Treinamento discriminativo de HMMs contínuos para reconhecimento de palavras isoladas

Amarildo Martins de Mattos

Dissertação apresentada ao Instituto Nacional de Telecomunicações, como parte dos requisitos para obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Prof. Dr. Carlos Alberto Ynoguti

Santa Rita do Sapucaí
2003

Dissertação defendida e aprovada em 17/10/2003, pela comissão julgadora:

Prof. Dr. Carlos Alberto Ynoguti - DTE / INATEL

Prof. Dr. Fábio Violaro - DECOM / UNICAMP

Prof. Dr. Francisco José Fraga da Silva - DTE / INATEL

Coordenador do Curso de Mestrado

Prof. Dr. Adonias Costa da Silveira

Ao meus pais Urias Leite da Cunha Mattos (*in memoriam*), Dalva Martins da Cunha Mattos e meus irmãos Abigail, Angelo, Augusto e João, e também aos meus sobrinhos Albert e Alberto, que são aqueles que me motivam a buscar novos ideais e alcançar todos os meus objetivos.

Agradecimentos

Ao Professor Doutor Carlos Alberto Ynoguti pela excelente orientação fornecida durante a elaboração deste trabalho. Neste período de orientação aprendi que ele, para mim, não era apenas um orientador e sim um AMIGO, um HOMEM e um PAI, que me ensinou não só sobre reconhecimento de fala mas também ver a tudo e a todos de outra maneira. Ensinou-me a ter PACIÊNCIA e escutar mais aos outros e hoje eu agradeço MUITO a ele, pois será alguém que em qualquer lugar ou momento, independentemente para onde a vida vai me levar, me lembrarei de todos os seus conselhos, das nossas conversas e também de que, se hoje estou apresentando esta dissertação, foi ele o meu grande incentivador.

Agradeço a Deus por ter me iluminado ao longo deste período e agradeço por todos os obstáculos que ele, sabiamente, colocou em meu caminho para que eu pudesse evoluir como pessoa e como profissional.

Aos colegas e amigos do laboratório de Pós-Graduação: Tião, Leandro, Edmilson, Egídio, Cris, Sílvio, Rodrigo, Carlos Paula, Daniela e sua filha Mariana, José, Fábio e Helder. Aos professores e funcionários do Instituto Nacional de Telecomunicações por proporcionarem um ambiente de trabalho muito agradável.

Agradeço aos amigos de final de semana que me incentivaram e que acompanharam a minha luta no desenvolvimento do trabalho e na minha vida pessoal durante a elaboração deste trabalho.

Às pessoas que emprestaram suas vozes na confecção da base de dados do INATEL.

E em especial ao INATEL e à ERICSSON, pela concessão da bolsa mensalidade e de manutenção.

Índice

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

ANN *Artificial Neural Network* - Rede Neural Artificial

DF *Discriminative Function* - Função Discriminativa

PDF *Probability Density Function* - Função Densidade de Probabilidade

PMF *Probability Mass Function* - Função Massa de Probabilidade

GPD *Generalized Probabilistic Descent*

HMM *Hidden Markov Model* - Modelo Oculto de Markov

MAP *Maximum a Posteriori Probability* - Máxima Probabilidade a Posteriori

MCE *Minimum Classification Error* - Erro Mínimo de Classificação

ML *Maximum Likelihood* - Máxima Verosimilhança

MMI *Maximum Mutual Information* - Máxima Informação Mútua

MSE *Minimum Square Error* - Mínimo Erro Quadrático

DT *Discriminative Training* - Treinamento Discriminativo

Lista de Símbolos

P	Probabilidade
Λ	Conjunto de modelos HMM's
λ_i	Modelo da palavra i
$g_i(X, q; \Lambda)$	Log-probabilidade de emissão da locução X associados a todos os modelos de palavra do vocabulário
T	Número de símbolos observados ou comprimento da sequência de observação
S	Sequência de estados, incluindo o estado inicial e o final
A	Matriz de função massa de probabilidade de transições
a_{ij}	Probabilidade de ocorrer uma transição entre os estados i e j
B	Matriz de função densidade de probabilidade de símbolos
$b_j(x_t)$	Probabilidade de ocorrer um símbolo x_t quando se atingir o estado j
N	Número de estados
Π	Matriz da função massa de probabilidade para o estado inicial
π_j	Probabilidade de o processo iniciar-se no estado j
η	Constante de overflow
γ	Limiar de overflow
$X_{treinamento}$	Conjunto de treinamento
σ^2	Variância
σ	Desvio padrão
D_i	Função erro
l_i	Função custo
I	Função Indicadora
ε	Passo de aprendizagem

∇	Gradiente
M	Número de gaussianas
c_{jm}	Coeficiente de ponderação
c_{jm}^{i}	Coeficiente de ponderação do modelo da palavra i
$N(x_t, \mu_{jm}, U_{jm})$	Função densidade de probabilidade multidimensional
μ_{jm}	Média
U_{jm}	Matriz de covariância
dim	Dimensão do vetor O_t
$\|W_{jm}\|$	Determinante da matriz de covariância
$\overline{a}_{ij}$	Probabilidade de transição para os modelos irrestritos
$\overline{c}_{jm}$	Coeficiente de ponderação para os modelos irrestritos
$\overline{\mu}_{jm}$	Média das gaussianas para os modelos irrestritos
$\overline{\sigma}_{jm}$	Desvio padrão das gaussianas para os modelos irrestritos
ϵ_2	Constante

Resumo

Tradicionalmente, para a aplicação de reconhecimento de fala, usa-se o critério de maximização da verossimilhança da locução dado o modelo. Desta forma, o algoritmo de treinamento maximiza a probabilidade de o modelo correto gerar a sequência de eventos acústicos correspondente à locução de entrada. Nos últimos anos vem tendo destaque uma forma alternativa de treinamento, baseada diretamente na minimização do erro de reconhecimento, ou mais formalmente, no critério de minimização do erro de classificação (MCE - *Mininum Classification Error*) para reestimação dos parâmetros dos modelos ocultos de Markov (Hidden Markov Models - HMMs). Este trabalho tem por objetivo o estudo teórico deste método, bem como uma avaliação de seu desempenho quando comparado com o critério ML tradicional. Os testes foram realizados em um cenário de reconhecimento de palavras isoladas, vocabulário pequeno, HMMs contínuos e parâmetros mel-cepstrais.

Palavras chave: reconhecimento de fala, reconhecimento de palavras isoladas, modelos ocultos de Markov, treinamento discriminativo, critério MCE.

Abstract

Traditionally, the maximum likelihood of the observation sequence given the model is the most widely used criterion for training the Hidden Markov Models (HMMs). The algorithm based on this criterion maximizes the probability of the model to generate the observation sequence. In the last years, some new alternative methods were developed in order to directly minimize the recognition error. In a more formal fashion, this methods rely on the Minimum Classification Error (MCE) criterion for the reestimation of the HMM parameters. The goal of this work is to make a theoretical study of this method, as well as a comparative performance evaluation between the MCE and the ML training methods. The tests were performed in an isolated word, small vocabulary scenario, using continuous HMMs and mel-cepstral parameters.

Keywords: speech recognition, isolated words speech recognition, hidden Markov models, discriminative training, Minimum Classification Error.

Capítulo 1

Introdução

1.1 A tecnologia de reconhecimento de fala no mundo atual

O amadurecimento da tecnologia de reconhecimento de fala vem gerando novas aplicações em várias áreas, tais como CRM (*Customer Relationship Management*), controle de algumas funções em automóveis, serviços telefônicos entre outros.

