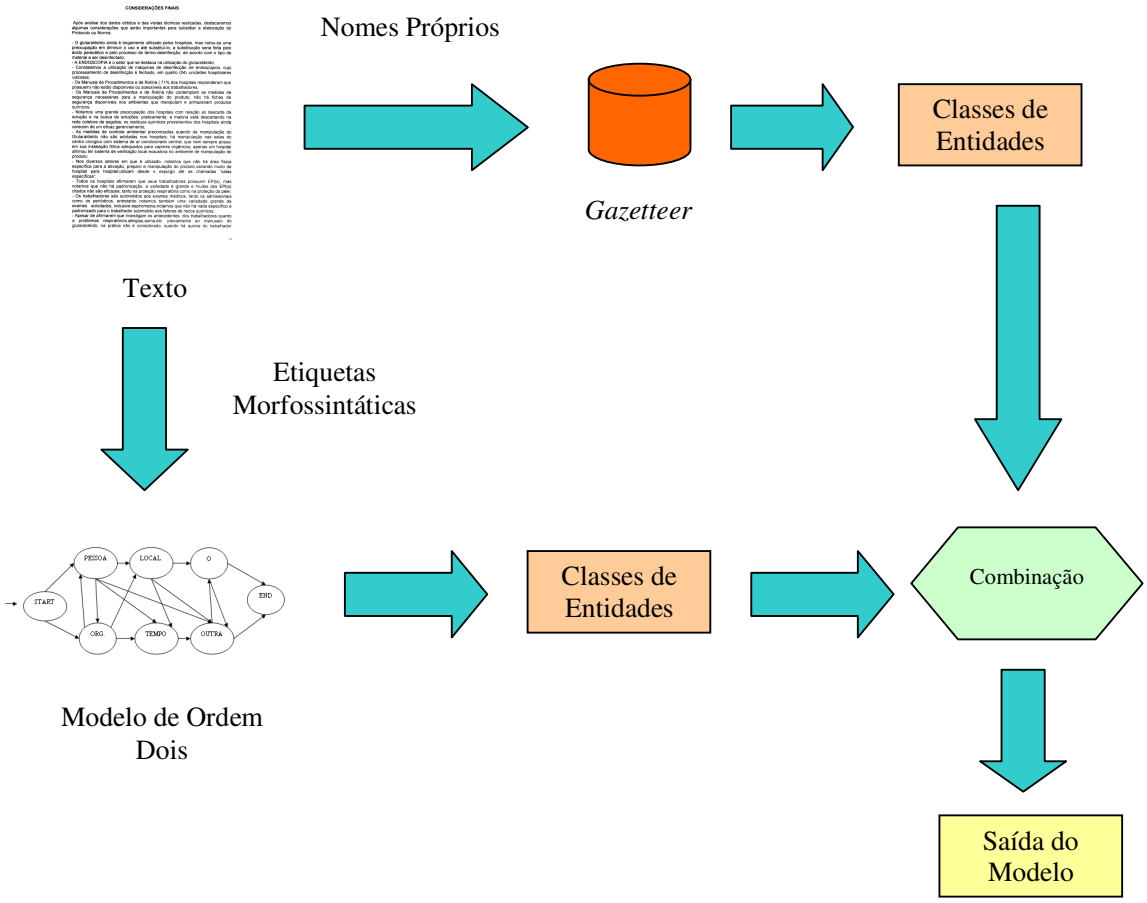

Figura 4.3 – Arquitetura do modelo complexo concebido.

4.3. Medidas de Avaliação

As medidas utilizadas na área de Extração da Informação e, conseqüentemente, na atividade de *NER*, foram definidas no *MUC-2* (GRISHMAN; SUNDHEIM, 1996). São essencialmente duas: *Recall* (Abrangência) e *Precision* (Precisão). A utilização preferencial da denotação em inglês se faz porque é como tem sido empregado em textos na língua portuguesa. Tais medidas pertencem originalmente à área de Recuperação da Informação, tendo sido descritas pela primeira vez por (RIJSBERGEN, 1979).