Uma aplicação que vem crescendo bastante atualmente é a de portais de voz (*speech portals*) [1], que consistem em viabilizar o acesso à Internet e a serviços de atendimento ao cliente através de comandos de voz, transmitidos via telefonia móvel e/ou via Internet. As vantagens são inúmeras, como, por exemplo, uma busca rápida por informações em listas e diretórios: é muito mais fácil e rápido um usuário dar um comando de voz do que digitar nomes usando as pequenas teclas dos celulares.

Entretanto, existem alguns desafios a serem vencidos. Primeiro, o reconhecimento de fala depende das condições ambientais (no caso de transmissão via celular) e da rede (no caso de transmissão via Internet). Se o canal apresentar um ruído muito alto ou houver muitos atrasos na rede de pacotes há o risco do sistema não identificar corretamente os comandos. Além disso o reconhecimento de voz exige uma capacidade de processamento bastante alta. O tempo de resposta da rede também deve ser quase instantâneo. De maneira geral, um indivíduo espera uma resposta em um diálogo de voz com o atraso menor que de 0,6 segundos . É o tempo de resposta típico em uma conversação normal entre duas pessoas ao telefone. Em segundo lugar, as pessoas gostariam de usar esta tecnologia como se estivessem conversando com alguém, no caso um atendente. O reconhecimento de fala espontânea é uma dos problemas mais difíceis de serem resolvidos, pois

as pessoas tendem a falar de forma relaxada, incompleta e, muitas vezes, de forma não objetiva. Some-se a isto as diferenças devido a regionalismos, estado emocional, grau de escolaridade e outros fatores, e tem-se então um retrato fiel das reais dificuldades com que se depara um sistema desses na prática.

Desta forma, várias empresas estão criando grupos de pesquisa e desenvolvimento na busca de novas tecnologias e montando uma estratégia de implementação de reconhecimento de fala dentro de uma abordagem consistente, com o objetivo de adotar soluções que convirjam para um padrão, mesmo que interno, que garanta a compatibilidade entre as aplicações. Neste sentido, algumas empresas de telecomunicações, dentre elas a Lucent e a Motorola, criaram o *VoiceXML Forum*, com o objetivo de padronizar os esforços nesta área, e permitir que conteúdos da Web sejam acessados por recursos de voz. A especificação 1.0 foi liberada em março de 2000 [2].

Com a evolução desta tecnologia, muitas empresas estão investindo milhares de dólares em pesquisas e lançamentos de produtos que tenham algum sistema de reconhecimento de voz. Deste modo, uma das principais empresas mundiais, a IBM, acabou de anunciar que forneceu uma avançada tecnologia de reconhecimento de voz para um novo e revolucionário sistema de navegação na Internet, que será uma característica dos modelos Honda Accord do próximo ano [3]. Com este sistema, os condutores podem fazer perguntas e obter respostas relativas a informações sobre rotas para um determinado destino.

Um outro produto da IBM é o software Via Voice no Brasil, que pode ser usado tanto para comandar algumas funções do microcomputador como para ditar cartas e editar documentos e tabelas[4].

Além da IBM, outras gigantes têm demonstrado interesse nesta área, como a Lucent Technologies, que anunciou este ano a criação da Speech Solutions, divisão para desenvolvimento de sistemas de reconhecimento de voz e a Corel, desenvolvedora do Corel Draw!, que estuda a possibilidade de incluir em seu editor de textos WordPerfect o módulo Dragon Dictate [6].

Nos últimos meses várias operadoras de telefones celulares do Brasil estão lançando serviços que utilizam a voz, por exemplo, a Telemig Celular lançou o VOZ [5]. Com ele os usuários pode acessar vários serviços utilizando apenas a voz, como: acessar e-mail, acessar a caixa postal, entre outros

Uma outra aplicação utilizando sistemas de reconhecimento de fala foi lançada em julho deste ano pela Universidade Federal do Rio de Janeiro (UFRJ): o Motrix [7, 8], que tem por objetivo facilitar o uso do computador por deficientes físicos. Segundo o coordenador do curso de fisioterapia do Instituto Brasileiro de Medicina de Reabilitação, José Francisco da Silva, "o computador permite que o deficiente físico envie e-mails, pague uma conta no banco e leia um jornal. Isso dá a ele maior capacidade de expressão e relacionamento social e ajuda no tratamento".

Cabe salientar que provavelmente as interfaces via voz vão ser uma tecnologia integrada e complementar às demais, pois sempre existirão funções que funcionarão de forma melhor através de um teclado, um mouse e um visor, e outras em que a utilização da voz é mais prática e conveniente.

1.2 Objetivos deste trabalho

Como visto anteriormente, com o crescimento da utilização de sistemas de reconhecimento de fala, muitas pesquisas estão sendo realizadas com o objetivo de melhorar o seu desempenho. Uma das tecnologias que vem tendo melhor desempenho nesta tarefa é a baseada em modelos ocultos de Markov - HMM (*Hidden Markov Models*), estruturas duplamente estocásticas que têm se mostrado eficientes para modelar, tanto as variabilidades acústicas como temporais do sinal de fala.

O critério de treinamento mais usual de um HMM baseia-se no método de máxima verossimilhança - ML (*Maximum Likelihood*), que procura maximizar a verossimilhança de um determinado modelo gerar uma dada sequência de observações.

Nos últimos anos, entretanto, os pesquisadores vêm tentando desenvolver métodos de treinamento que minimizem diretamente a taxa de erro de reconhecimento. Um critério resultante deste esforço é o de maximização da informação mútua - MMI (*Maximum Mutual Information*) [9]. De forma simplificada, pode-se dizer que, para uma dada sequência X de vetores acústicos, o critério MMI procura maximizar a verossimilhança do modelo M_i (modelo associado a X) e minimizar a verossimilhança dos modelos concorrentes M_j, na tentativa de melhorar a discriminabilidade entre estes modelos. Infelizmente o critério MMI não pode ser solucionado por análise direta ou por reestimação.

Entretanto, pode-se mostrar que o critério MMI é equivalente ao critério de maximização a posteriori - MAP (*Maximum a Posteriori Probabilities*), o qual é o mesmo método utilizado por uma rede neural artificial - ANN (*Artificial Neural Network*) otimizada a partir do critério de minimização do erro de classificação - MCE (*Minimum Classification Error*), este sim com uma solução analítica.

Desta forma, o objetivo principal deste trabalho é determinar o ganho que se obtém em se utilizar um algoritmo de treinamento baseado no critério de minimização da taxa de erros (MCE) em relação ao algoritmo tradicional, baseado no critério ML, para um sistema de reconhecimento de palavras isoladas baseado em HMMs contínuos.

Para isto, tomou-se por base um sistema de reconhecimento de palavras isoladas treinado segundo o critério ML, sistema este desenvolvido pelo Dr. José

Antônio Martins [10]. Depois foi implementado o algoritmo *Segmental Generalized Probabilistic Descent - GPD*, conhecido como algoritmo Segmental GPD, que é baseado no critério MCE. A medida de desempenho utilizada foi a taxa de acertos para os dois sistemas em uma tarefa de reconhecimento de palavras isoladas e vocabulário pequeno, utilizando modelos ocultos de Markov contínuos e parâmetros mel-cepstrais.

1.3 Estrutura da Dissertação

A dissertação está organizada da seguinte maneira:

No capítulo 2 é apresentada de forma breve a teoria relativa à técnica de Modelos Ocultos de Markov, bem como o algoritmo de reestimação dos parâmetros utilizando o critério ML.