As duas medidas utilizadas são definidas da seguinte forma:

$$Recall = \frac{\text{nº de entidades recuperadas corretamente}}{\text{total de entidades no corpus}} \quad (4.2)$$

$$Precision = \frac{\text{nº de entidades recuperadas corretamente}}{\text{nº de entidades recuperadas}} \quad (4.3)$$

Como há situações extremas em que uma medida está muito grande e a outra, muito pequena, toma-se também uma medida que fornece um valor condensado das duas, resultando em grandes valores quando ambas estão balanceadas. Esta corresponde à *F-Measure* (Medida-F) (RIJSBERGEN, 1979):

$$F - Measure = \frac{(\phi^2 + 1) \times Precision \times Recall}{\phi^2 \times Precision + Recall} \quad (4.4)$$

em que ϕ é um valor entre 0 e ∞, de modo a dar pesos variados a cada uma das medidas. Neste trabalho, adota-se $\phi = 1$, caso em que a *F-Measure* é a média harmônica entre *Recall* e *Precision*, fornecendo a mesma importância a ambas:

$$F - Measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (4.5)$$

Adota-se também neste trabalho a medida de taxa de erro sugerida em (WEN, 2001), que fornece um indicador da qualidade da classificação:

$$\text{Taxa de Erro} = \frac{\text{nº de entidades recuperadas de forma incorreta}}{\text{total de entidades no corpus}} \quad (4.6)$$

Se um sistema identificar o mesmo número de entidades presentes em um dado *corpus*, as medidas *Recall* e *Precision* tornam-se iguais a 1 menos a taxa de erro. Um sistema de identificação perfeito obteria zero na taxa de erro e 100% para *Recall* e *Precision* (WEN, 2001).

4.4. Testes Efetuados

Os dois textos que constituem os *corpora* foram utilizados em treinamento e avaliação. O procedimento adotado para testes se constituiu em duas etapas:

Etapa 1 – Treinamento com o *corpus* 1 e avaliação com o *corpus* 2; e

Etapa 2 – Treinamento com o *corpus* 2 e avaliação com o *corpus* 1.

Foram tentadas três combinações diferentes de treinamento para ambos os modelos: (1) aprendizado supervisionado através de contagens obtidas nos textos, conforme descrito na seção 3.3; (2) aplicação apenas do algoritmo de *Baum-Welch*, vide seção 3.3; e (3) aprendizado supervisionado primeiro e aplicação do algoritmo de *Baum-Welch* posteriormente.

O treinamento que obteve o melhor resultado para o Modelo Simples foi o (3), em que uma iteração do algoritmo de *Baum-Welch* melhorou ligeiramente os escores obtidos primeiramente com o aprendizado supervisionado. Da segunda iteração em diante, os resultados foram degradados.

Para o Modelo de Ordem Dois, o treinamento que obteve os melhores escores foi o (1). O treinamento (2) não atingiu escores satisfatórios, enquanto no (3) a aplicação do algoritmo de *Baum-Welch* degradou os resultados já a partir da primeira iteração, de modo que o modelo perdeu a capacidade de generalização, tendo ocorrido supertreinamento (*overtraining*). Esse resultado era esperado, na medida em que Schütze e Manning (SCHÜTZE; MANNING, 1999) já apontam para a ineficiência do algoritmo de *Baum-Welch* quando aplicado a problemas de etiquetação de textos.

Para mensurar a qualidade dos modelos, utilizaram-se as quatro medidas descritas na seção 4.3, ou seja, *Recall*, *Precision*, *F-Measure* e Taxa de Erro. Por exemplo,

Marcos Borges vai viajar para São Paulo em o dia 23 de março .

cuja seqüência correta de etiquetas correspondentes às entidades é:

PESSOA PESSOA O O O LOCAL LOCAL O O TEMPO TEMPO TEMPO TEMPO O

Suponha a seguinte saída do modelo implementado:

PESSOA PESSOA O O O PESSOA LOCAL O O O O O TEMPO O

Assim, obtêm-se as seguintes estatísticas:

1) entidades identificadas = 5 ;
2) entidades corretas = 4 ; e

3) entidades existentes $= 8$.

Logo, tem-se o seguinte valor para as medidas:

- $Precision = \frac{4}{5} = 0{,}8$;
- $\text{Re}\,call = \frac{4}{8} = 0{,}5$;
- $F - Measure = \frac{2 \times 0{,}8 \times 0{,}5}{0{,}8 + 0{,}5} = 0{,}62$; e
- $\text{Taxa de Erro} = \frac{5 - 4}{8} = 0{,}13$.

Para calcular o tempo de execução dos modelos, foi utilizada a função *clock()* da biblioteca *ctime* (*cplusplus.com*) que retorna o número de ciclos de *clock* passados desde o início da execução do programa.