O capítulo 3 possui uma descrição teórica sobre o algoritmo de reestimação dos parâmetros dos HMMs segundo o critério de minimização do erro de classificação - MCE. São demonstradas as equações de reestimação para todos os parâmetros: probabilidade de transição a_{ij}, coeficiente de ponderação c_{jm}^{i}, média das gaussianas μ_{jm}^{i} e variâncias das gaussianas σ_{jm}^{i}.

No capítulo 4 são descritas as base de dados e o ambiente de simulação utilizados para desenvolvimento do sistema.

No capítulo 5 são descritos alguns aspectos de implementação e os resultados das simulações. Também é realizado um estudo comparativo dos resultados apresentados pelos sistemas treinados através do método tradicional (ML) e do discriminativo (MCE).

No Capítulo 6 são apresentadas as conclusões gerais deste trabalho e também sugestões para próximas incursões nesta área.

Capítulo 2

Modelos Ocultos de Markov e o critério ML

A teoria relativa aos modelos ocultos de Markov já é bem conhecida e extensivamente documentada. Desta forma, nesta seção são apresentados apenas os conceitos básicos e notações importantes para a compreensão das seções posteriores. Uma boa referência sobre este assunto é [12].

2.1 Modelos Ocultos de Markov

Uma seqüência de valores s_t de uma variável aleatória discreta S_t caracteriza uma cadeia de Markov se:

$$P(S_{t+1} = s_{t+1}|S_t = s_t, S_{t-1} = s_{t-1}, ..., S_1 = s_1) = P(S_{t+1} = s_{t+1}|S_t = s_t) \quad (2.1)$$

i.e., a probabilidade de o processo passar ao estado $S_{t+1} = s_{t+1}$ depende apenas do estado presente ($S_t = s_t$), sendo independente dos estados anteriores.

Em um modelo oculto de Markov (HMM) existem dois processos estocásticos associados: o primeiro (observável), modela as observações de saída de cada estado através de uma função densidade de probabilidade (fdp); o segundo (oculto), modela as transições entre os estados (uma cadeia de Markov).

2.1.1 Elementos de um HMM

Em geral, para o reconhecimento de fala, utiliza-se um modelo simplificado conhecido como modelo *left-right*, ou modelo de Bakis [7]. Neste, exemplificado

na figura 2.1, são permitidas apenas transições para o mesmo estado, ou de um estado i para um estado j mais à direita ($i < j$).

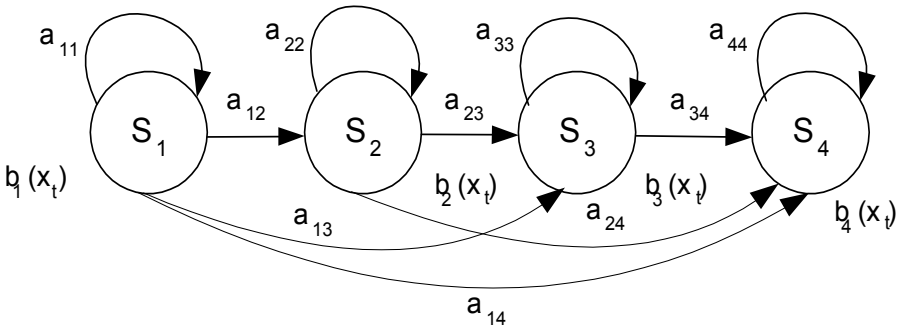

Figura 2.1: *Representação de um modelo oculto de Markov tipo left right.*

Nesta figura, podemos identificar:

- S_1, S_2, S_3, S_4: estados do modelo.
- a_{ij}: probabilidade de transição do estado i para o estado j.
- $b_j(x_t)$: probabilidade de emissão do símbolo x_t no estado j.

Para definir um HMM precisamos então dos seguintes parâmetros:

- Um conjunto de estados $S = S_1, S_2, S_3, \ldots, S_N$, incluindo um estado inicial S_1 e um estado final S_N.
- Uma matriz de transições A $= a_{ij}$, onde a_{ij} representa a probabilidade de se efetuar uma transição do estado i para o estado j, $1 \leq i, j \leq N$.
- Uma matriz de probabilidades de emissão de símbolo B $= b_j(x_t)$, onde $b_j(x_t)$ é a probabilidade de emitir um símbolo x_t quando se atingir o estado j, $1 \leq j \leq N$, $1 \leq t \leq T$ onde T é o o número de quadros da locução de entrada. A cada valor de t calcula-se um ou mais vetores de parâmetros a partir do sinal acústico.
- Uma matriz de função massa de probabilidade (fmp) do estado inicial $\Pi = \pi_j$, onde π_j é a probabilidade de o processo iniciar-se no estado j, $1 \leq j \leq N$.

A especificação do modelo pode ser descrita através da notação abreviada

$$\lambda = (A, B, \Pi) \tag{2.2}$$

Como os parâmetros A, B e Π são grandezas probabilísticas, as seguintes condições devem ser observadas.

$$a_{ij} > 0, \qquad \forall i, j \tag{2.3}$$

$$b_j(x_t) > 0, \qquad \forall j, t \tag{2.4}$$

$$\pi_j > 0, \qquad \forall j \tag{2.5}$$

$$\sum_{j=1}^{N} a_{ij} = 1, \qquad \forall i \tag{2.6}$$

$$\sum_{j=1}^{N} \pi_j = 1 \tag{2.7}$$

Para HMMs discretos temos:

$$\sum_{l=1}^{L} b_j(l) = 1, \qquad \forall j \tag{2.8}$$

onde L é o número de símbolos possíveis, que é igual a dimensão do codebook VQ.
e para HMMs contínuos:

$$\int_{-\infty}^{+\infty} b_j(x)dx = 1, \forall j \tag{2.9}$$

onde $b_j(x)$ é a função densidade de probabilidade de emissão do vetor de observação x no estado j, é dada por uma mistura de gaussianas da seguinte forma:

$$b_j^i(x_t) = \sum_{m=1}^{M} c_{jm} N(x_t, \mu_{jm}, U_{jm}) \quad 1 \leq j \leq N \tag{2.10}$$

onde:

- x_t é o vetor de entrada,
- M é o número de gaussianas,
- c_{jm} é o coeficiente da m-ésima gaussiana no estado S_j,
- $N(\cdot)$ é uma função densidade de probabilidade Gaussiana multidimensional com vetor média μ_{jm} e matriz de covariância U_{jm},
- N é o número de estados do modelo i.

A função densidade de probabilidade Gaussiana multidimensional diagonal é dada por

$$N(x_t, \mu_{jm}, U_{jm}) = \frac{1}{(2\pi)^{\frac{D}{2}} |W_{jm}|^{\frac{1}{2}}} \; e^{\left\{-\frac{1}{2}\sum_{l=1}^{D}\left(\frac{x_{tl}-\mu_{jml}}{\sigma_{jml}}\right)^2\right\}} \tag{2.11}$$

onde:

- D é a dimensão do vetor x_t
- $|W_{jm}|$ é o determinante da matriz covariância W_{jm}
- σ_{jml} é o l-ésimo elemento da m-ésima gaussiana do estado S_j.

E o coeficiente de ponderação c_{jm} é dado por

$$c_{jm} > 0, \forall j, m \tag{2.12}$$

$$\sum_{m=1}^{M} c_{jm} = 1, \qquad \forall j \tag{2.13}$$

sendo M é o número de gaussianas na mistura.