O ambiente computacional utilizado nos testes realizados segue abaixo:

1) *Hardware*: Intel Core2Duo CPU @ 2.33Ghz com 2048MB DDR2 de memória;
2) Sistema Operacional: Windows XP Service Pack 3;
3) Linguagem de Programação: C++;
4) Ambiente de Desenvolvimento: Dev-C++ versão 4.9.9.2; e
5) Compilador: gcc/g++ versão 3.4.2.

4.5. Avaliação dos Modelos

Os experimentos foram conduzidos com quatro implementações: (1) **Modelo Simples**; (2) **Modelo de Ordem Dois**; (3) **Apenas *Gazetteer***; e (4) **Modelo de Ordem Dois com *Gazetteer***.

O Quadro 4.3 exibe, para cada uma das implementações, a quantidade de entidades identificadas e corretas na ETAPA 1 do procedimento de testes, conforme descrito na seção 4.4. Note-se que, para o caso de Apenas *Gazetter*, o número de entidades presentes no texto se reduz, na medida em que são levadas em consideração apenas PESSOA, LOCAL e ORGANIZAÇÃO.

Analogamente, o Quadro 4.4 exibe o quantitativo de entidades identificadas e corretas obtido na ETAPA 2 do procedimento de testes.

Quadro 4.3 – Resultado absoluto da ETAPA 1.

	Modelo Simples	**Modelo de Ordem Dois**	**Apenas *Gazetteer***	**Modelo de Ordem Dois com *Gazetteer***
Entidades Identificadas	6311	8930	4795	8944
Entidades Corretas	2424	3525	1690	3194
Entidades Presentes no *Corpus* de Avaliação	7340	7340	4514	7340
Tempo de Processamento (s)	1,28	26,48	329,92	365,08

Quadro 4.4 – Resultado absoluto da ETAPA 2.

	Modelo Simples	**Modelo de Ordem Dois**	**Apenas *Gazetteer***	**Modelo de Ordem Dois com *Gazetteer***
Entidades Identificadas	8275	11761	6798	11782
Entidades Corretas	3078	4458	2043	3928
Entidades Presentes no *Corpus* de Avaliação	10188	10188	6452	10188
Tempo de Processamento (s)	1,89	38,81	439,47	507,22

Nos quadros acima, destaca-se o fato de que o modelo Apenas *Gazetteer* conseguiu identificar mais entidades do que, de fato, existem nos *corpora*. Isso se dá pelo fato de que alguns nomes comuns foram erroneamente marcados como nomes próprios pelo etiquetador morfossintático que pré-processou os *corpora*. Assim,

como o modelo implementado se baseia na presença das etiquetas NPROP para buscar por entidades, alguns *tokens* foram erroneamente classificados como entidades.

Por fim, o Quadro 4.5 expõe, para cada implementação, o valor das medidas de avaliação aplicadas aos resultados expostos nos quadros acima, em que a média da ETAPA 1 e da ETAPA 2 foi empregada. Destacado em negrito, para cada linha, o modelo que obteve o melhor desempenho na referida medida.

Quadro 4.5 – Avaliação dos modelos.

	Modelo Simples	**Modelo de Ordem Dois**	**Apenas *Gazetteer***	**Modelo de Ordem Dois com *Gazetteer***
***Recall* (%)**	31,62	**45,89**	34,55	41,04
***Precision* (%)**	37,80	**38,69**	32,65	34,53
***F-Measure* (%)**	34,43	**41,98**	33,57	37,49
Taxa de Erro (%)	**51,98**	72,66	71,24	77,71
Tempo de Processamento (s)	1,59	32,65	384,70	436,15

O modelo que obteve os melhores escores para as medidas *Recall*, *Precision e F-Measure* foi o de Ordem Dois. Entretanto, obteve uma taxa de erro relativamente alta, de 72,66%, indicando que muitos *tokens* foram erroneamente identificados como entidades mencionadas e outros foram associados às classes erradas. Da mesma forma, as taxas de erro para os modelos Apenas *Gazetteer* e de Ordem Dois com *Gazetteer* também foram altas, ficando acima de 70%. Nesse sentido, o modelo que obteve a menor taxa de erro foi o Modelo Simples, pelo baixo número de entidades identificadas. A taxa de erro é uma medida que premia a ausência de identificações errôneas, sendo nula quando não há alguma.

O uso do *gazetteer* piorou o desempenho do Modelo de Ordem Dois, pois diminuiu o número de entidades identificadas corretamente, o que era esperado na medida em que o modelo *Apenas Gazetteer* teve uma precisão de apenas 32,65%.

Os tempos de processamento estão dentro do esperado, sendo pouco mais do que 1 segundo para o Modelo Simples e aproximadamente 0,5 minuto para o Modelo de Ordem Dois, já que, além do maior número de estados e emissões, utiliza o algoritmo de *Viterbi* com logaritmo das probabilidades, o que necessita de sucessivas chamadas à função logaritmo e, conseqüentemente, mais processamento.

O tempo de execução do modelo Apenas *Gazetteer* ficou elevado porque, para cada *token*, é feita uma busca na lista de entidades por uma entrada que o contenha como subcadeia. Assim, o Modelo de Ordem Dois que o utiliza teve tempos de execução também elevados.

Capítulo 5 – Conclusão

5.1. Considerações Finais

Apresentou-se neste trabalho uma formalização da teoria de *Hidden Markov Models*, que ganhou destaque como objeto de estudo e aplicação ao longo da década de 80, principalmente na área de reconhecimento da linguagem falada. Posteriormente, na década de 90, a teoria passou a ser empregada com sucesso em atividades de processamento de linguagem natural.

Com a motivação do sucesso deste emprego de *HMMs*, implementações para o problema de Reconhecimento de Entidades Mencionadas – REM – foram realizadas neste trabalho. Foram implementados dois modelos de *HMM*: um de primeira ordem e outro que realiza uma simulação de um modelo de segunda ordem através das etiquetas.

O Modelo de Primeira Ordem alcançou um desempenho ruim, obtendo baixos escores para as medidas utilizadas, com a exceção da taxa de erro, que é uma medida muito conservadora. Entretanto, sua implementação serviu de aprendizado para o desenvolvimento do modelo de segunda ordem, mais complexo.

O Modelo de Ordem Dois obteve o melhor desempenho, alcançando aproximadamente 42% de *F-Measure*. O uso do *gazetteer* REPENTINO piorou esse escore. Ainda assim, é um número relativamente baixo perto de algumas implementações existentes na literatura, que apresentam valores da ordem de 85%.

Os *corpora* utilizados, únicos em língua portuguesa, são pequenos (98810 e 66757 *tokens*), devendo-se ainda proceder ao incremento de seu volume, a fim de auxiliar no treinamento e na validação de outros modelos para *NER*. Da mesma forma, o *gazetter* REPENTINO ainda contém uma quantidade pequena de entidades nomeadas, devendo ser expandido. Estes dois fatores podem contribuir para uma melhoria considerável dos resultados.

5.2. Trabalhos Futuros

Algumas oportunidades de trabalhos futuros na utilização da abordagem *HMM* ou de outras técnicas para *NER* são:

- Estudo aprofundado das adaptações na modelagem e no algoritmo de *Viterbi* para um modelo de ordem dois real conforme descrito em (THEDE; HARPER, 1999). Análise desses resultados obtidos, que parecem ser promissores e não consta na literatura trabalho que o realize para *NER*;
- Utilização de outras propriedades de palavras (*token features*) e não apenas etiquetas morfossintáticas, na modelagem *HMM*, conforme trabalho realizado por (BIKEL et al., 1997);
- Implementação de *HMM* com uso de estados dinâmicos ou, ainda, de ordem variável (DUARTE; MILIDIÚ, 2007), (SCHÜTZE; MANNING, 1999) e (KEPLER, 2005);
- Uso do **algoritmo A*** para resolução do Problema da Seqüência de Estados Ótima (seção 3.2) de *HMM* (JURAFSKY; MARTIN, 2000); e
- Implementação de outras abordagens ao problema, como *Support Vector Machines* e *Bootstrapping* (FELDMAN; SANGER, 2007).

REFERÊNCIAS BIBLIOGRÁFICAS

1. BIKEL, Daniel et al. *Nymble:* A High-Performance Learning Name-Finder. In: Proceedings of the 5th Applied Natural Language Processing, Washington, 1997, 194-201 p.

2. BILISOLY, Roger. *Practical Text Mining with Perl*. Nova Jersey: John Wiley & Sons, 2008. 295 p.

3. CARDOSO, Nuno. *Avaliação de Sistemas de Reconhecimento de Entidades Mencionadas*. 2006. 126f. Dissertação de Mestrado em Inteligência Artificial e Sistemas Inteligentes, Faculdade de Engenharia da Universidade do Porto, Porto, Portugal.

4. CIMIANO, Philipp. *Ontology Learning and Population from Text*: Algorithms, Evaluation and Applications. Nova Iorque: Springer, 2006. 347 p.