2.1.2 Os 3 problemas dos HMM's

No estudo dos modelos ocultos de Markov três problemas se apresentam :

A avaliação - dados um modelo com A, B, e Π definidos e uma seqüência de observações X, determinar qual a probabilidade de que o modelo tenha gerado esta sequência. Este problema pode ser solucionado utilizando-se o algoritmo *Forward.*

A decodificação - dados um modelo com A, B, e Π definidos e uma seqüência de observações X, determinar qual a seqüência de estados mais provável. Este problema pode ser solucionado utilizando-se o algoritmo de *Viterbi* [12].

O aprendizado - dados um modelo e sequências de observações (que se supõe geradas pelo modelo), determinar quais os parâmetros de A, B, e Π que maximizem a probabilidade deste modelo gerar estas sequências. Este problema é geralmente resolvido através do algoritmo *Forward-Backward* também conhecido como *Baum-Welch.* O critério utilizado por este algoritmo para a reestimação dos parâmetros é o de máxima verossimilhança, que consiste em aumentar, a cada época de treinamento, a probabilidade a posteriori, ou seja, a probabilidade do modelo gerar a sequência de observações, dada pela equação (2.14).

$$P(X_{treinamento}|\Lambda) = \prod_{k=1}^{K} P(X^k|\Lambda) \tag{2.14}$$

onde:

- $X_{treinamento}$: conjunto de treinamento $X^1, X^2, ..., X^k$, sendo K o número de locuções de treinamento
- $X^k = (x_1^k, x_2^k, ..., x_{T_k}^k)$: sequência de vetores acústicos que representa a k-ésima locução de treinamento
- $\Lambda = \lambda_1, \lambda_2, .., \lambda_U$: conjunto de modelos HMMs, sendo U o número de modelos.

2.1.3 Reconhecimento de palavras isoladas usando HMMs

A construção de um sistema de reconhecimento de palavras isoladas utilizando HMMs segue em geral os seguintes passos:

- definição do vocabulário da aplicação. Este determina o conjunto de palavras que o sistema pode reconhecer, bem como o número U de modelos a serem treinados;
- definição da topologia dos HMMs para cada palavra (número de estados, salto máximo entre estados adjacentes, função probabilística para representar a emissão dos símbolos, etc.);
- obtenção de um conjunto de locuções de treinamento para cada palavra;
- escolha de um conjunto adequado de parâmetros para representar o sinal de voz (LPC, cepstrais, log-energia, mel-cepstrais, etc.);
- treinamento dos modelos utilizando o algoritmo *Baum-Welch.*

Na etapa de reconhecimento, toma-se uma locução desconhecida e extraem-se os parâmetros que irão constituir a sequência de observações. Esta é então aplicada a todos os modelos do vocabulário gerando, para cada um, uma medida de mérito correspondente à probabilidade daquele modelo gerar a sequência de observações. Assume-se então que a palavra reconhecida é aquela cujo modelo correspondente apresentar a máxima verossimilhança com a sequência de observações:

$$P(X|\lambda_i) = max[P(X|\lambda_j)], \quad 1 \leq j \leq L \tag{2.15}$$

Na Figura 2.2 tem-se um diagrama em blocos ilustrando este procedimento.

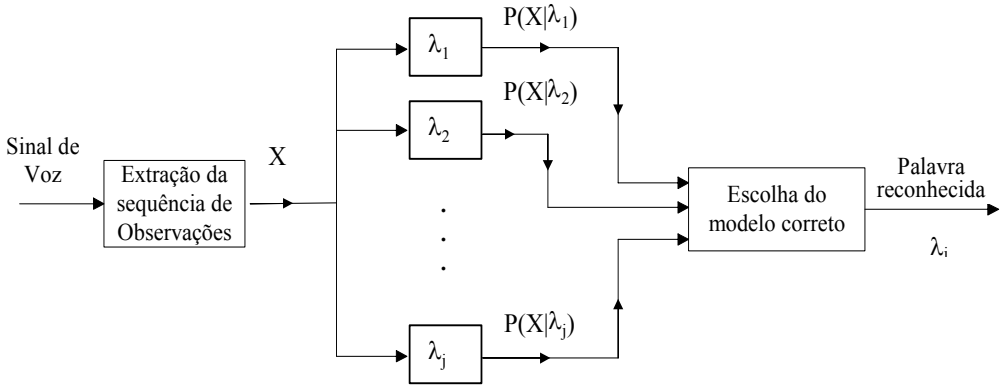

Figura 2.2: *Diagrama em blocos de um sistema de reconhecimento de palavras isoladas (Segundo [10]).*

2.2 Conclusões

O critério ML para treinamento dos HMMs procura encontrar um conjunto de parâmetros que maximize a probabilidade de emitir uma sequência de vetores acústicos X. Desta forma garante-se que, a cada época de treinamento, a probabilidade de cada modelo gerar a sequência de estados correspondente aumenta, mas não é feita nenhuma consideração sobre o que acontece com os modelos concorrentes.

Uma forma de melhorar o desempenho do sistema seria considerar, para o treinamento de cada modelo, não apenas as locuções a ele correspondentes, mas todo o espaço de locuções de treinamento. O algoritmo de treinamento poderia então não só maximizar a probabilidade de o modelo correto gerar a locução em questão, mas também minimizar a probabilidade de os outros modelos gerarem esta mesma locução. Este é o princípio do treinamento discriminativo, que será apresentado no próximo Capítulo.

Capítulo 3

Treinamento discriminativo e o critério MMI

3.1 Critério MMI

Como visto no capítulo anterior, o método de treinamento tradicional para os HMMs baseia-se no critério ML, que procura aumentar a verossimilhança do modelo gerar a locução de treinamento.

Nos últimos anos alguns pesquisadores, talvez inspirados na teoria de redes neurais, vêm buscando novas alternativas para o treinamento dos HMMs utilizando-se de critérios que minimizem diretamente a taxa de erros de reconhecimento.

Um critério que satisfaz esta filosofia é o de maximização da informação mútua - MMI (*Maximum Mutual Information*) [15], que procura ressaltar a diferença entre os modelos que competem entre si, na tentativa de utilizar da melhor forma possível as informações disponíveis no conjunto de treinamento.

Assim, para uma dada sequência de vetores acústicos X, correspondente ao modelo M_i, o objetivo do treinamento baseado no critério MMI é maximizar a probabilidade do modelo correto M_i gerar a locução de treinamento e minimizar a probabilidade dos modelos concorrentes M_j gerarem esta mesma locução, ajudando a diferenciar melhor os modelos que competem entre si durante a etapa de avaliação.

Um dos algoritmos que implementa este critério, e aquele escolhido para este trabalho é o algoritmo *Generalized Probabilistic Descent* (GPD) [16, 17]. Nas seções que se seguem será apresentado um estudo teórico bastante completo e minucioso deste algoritmo, bem como um estudo de como utilizá-lo para o treinamento discriminativo de HMMs contínuos num contexto de reconhecimento de

palavras isoladas e modelos de palavras.

3.2 Algoritmo *Segmental GPD*

Este algoritmo apresenta as seguintes características principais:

- Utiliza o critério MCE (*Minimum Classification Error*) [17, 19], onde o processo de reestimação dos parâmetros dos HMMs baseia-se no critério de minimização da taxa de erro do sistema, isto é, busca-se aumentar a probabilidade do modelo correto gerar a locução de treinamento e diminuir a probabilidade dos modelos concorrentes gerarem a mesma.
- O algoritmo utiliza tanto os erros quanto os acertos de classificação do sistema para ajustar os parâmetros dos HMMs.
- A inicialização pode ser feita a partir de um HMM pré-treinado a partir de outros critérios, tal como ML (*Maximum Likelihood*).