5. CORMEN, Thomas et al. *Introduction to Algorithms*. 2ª ed. Massachusetts: McGraw-Hill, 2005. 323-369 e 906-932 p.

6. CUNHA, Anderson Mayrink. *Hidden Markov Models*. Relatório Técnico, Instituto de Matemática Pura e Aplicada, 2002.

7. DUARTE, Julio; MILIDIÚ, Ruy. *Machine Learning Algorithms for Portuguese Named Entity Recognition*. Monografias em Ciência da Computação, Departamento de Informática, PUC-Rio, 2007.

8. FELDMAN, Ronen; SANGER, James. *The Text Mining Handbook*: Advanced Approaches in Analyzing Unstructured Data. Nova Iorque: Cambridge University Press, 2007. 410 p.

9. FERREIRA, José. *Algoritmos de Programação Dinâmica Usados em Modelos Markovianos Ocultos (HMMs)*. 2005. 252f. Tese de Doutorado do Curso de Pós-Graduação em Computação Aplicada, Instituto Nacional de Pesquisas Espaciais, São José dos Campos, São Paulo.

10. GOMES, Geórgia. *Integrando Repositórios de Sistemas de Bibliotecas Digitais e Sistemas de Aprendizagem*. 2006. 144f. Tese de Doutorado do Programa de Pós-Graduação em Informática, PUC-Rio, Rio de Janeiro, Rio de Janeiro.

11. GRISHMAN, Ralph; SUNDHEIM, Beth. *Message Understanding Conference – 6*: A Brief History. In: Proceedings of the 16th International Conference on Computational Linguistics, COLING, Kopenhagen, 1996, 466-471 p.

12. JACKSON, Peter; MOULINIER, Isabelle. *Natural Language Processing for Online Applications*: Text Retrieval, Extraction and Categorization. Amsterdã: John Benjamins, 2002. 226 p.

13. JURAFSKY, Daniel; MARTIN, James. *Speech and Language Processing*: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition. Nova Jérsei: Prentice-Hall, 2000. 950 p.

14. KEPLER, Fábio. *Um Etiquetador Morfo-Sintático Baseado em Cadeias de Markov de Tamanho Variável*. 2005. 70f. Dissertação de Mestrado do Programa de Pós-Graduação em Ciência da Computação, Universidade de São Paulo, São Paulo, Brasil.

15. KONCHADY, Manu. *Text Mining Application Programming*. Massachusetts: Charles River Media, 2006. 412 p.

16. LINGUATECA. *Centro de Recursos para o Processamento Computacional da Língua Portuguesa*. Disponível em: http://www.linguateca.pt/. Acesso em: set. 2008.

17. MEDEIROS, Ivo. *Um Etiquetador Morfossintático Híbrido Baseado em Modelos de Markov Escondidos e Aprendizado Baseado em Transformações para o Português do Brasil*. 2008. 67f. Trabalho de Conclusão de Curso do Bacharelado em Engenharia da Computação, Faculdade de Engenharia da Computação, Universidade Federal do Pará, Pará, Brasil.

18. MIKHEEV, Andrei; MOENS, Marc; GROVER, Claire. *Named Entity Recognition without Gazetteers*. In: Proceedings of the 19th European Chapter of Association for Computational Linguistics, Bergen, 1999.

19. MOTTA, Eduardo. *Reconhecimento de Entidades Mencionadas Utilizando HMM – Estudo de Modelagens*. Monografias em Ciência da Computação, Departamento de Informática, PUC-Rio, 2008.

20. NADEAU, David. *Semi-Supervised Named Entity Recognition:* Learning to Recognize 100 Entity Types with Little Supervision. 2007. 150f. Tese de Doutorado em Ciência da Computação, Escola de Tecnologia da Informação e Engenharia, Universidade de Ottawa, Ottawa, Canadá.

21. NUGUES, Pierre. *An Introduction to Language Processing with Perl and Prolog*: An Outline of Theories, Implementation, and Application with Special Consideration of English, French, and German. Berlim: Springer, 2006. 513 p.

22. PRADO, Hercules; FERNEDA, Edilson. *Emerging Technologies of Text Mining*: Techniques and Applications. Nova Iorque: Information Science Reference, 2008. 358 p.

23. RABINER, Lawrence. *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*. In: Proceedings of the IEEE, vol. 77, no. 2, 1989, 257-286.

24. *REPENTINO – REPositório para reconhecimento de ENTidades com NOme*. Linguateca, 2005. Disponível em: http://poloclup.linguateca.pt/repentino/. Acesso em: nov. 2008.

25. RIJSBERGEN, C. J. Van. *Information Retrieval*. 2ª ed. Londres: Butterworths, 1979. 147 p.

26. SANTOS, Diana; CARDOSO, Nuno. *A Golden Resource for Named Entity Recognition in Portuguese*. In: Proceedings of the 7th Workshop on Computational Processing of Written and Spoken Portuguese, PROPOR, Itatiaia, Rio de Janeiro, 2006. 69-79 p.

27. SANTOS, Diana et al. *HAREM: An Advanced NER Evaluation Contest for Portuguese*. In: Proceedings of the 5th International Conference on Language Resources and Evaluation, LREC, Gênova, Itália, 2006. 1986-1991 p.

28. SANTOS, Diana et al. *HAREM:* Avaliação de sistemas de Reconhecimento de Entidades Mencionadas. Disponível em: <http://poloxldb.linguateca.pt/harem.php>. Acesso em: set. 2008.

29. SARMENTO, Luís; PINTO, Ana Sofia; CABRAL, Luís. *REPENTINO – A Wide-Scope Gazetteer for Entity Recognition in Portuguese*. In: Proceedings of the 7th International Workshop on Computational Processing of the Portuguese Language, PROPOR 2006, Itatiaia, Brasil, 2006, 31-40 p.

30. SCHÜTZE, Hinrich; MANNING, Christopher. *Foundations of Statistical Natural Language Processing*. Massachusetts: Massachusetts Institute of Technology, 1999. 680 p.

31. SEDGEWICK, Robert. *Algorithms*. Massachusetts: Addison-Wesley, 1984. 241-255 p.

32. THEDE, Scott; HARPER, Mary. *A Second-Order Hidden Markov Model for Part-of-Speech Tagging*. In: Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics, 1999. 175-182 p.

33. VITERBI, Andrew James. *Error bounds for convolutional codes and an asymptotically optimum decoding algorithm*. In: IEEE Transactions on Information Theory, IT-13(2), 1967, 260-269.

34. WEN, Yingying. *Text Mining Using HMM and PPM*. 2001. 90f. Dissertação (Mestrado em Ciência da Computação) – Departamento de Ciência da Computação, Universidade de Waikato, Hamilton, Nova Zelândia.

35. Entrada para a função *strstr()* em *cplusplus.com – The C++ Resource Network*. Disponível em: http://www.cplusplus.com/reference/clibrary/cstring/strstr.html. Acesso em: jan. 2009.

36. Entrada para a função *clock()* em *cplusplus.com – The C++ Resource Network*. Disponível em: http://www.cplusplus.com/reference/clibrary/ctime/clock.html. Acesso em: jan. 2009.

APÊNDICES

Apêndice A - Classes de entidades mencionadas e mapeamento realizado

As etiquetas das classes de entidades mencionadas presentes nos *corpora* e o mapeamento realizado segue no Quadro I.

Quadro I – Classes de entidades mencionadas e mapeamento adotado (adaptado de MOTTA, 2008).

Etiqueta de Entidade Mencionada	Mapeamento
PESSOA	PESSOA
LOCAL	LOCAL
ORGANIZACAO	OUTRA
TEMPO	TEMPO
VALOR	VALOR
OBRA	OUTRA
ABSTRACCAO	OUTRA
COISA	OUTRA
ACONTECIMENTO	OUTRA
OUTRO	OUTRA
OBJECTO	OUTRA
O (Não-Entidade Mencionada)	O

Apêndice B - Mapeamento realizado para as etiquetas morfossintáticas

O mapeamento realizado para as etiquetas morfossintáticas segue no Quadro II.

Quadro II – Mapeamento para as etiquetas morfossintáticas (adaptado de MOTTA, 2008).

Etiqueta Morfossintática	**Mapeamento**
.	PONTUACAO
...	PONTUACAO
?	PONTUACAO
!	PONTUACAO
:	PONTUACAO
,	PONTUACAO
;	PONTUACAO
"	DELIMITADOR
'	DELIMITADOR
(	DELIMITADOR
)	DELIMITADOR
-	DELIMITADOR
/	DELIMITADOR
=	DELIMITADOR
[	DELIMITADOR
]	DELIMITADOR
`	DELIMITADOR
ADJ	ADJ
ADV	ADV
ADV-KS	ADV
ADV-KS-REL	ADV
ART	ART

CUR	CUR
IN	IN
KC	KC
KS	KS
N	N
NPROP	NPROP
NUM	NUM
PCP	PCP
PDEN	PDEN
PREP	PREP
PRO-KS	PRONOME
PRO-KS-REL	PRONOME
PROADJ	PRONOME
PROPESS	PRONOME
PROSUB	PRONOME
V	VERBO
VAUX	VERBO

ANEXOS

Anexo A - Etiquetas morfossintáticas presentes nos *corpora*

O conjunto de etiquetas morfossintáticas utilizadas nos *corpora* segue no Quadro III.