3.2.1 Etapas para implementação

A base do algoritmo *Segmental GPD* [16, 20, 22, 23, 24] é a chamada *função de custo de classificação*, que representa uma medida da distância entre o modelo correto e os demais modelos concorrentes. Esta é obtida segundo um procedimento composto de três etapas [21]:

1. Definir a função discriminante;
2. Definir a função de erro de classificação para cada época de treinamento;
3. Definir a função de custo dependente da função de erro;
4. Definir a nova função custo baseado na função indicadora;

Primeira Etapa: Definição da Função Discriminante

A função discriminante (FD) mede a probabilidade de cada modelo gerar a sequência de observações X. Neste trabalho será utilizada a seguinte forma para a FD:

$$g_k(X|\lambda_k) = \max_q g_k(X, q|\lambda_k) \tag{3.1}$$

onde X é a locução de entrada, q é uma sequência de estados genérica, λ_k é o HMM correspondente à k-ésima palavra do vocabulário, $g_k(X, q|\lambda_k) = ln\{(p(X, q \mid$

$\lambda_k))\}$ e $p(X, q \mid \lambda_k)$ é a probabilidade de emissão da locução X e a ocorrência da sequência de estados q, dado o modelo k.

Estas definições são adequadas aos HMMs, uma vez que o processo de decodificação acústica baseia-se nos valores de verossimilhança obtidos a partir de sequências de estados q. Desta forma, pode-se definir a probabilidade de emissão da locução X dado o modelo k, como:

$$p_k(X|\lambda_k) = \max_q p_k(X, q|\lambda_k) \tag{3.2}$$

Assim, a probabilidade para o modelo da palavra k e para uma locução X composta por T quadros é definida por:

$$p_k(X, q|\lambda_k) = \pi^k_{q_0} \prod_{t=1}^{T} a^k_{q_{t-1}q_t} b^k_{q_t}(x_t) \tag{3.3}$$

onde:

- $\pi^k_{q_0}$ é a probabilidade de o processo iniciar-se no estado q_0,
- $a^k_{q_{t-1}q_t}$ é a probabilidade de ocorrer uma transição entre os estados q_{t-1} e q_t do modelo λ_k,
- $b^k_{q_t}(x_t)$ é a probabilidade de emitir o símbolo x_t no estado q_t do k-ésimo modelo.

Aplicando a equação (3.2) e tomando o logaritmo, tem-se a seguinte definição para a função discriminante:

$$g_k(X|\lambda_k) = \ln\{\max_q p_k(X, q|\lambda_k)\} = \ln\{p_k(X, \bar{q}|\lambda_k)\} \tag{3.4}$$

onde $\bar{q}$ é a sequência de estados ótima ou caminho de máxima verossimilhança, obtido através do algoritmo de Viterbi.

Usando a equação (3.3) podemos reescrever (3.4) como:

$$g_k(X|\lambda_k) = \ln\{\pi^k_{\bar{q}_0}\} + \sum_{t=1}^{T} \ln\{a^k_{\bar{q}_{t-1}\bar{q}_t}\} + \sum_{t=1}^{T} \ln\{b^k_{\bar{q}_t}(x_t)\} \tag{3.5}$$

Em outras palavras, a função discriminante nada mais é que o log-probabilidade de cada HMM gerar a sequência de observações X, passando pelo caminho ótimo, dado pelo algoritmo de Viterbi.

Após definir a função discriminante, vamos em seguida definir a função de erro de classificação, que é uma medida da distância entre a probabilidade do modelo correto gerar a sequência de observações, e a média das probabilidades dos modelos concorrentes gerarem esta mesma sequência.

Segunda Etapa: Definição da função de erro de classificação de cada época de treinamento

Esta função mede a diferença entre a média das probabilidades dos modelos concorrentes emitirem a mesma locução e a probabilidade do modelo correto emitir a locução X . Considerando-se a função discriminante $g_j(X|\lambda_j)$ como sendo o logaritmo da verossimilhança para a entrada X e para o modelo λ_j da j-ésima palavra do vocabulário, define-se a função de erro de classificação para palavra i como [16, 17, 18, 19, 20]:

$$d_i(X|\Lambda) = \underbrace{-g_i(X|\lambda_i)}_{1} + \underbrace{\ln\left[\frac{1}{W-1}\sum_{j\neq i} e^{g_j(X|\lambda_j)\eta}\right]^{\frac{1}{\eta}}}_{2} \tag{3.6}$$

onde:

- 1 é a probabilidade do modelo correspondente à locução de treinamento X gerar a mesma;
- 2 é a média das probabilidades dos modelos concorrentes gerarem a locução de treinamento X;
- η é uma constante de normalização;
- U é o número de palavras do vocabulário.

Observação

Vale salientar que se X é uma locução da palavra i, o modelo correto gera a função discriminante $g_i(X|\lambda_i)$, e os demais modelos do vocabulário geram as funções discriminantes concorrentes $g_j(X|\lambda_j)$. De fato, a contribuição dos modelos concorrentes é introduzida na medida $d_i(X|\Lambda)$ em (3.6) com sinal invertido em relação à função discriminante $g_i(X|\lambda_i)$ do modelo correto.

Para uma locução X pertencente à classe i, valem as seguintes relações:

- $d_i(X|\Lambda) > 0$ houve erro de classificação, isto é, a média das probabilidades dos modelos concorrentes foi maior que a probabilidade do modelo correto;
- $d_i(X|\Lambda) < 0$ não houve erro de classificação, isto é, a probabilidade do modelo correto foi maior que a média das probabilidade dos modelos concorrentes;

Terceira Etapa: Definição da função custo dependente da função de erro

Verifica-se na prática que a função erro $d_i(X|\Lambda))$ assume valores em uma faixa bastante extensa. Estes valores podem gerar problemas numéricos nos passos subsequentes do algoritmo e, desta forma, é necessária uma forma de representar a função erro dentro de uma escala numericamente gerenciável. Assim, define-se a função custo em função da função erro como

$$l_i(X|\Lambda) = l_i(d_i(X|\Lambda)) = \frac{1}{1 + e^{-\gamma d_i(X|\Lambda)}} \tag{3.7}$$

Na Figura 3.1 tem-se um gráfico da função custo contra a função erro. Nesta pode-se observar que a função custo consegue mapear a função erro (que tem uma faixa dinâmica bastante larga) para o intervalo $(0, 1)$.