Quadro III – Etiquetas morfossintáticas dos *corpora*.

Classe Gramatical	**Etiqueta**
Adjetivo	**ADJ**
Advérbio	**ADV**
Advérbio Conectivo Subordinativo	**ADV-KS**
Advérbio Relativo Subordinativo	**ADV-KS-REL**
Artigo (definido ou indefinido)	**ART**
Conjunção Coordenativa	**KC**
Conjunção Subordinativa	**KS**
Interjeição	**IN**
Nome	**N**
Nome Próprio	**NPROP**
Numeral	**NUM**
Particípio	**PCP**
Palavra Denotativa	**PDEN**
Preposição	**PREP**
Pronome Adjetivo	**PROADJ**
Pronome Conectivo Subordinativo	**PRO-KS**
Pronome Pessoal	**PROPESS**
Pronome Relativo Conectivo Subordinativo	**PRO-KS-REL**
Pronome Substantivo	**PROSUB**
Verbo	**V**

Verbo Auxiliar	**VAUX**
Símbolo de Moeda Corrente	**CUR**

Anexo B - Fragmento do *corpus* do *MiniHAREM*

Neste anexo, é exibido um fragmento do *corpus* 2, referente à coleção dourada do *MiniHAREM*, com a etiquetação morfossintática. O texto está separado em sentenças e estas divididas em *tokens*. Cada linha está estruturada da seguinte forma:

<*Token*> <*Classe de Entidade Mencionada*> <*Tipo*> <*POS Tag*>

Assim, a terceira etiqueta, por exemplo, INDIVIDUAL, corresponde ao tipo, que é uma subcategoria da entidade mencionada. Tais etiquetas não foram consideradas neste trabalho. Quando um *token* não corresponde a uma entidade mencionada, suas etiquetas de entidade e de tipo são designadas por O. Para caracteres de pontuação e delimitadores, não foi considerada uma classe gramatical específica, havendo a repetição do próprio caractere na quarta etiqueta (*POS Tag*). Entretanto, isso foi contornado através do mapeamento realizado no Apêndice B.

W. PESSOA INDIVIDUAL NPROP
JAMES PESSOA INDIVIDUAL NPROP

Willian PESSOA INDIVIDUAL N
James PESSOA INDIVIDUAL NPROP

Willian PESSOA INDIVIDUAL NPROP
James PESSOA INDIVIDUAL NPROP
, O O ,
filósofo O O N
e O O KC
psicólogo O O N
. O O .

Foi O O V
o O O ART
mais O O ADV
influente O O ADJ
de O O PREP
os O O ART
pensadores O O N
de O O PREP
os O O ART
EUA LOCAL HUMANO NPROP
, O O ,
criador O O N

de O O PREP
o O O ART
pragmatismo O O N
. O O .

Nasceu O O V
e O O KC
, O O ,
Nova LOCAL HUMANO NPROP
Iorque LOCAL HUMANO NPROP
, O O ,
a O O ART
11 TEMPO DATA NPROP
de TEMPO DATA NPROP
Janeiro TEMPO DATA NPROP
de TEMPO DATA PREP
1842 TEMPO DATA N
. O O .

O O O ART
seu O O PROADJ
pai O O N
, O O ,
Henru PESSOA INDIVIDUAL NPROP
James PESSOA INDIVIDUAL NPROP
, O O ,
era O O V
um O O ART
teólogo O O N
seguidor O O ADJ
de O O PREP
Emanuel PESSOA|ABSTRACCAO INDIVIDUAL|ABSTRACCAO NPROP
Swedenborg PESSOA|ABSTRACCAO INDIVIDUAL|ABSTRACCAO NPROP
. O O .

Um O O PROSUB
de O O PREP
os O O ART
seus O O PROADJ
irmãos O O N
foi O O V
o O O ART
conhecido O O PCP
novelista O O ADJ
Henry PESSOA INDIVIDUAL NPROP
James PESSOA INDIVIDUAL NPROP
. O O .

Concluiu O O V
os O O ART
seus O O PROADJ
estudos O O N
de O O PREP
medicina O O N
, O O ,
em O O PREP
1870 TEMPO DATA N
, O O ,
em O O PREP
a O O ART
Universidade ORGANIZACAO INSTITUICAO NPROP
de ORGANIZACAO INSTITUICAO NPROP
Harvard ORGANIZACAO INSTITUICAO NPROP
, O O ,
onde O O ADV-KS-REL
iniciou O O V
a O O ART
sua O O PROADJ
carreira O O N
como O O PREP
professor O O N
de O O PREP
fisiologia O O N
em O O PREP
1872 TEMPO DATA N
. O O .

Anexo C - Fragmento do *gazetteer* REPENTINO

Neste anexo, é exposto um fragmento do arquivo *XML* de definição do *gazetteer* REPENTINO, exibindo apenas uma pequena parte das categorias de entidades utilizadas (LOC para um local, ORG para uma organização e SER para uma pessoa). O atributo *subcat*, que indica uma subcategoria, não foi utilizado neste trabalho.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE REPENTINO SYSTEM "./repentino.dtd">
<REPENTINO ver="1.0" data="25-3-2006">
<LOC>
        <EN_LOC subcat="TERR">alto da Serra de Meda</EN_LOC>
        <EN_LOC subcat="TERR">alto da serra do Caramulo</EN_LOC>
        <EN_LOC subcat="TERR">alto do monte de Santa Maria</EN_LOC>
        <EN_LOC subcat="TERR">alto do morro da Sé</EN_LOC>
        <EN_LOC subcat="TERR">Alto Douro</EN_LOC>
        <EN_LOC subcat="TERR">Alto Tâmega</EN_LOC>
        <EN_LOC subcat="TERR">América</EN_LOC>
        <EN_LOC subcat="TERR">América do Sul</EN_LOC>
        <EN_LOC subcat="TERR">América Latina</EN_LOC>
        <EN_LOC subcat="TERR">Antártida</EN_LOC>
        <EN_LOC subcat="TERR">arquipélago da Madeira</EN_LOC>
        <EN_LOC subcat="TERR">arquipélago das Açores</EN_LOC>
        <EN_LOC subcat="TERR">arquipélago das Baleares</EN_LOC>
        ...
</LOC>
<ORG>
        <EN_ORG subcat="EMPR">20th Century Fox</EN_ORG>
        <EN_ORG subcat="EMPR">Palimage</EN_ORG>
        <EN_ORG subcat="EMPR">Quasi</EN_ORG>
        <EN_ORG subcat="EMPR">Adega Cooperativa Almeirim</EN_ORG>
        <EN_ORG subcat="EMPR">Adega Cooperativa Beira Serra</EN_ORG>
        <EN_ORG subcat="EMPR">Adega Cooperativa Borba</EN_ORG>
        <EN_ORG subcat="EMPR">Adega Cooperativa Cantanhede</EN_ORG>
        <EN_ORG subcat="EMPR">Adega Cooperativa Cartaxo</EN_ORG>
        <EN_ORG subcat="EMPR">Adega Cooperativa Covilha</EN_ORG>
        <EN_ORG subcat="EMPR">Adega Cooperativa da Azueira</EN_ORG>
        <EN_ORG subcat="EMPR">Adega Cooperativa da Batalha</EN_ORG>
        <EN_ORG subcat="EMPR">Adega Cooperativa da Carvoeira</EN_ORG>
        <EN_ORG subcat="EMPR">Adega Cooperativa da Carvoeira Bela
Fonte</EN_ORG>
        ...
</ORG>
<SER>
```

<EN_SER subcat="HUM">Abílio Albino As Silva Nunes</EN_SER>
<EN_SER subcat="HUM">Abdul</EN_SER>
<EN_SER subcat="HUM">Abel De Pinho Soares</EN_SER>
<EN_SER subcat="HUM">Abel Feldmann Da Câmara Carreiro</EN_SER>
<EN_SER subcat="HUM">Abel Fernando Queiros Figueiredo</EN_SER>
<EN_SER subcat="HUM">Abraham Lincoln</EN_SER>
<EN_SER subcat="HUM">Achille Talon</EN_SER>
<EN_SER subcat="HUM">Adalberto Alves</EN_SER>
<EN_SER subcat="HUM">Adalberto Nuno da Silva Leite de Freitas</EN_SER>
<EN_SER subcat="HUM">Adalberto Nuno de Silva Leite de Freitas</EN_SER>
<EN_SER subcat="HUM">Adélio Amaro</EN_SER>
<EN_SER subcat="HUM">Adília Lopes</EN_SER>
<EN_SER subcat="HUM">Adelaide Rosa Coelho Teles Madureira</EN_SER>
...
</SER>
</REPENTINO>