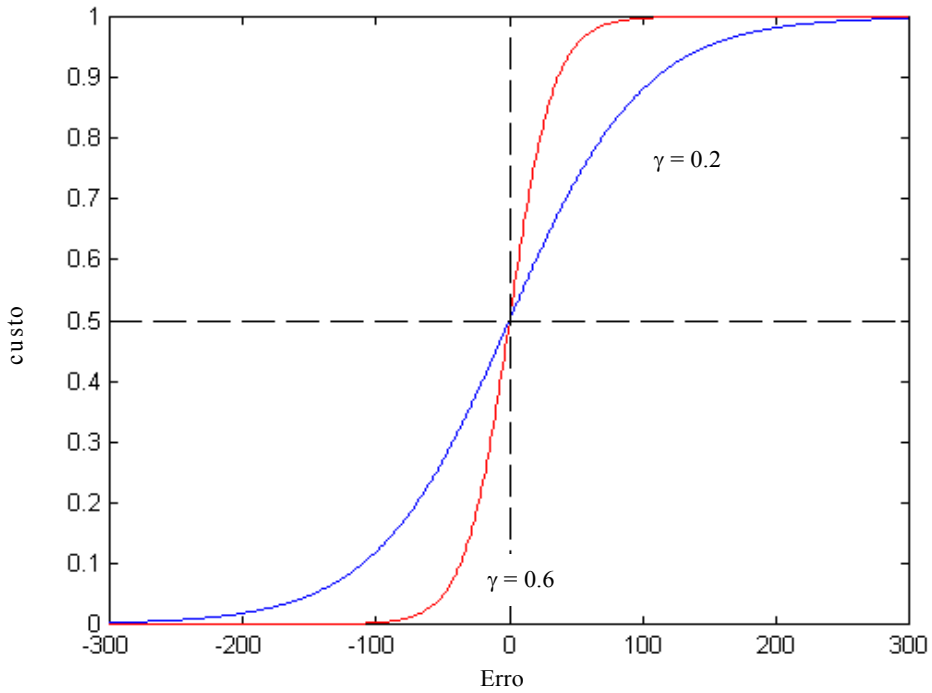

Figura 3.1: *Relação entre a função custo e a função erro*

O parâmetro γ é o limiar de *overflow*, utilizado com o mesmo objetivo do parâmetro η na equação (3.6), ou seja, evitar problemas de ordem numérica no cálculo da função custo.

Quarta Etapa: Definição da nova função custo, baseada na função indicadora

Esta etapa pode ser desconsiderada para sistemas de vocabulários pequenos, como é o caso deste trabalho. Está sendo colocada aqui apenas a título de informação.

Em sistemas de vocabulário extenso e modelamento por subunidades, nem todas as palavras utilizadas para o treinamento irão constar do vocabulário final de reconhecimento. Isto ocorre porque, nestes casos, geralmente faz-se o treinamento para modelos de subunidades fonéticas, e não para modelos de palavras. Como já mencionado anteriormente, este problema não acontece para sistemas de vocabulário pequeno e modelos de palavras, onde locução de treinamento deve corresponder a uma palavra do vocabulário de reconhecimento.

Entretanto, como o objetivo do treinamento discriminativo é minimizar diretamente a taxa de erro do sistema, devem ser utilizadas para o treinamento, apenas as locuções correspondentes a alguma palavra do vocabulário. A função indicadora, mostrada abaixo, cumpre o papel de verificar se determinada locução corresponde ou não a uma palavra do vocabulário:

$$I(X;W_k) = \begin{cases} 1, & X \in W_k \\ 0, & c.c. \end{cases} \tag{3.8}$$

onde W_k é a porção do conjunto de treinamento correspondente à palavra k.

Desta forma, pode-se generalizar a função custo para todo o conjunto de treinamento, utilizando a função $l_i(X|\Lambda)$ e a função indicadora $I(X;W_k)$, através da seguinte expressão:

$$l_i(X|\Lambda) = \sum_{k=1}^{W} l_k(X|\Lambda) I(X;W_k) \tag{3.9}$$

Neste algoritmo, o problema da estimação dos parâmetros dos modelos HMM é mapeado em um problema de otimização baseado na minimização do custo, isto é, minimização da probabilidade de erro.

Esta minimização pode ser resolvida utilizando-se métodos tradicionais de otimização, tal com o Método do Gradiente Descendente, o qual ajusta os parâmetros dos modelos Λ recursivamente, resultando no seguinte processo interativo:

$$\Lambda_{n+1} = \Lambda_n - \varepsilon \nabla l(X_n|\Lambda_n) \tag{3.10}$$

onde Λ_n é o conjunto de HMMs na iteração n, ε é o passo de aprendizagem e $\nabla l(X_n|\Lambda_n)$ é o gradiente da função custo.

3.2.2 Reestimação dos parâmetros dos HMMs segundo o algoritmo Segmental GPD

Nesta seção serão derivadas as equações de reestimação dos parâmetros de um HMM contínuo segundo o algoritmo Segmental GPD. Os parâmetros a serem estimados são as probabilidades de transição entre estados, o coeficiente de ponderação para cada uma das gaussianas da mistura e, finalmente, as médias e variâncias de cada gaussiana.

Por ser um método de otimização baseado no algoritmo gradiente descendente, o Segmental GPD não garante que, ao final de cada época, as restrições probabilísticas impostas aos HMMs se mantenham, de forma que é necessário realizar, ao final de cada época, uma normalização, descrita mais adiante. Assim, nas deduções a seguir, os parâmetros reestimados a partir do algoritmo Segmental GPD *antes* da normalização serão chamados de *parâmetros irrestritos* (com barra), e os parâmetros normalizados, de *parâmetros restritos* (sem barra).

Probabilidade de transição (a_{kj})

No desenvolvimento a seguir, para evitar ambigüidades de notação, considera-se o processo de estimação referente ao modelo i e estados k e j.

Para efetuar o ajuste do parâmetro a_{kj} aplicando o algoritmo *Segmental GPD* obtém-se o parâmetro irrestrito $\overline{a}_{kj}$ referente ao modelo i como mostrado pela equação (3.10), resultando:

$$\overline{a}^i_{kj}(n+1) = a^i_{kj}(n) - \varepsilon \frac{\partial l_i(X|\Lambda)}{\partial a^i_{kj}} \tag{3.11}$$

A partir das equações (3.5),(3.6), (3.7), e a regra da cadeia para derivadas, o gradiente pode-se ser obtido como se segue:

$$\frac{\partial l_i(X|\Lambda)}{\partial a^i_{kj}} = \frac{\partial l_i(X|\Lambda)}{\partial d_i(X|\Lambda)} \frac{\partial d_i(X|\Lambda)}{\partial g_i(X|\lambda_i)} \frac{\partial g_i(X|\lambda_i)}{\partial a^i_{kj}} \tag{3.12}$$

Resolvendo cada um dos termos de forma independente, tem-se:

$$\begin{aligned}
\frac{\partial l_i(X|\Lambda)}{\partial d_i(X|\Lambda)} &= -\frac{1}{(1+e^{-\gamma d_i(X|\Lambda)})^2}\frac{\partial(1+e^{-\gamma d_i(X|\Lambda)})}{\partial d_i(X|\Lambda)} \\
&= -\frac{1}{(1+e^{-\gamma d_i(X|\Lambda)})^2}(-\gamma)e^{-\gamma d_i(X|\Lambda)} \\
&= \gamma l_i(X|\Lambda)\frac{e^{-\gamma d_i(X|\Lambda)}}{(1+e^{-\gamma d_i(X|\Lambda)})} \\
&= \gamma l_i(X|\Lambda)\left[\frac{e^{-\gamma d_i(X|\Lambda)}}{(1+e^{-\gamma d_i(X|\Lambda)})} - 1 + 1\right] \\
&= \gamma l_i(X|\Lambda)\left[\frac{e^{-\gamma d_i(X|\Lambda)} - 1 - e^{-\gamma d_i(X|\Lambda)}}{(1+e^{-\gamma d_i(X|\Lambda)})} + 1\right] \\
&= \gamma l_i(X|\Lambda)\left[1 - \frac{1}{(1+e^{-\gamma d_i(X|\Lambda)})}\right]
\end{aligned} \tag{3.13}$$

Pode-se reescrever a equação (3.13) em função da função custo, como mostrado abaixo:

$$\frac{\partial l_i(X|\Lambda)}{\partial d_i(X|\Lambda)} = \gamma l_i(X|\Lambda)\{1 - l_i(X|\Lambda)\} \tag{3.14}$$

O termo $\frac{\partial d_i(X|\Lambda)}{\partial g_i(X|\Lambda)}$ é dado por:

$$\frac{\partial d_p(X|\Lambda)}{\partial g_p(X|\Lambda)} = -1, \qquad p = i \tag{3.15a}$$

isto é, quando o modelo que está sendo treinado representa a locução de treinamento. Para os modelos concorrentes esta derivada é:

$$\frac{\partial d_p(X|\Lambda)}{\partial g_p(X|\Lambda)} = \frac{e^{g_p(X|\lambda_p)\eta}}{\sum_{j\neq i} e^{g_j(X|\lambda_j)\eta}} \qquad p \neq i \tag{3.15b}$$

A derivada $\frac{\partial g_i(X|\Lambda)}{\partial a^i_{kj}}$ é dada por:

$$\frac{\partial g_i(X|\lambda_i)}{\partial a^i_{kj}} = \sum_{t=2}^{T} \delta(q_{t-1}, k)\delta(q_t, j)\frac{1}{a^i_{kj}} \tag{3.16}$$

onde $\delta(x_t, y)$ é a função que verifica se o segmento x_t da locução de treinamento X está no estado y:

$$\delta(x_t, y) = \begin{cases} 1, & x_t = y \\ 0, & c.c. \end{cases} \tag{3.17}$$

Deste modo, pode-se reescrever a equação (3.11), a partir das equações (3.14), (3.15a), (3.15b) e (3.16) como segue abaixo:

1. Para o modelo correspondente à locução de treinamento:

$$\overline{a}^i_{kj}(n+1) = a^i_{kj}(n) + \varepsilon\gamma l_i(X|\Lambda)\{1 - l_i(X|\Lambda)\}\sum_{t=2}^{T}\delta(q_{t-1}, k)\delta(q_t, j)\frac{1}{a^i_{kj}} \tag{3.18}$$

2. Para os modelos concorrentes

$$\overline{a}^i_{kj}(n+1) = a^i_{kj}(n) - \varepsilon\gamma l_i(X|\Lambda)\{1 - l_i(X|\Lambda)\}\frac{e^{g_p(X|\lambda_p)\eta}}{\sum_{j\neq i} e^{g_j(X|\lambda_j)\eta}} \sum_{t=2}^{T}\delta(q_{t-1}, k)\delta(q_t, j)\frac{1}{a^i_{kj}} \tag{3.19}$$

Probabilidade de emissão de símbolo $b^i_j(x_t)$

No caso dos HMMs contínuos $b^i_j(x_t)$, a função densidade de probabilidade de emitir o símbolo x_t no estado j do modelo i no instante t, é dada por uma mistura de gaussianas da seguinte forma:

$$b^i_j(x_t) = \sum_{m=1}^{M} c_{jm} N(x_t, \mu_{jm}, U_{jm}) \quad 1 \leq j \leq N \tag{3.20}$$

onde:

- x_t é o vetor de entrada,
- M é o número de gaussianas,
- c_{jm} é o coeficiente da m-ésima gaussiana no estado S_j,
- $N(\cdot)$ é uma função densidade de probabilidade Gaussiana multidimensional com vetor média μ_{jm} e matriz de covariância U_{jm},
- N é o número de estados do modelo i.

A função densidade de probabilidade Gaussiana multidimensional diagonal é dada por

$$N(x_t, \mu_{jm}, W_{jm}) = \frac{1}{(2\pi)^{\frac{D}{2}}|W_{jm}|^{\frac{1}{2}}} \; e^{\left\{-\frac{1}{2}\sum_{l=1}^{D}\left(\frac{x_{tl}-\mu_{jml}}{\sigma_{jml}}\right)^2\right\}} \tag{3.21}$$

onde:

- D é a dimensão do vetor x_t
- $|W_{jm}|$ é o determinante da matriz covariância W_{jm}
- σ_{jml} é o l-ésimo elemento da m-ésima gaussiana do estado S_j.

Após a definição da função de emissão de símbolo $\{b_j^i(x)\}$, pode-se determinar as equações de ajuste de cada um dos seus parâmetros.

Coeficiente de ponderação c_{jm}^i

A equação de ajuste de c_{jm}^i é dada por:

$$\overline{c}_{jm}^i(n+1) = c_{jm}^i(n) - \varepsilon\frac{\partial l_i(X;\Lambda)}{\partial c_{jm}^i} \tag{3.22}$$

Assim, pode-se escrever

$$\frac{\partial l_i(X|\Lambda)}{\partial c_{jm}^i} = \frac{\partial l_i(X|\Lambda)}{\partial d_i(X;\Lambda)}\frac{\partial d_i(X|\Lambda)}{\partial g_i(X|\lambda_i)}\frac{\partial g_i(X|\lambda_i)}{\partial b_j^i(x_t)}\frac{\partial b_j^i(x_t)}{\partial c_{jm}^i(n)} \tag{3.23}$$

As derivadas $\frac{\partial l_i(X|\Lambda)}{\partial d_i(X|\Lambda)}$ e $\frac{\partial d_i(X|\Lambda)}{\partial g_i(X|\lambda_i)}$ já foram calculadas anteriormente, (ver equações (3.14), (3.15a) e (3.15b)). Para as demais derivadas tem-se:

$$\frac{\partial g_i(X|\Lambda)}{\partial b_j^i(x_t)} = \sum_{t=1}^{T}\delta(q_t, j)\frac{1}{b_j^i(x_t)} \tag{3.24}$$

$$\frac{\partial b_j^i(x_t)}{\partial c_{jm}^i} = N(x_t; \mu_{jm}^i, W_{jm}^i) \tag{3.25}$$

Finalmente, pode-se reescrever a equação de reestimação dos coeficientes das gaussianas (3.22), a partir das equações (3.14), (3.15a),(3.15b), (3.16), (3.24) e (3.25) como:

1. Para o modelo que está sendo treinado

$$\overline{c}_{jm}^i(n+1) = c_{jm}^i(n) + \varepsilon\gamma l_i(X;\Lambda)\{1 - l_i(X;\Lambda)\}\sum_{t=1}^{T}\delta(q_t, j)\frac{1}{b_j^i(x_t)}N(x_t; \mu_{jm}^i, W_{jm}^i) \tag{3.26a}$$

2. Para os modelos concorrentes

$$\overline{c}^i_{jm}(n+1) = c^i_{jm}(n) - \varepsilon\gamma l_i(X;\Lambda)\{1 - l_i(X;\Lambda)\}\frac{e^{g_p(X|\lambda_p)\eta}}{\sum_{j\neq i} e^{g_j(X|\lambda_j)\eta}} \sum_{t=1}^{T} \delta(q_t, j)\frac{1}{b^i_j(x_t)} N(x_t;\mu^i_{jm}, W^i_{jm}) \tag{3.26b}$$

Média das gaussianas μ^i_{jm}

Para a reestimação das médias das gaussianas μ^i_{jm}, a equação do algoritmo Segmental GPD (3.10) assume a seguinte forma:

$$\overline{\mu}^i_{jml}(n+1) = \mu^i_{jml}(n) - \varepsilon\frac{\partial l_i(X;\Lambda)}{\partial \mu^i_{jml}} \tag{3.27}$$

A equação de ajuste $\frac{\partial l_i(X;\Lambda)}{\partial \mu^i_{jml}}$ é dada por:

$$\frac{\partial l_i(X|\Lambda)}{\partial \mu^{(i)}_{jml}} = \frac{\partial l_i(X|\Lambda)}{\partial d_i(X|\Lambda)}\frac{\partial d_i(X|\Lambda)}{\partial g_i(X|\lambda_i)}\frac{\partial g_i(X|\lambda_i)}{\partial b^i_j(x_t)}\frac{\partial b^i_j(x_t)}{\partial N(x_t;\mu^i_{jm}, W^i_{jm})}\frac{\partial N(x_t;\mu^i_{jm}, W^i_{jm})}{\mu^i_{jml}} \tag{3.28}$$

As derivadas $\frac{\partial l_i(X|\Lambda)}{\partial d_i(X|\Lambda)}$, $\frac{\partial d_i(X|\Lambda)}{\partial g_i(X|\lambda_i)}$ e $\frac{\partial g_i(X|\lambda_i)}{\partial b^i_j(x_t)}$ já foram calculadas anteriormente, equações (3.14), (3.15a), (3.15b) e (3.24). Para as demais, temos

$$\frac{\partial b^i_j(x_t)}{\partial N(x_t;\mu^i_{jm}, W^i_{jm})} = c^i_{jm} \tag{3.29}$$

$$\begin{aligned}\frac{\partial N(x_t;\mu^i_{jm}, W^i_{jm})}{\mu^i_{jml}} &= N(x_t;\mu^i_{jm}, W^i_{jm})(-\frac{1}{2})\frac{2}{{\sigma^i_{jml}}^2}(x_{tl} - \mu^i_{jml})(-1) \\ &= N(x_t;\mu^i_{jm}, W^i_{jm})\frac{(x_{tl} - \mu^i_{jml})}{{\sigma^i_{jml}}^2}\end{aligned} \tag{3.30}$$

Deste modo, pode-se reescrever a esquação (3.27), a partir das equações (3.14), (3.15a), (3.15b), (3.24), (3.29) e (3.30) como segue abaixo:

1. Para o modelo que está sendo treinado:

$$\overline{\mu}^i_{jml}(n+1) = \mu^i_{jml}(n) + \varepsilon\gamma l_i(X;\Lambda)\{1 - l_i(X;\Lambda)\}\sum_{t=1}^{T}\delta(q_t,j)\frac{1}{b^i_j(x_t)} N(x_t;\mu^i_{jm},W^i_{jm})c^i_{jm}\frac{(x_{tl}-\mu_{jml})}{\sigma^2_{jml}} \tag{3.31a}$$

2. Para os modelos concorrentes

$$\overline{\mu}^i_{jml}(n+1) = \mu^i_{jml}(n) - \varepsilon\gamma l_i(X;\Lambda)\{1 - l_i(X;\Lambda)\}\frac{e^{g_p(X|\lambda_p)\eta}}{\sum_{j\neq i} e^{g_j(X|\lambda_j)\eta}} \sum_{t=1}^{T}\delta(q_t,j)\frac{1}{b^i_j(x_t)} N(x_t;\mu^i_{jm},W^i_{jm})c^i_{jm}\frac{(x_{tl}-\mu_{jml})}{\sigma^2_{jml}} \tag{3.31b}$$

Desvio padrão das gaussianas σ^i_{jm}

Neste trabalho utilizou-se uma matriz de covariância diagonal para a fdp gaussiana multidimensional, uma simplificação bastante usada na literatura [12]. Assim, a equação de ajuste para o l-ésimo elemento da diagonal da matriz de covariância é obtida a partir da equação (3.10), resultando:

$$\overline{\sigma}^i_{jml}(n+1) = \sigma^i_{jml}(n) - \varepsilon\frac{\partial l_i(X;\Lambda)}{\partial\sigma^i_{jml}} \tag{3.32}$$

A equação de ajuste $\frac{\partial l_i(X;\Lambda)}{\partial\sigma^i_{jml}}$ é dada por:

$$\frac{\partial l_i(X;\Lambda)}{\partial\sigma^i_{jml}} = \frac{\partial l_i(X;\Lambda)}{\partial d_i(X;\Lambda)}\frac{\partial d_i(X;\Lambda)}{\partial g_i(X;\Lambda)}\frac{\partial g_i(X;\Lambda)}{\partial b^i_j(x_t)}\frac{\partial b^i_j(x_t)}{\partial N(x_t;\mu^i_{jm},W^i_{jm})} \frac{\partial N(x_t;\mu^i_{jm},W^i_{jm})}{\sigma^i_{jml}} \tag{3.33}$$

Todas as diferenciais, exceto a última já foram derivadas nas seções anteriores. Para a última, $\frac{\partial N(x_t;\mu^i_{jm},W^i_{jm})}{\sigma^i_{jml}}$, usa-se a equação (3.21) para obter:

$$\frac{\partial N(x_t;\mu^i_{jm},W^i_{jm})}{\partial \sigma^i_{jml}} = \left\{ \frac{\partial}{\partial \sigma^i_{jml}} \left(\frac{1}{2\pi^{\frac{D}{2}}} \left[\prod_{l=1}^{D} \sigma^2_{jml} \right]^{-\frac{1}{2}} \right) \right\} e^{-\frac{1}{2}\sum_{l=1}^{D}\left(\frac{x_{tl}-\mu^i_{jml}}{\sigma^i_{jml}}\right)^2} + \frac{1}{2\pi^{\frac{D}{2}}} \left[\prod_{l=1}^{D} {\sigma^i_{jml}}^2 \right]^{-\frac{1}{2}} \left\{ \frac{\partial}{\partial \sigma^i_{jml}} \left(e^{-\frac{1}{2}\sum_{l=1}^{D}\left(\frac{x_{tl}-\mu^i_{jml}}{\sigma^i_{jml}}\right)^2} \right) \right\}$$

$$\frac{\partial N(x_t;\mu^i_{jm},W^i_{jm})}{\partial \sigma^i_{jml}} = \frac{1}{2\pi^{\frac{D}{2}}} \left\{ \frac{\partial}{\partial \sigma^i_{jml}} \left(\left[\prod_{l=1}^{D} {\sigma^i_{jml}}^2 \right]^{-\frac{1}{2}} \right) \right\} e^{-\frac{1}{2}\sum_{l=1}^{D}\left(\frac{x_{tl}-\mu^i_{jml}}{\sigma^i_{jml}}\right)^2} + \left[\prod_{l=1}^{D} {\sigma^i_{jml}}^2 \right]^{-\frac{1}{2}} \left\{ \frac{\partial}{\partial \sigma^i_{jml}} \left(e^{-\frac{1}{2}\sum_{l=1}^{D}\left(\frac{x_{tl}-\mu^i_{jml}}{\sigma^i_{jml}}\right)^2} \right) \right\}$$

$$\frac{\partial N(x_t;\mu^i_{jm},W^i_{jm})}{\partial \sigma^i_{jml}} = \frac{1}{2\pi^{\frac{D}{2}}} \left\{ -\frac{1}{2} \left[\prod_{l=1}^{D} {\sigma^i_{jml}}^2 \right]^{-\frac{3}{2}} 2\sigma^i_{jml} \left[\prod_{l`=1 \; l`\neq l}^{D} {\sigma^i_{jml}}^2 \right] e^{-\frac{1}{2}\sum_{l=1}^{D}\left(\frac{x_{tl}-\mu^i_{jml}}{\sigma^i_{jml}}\right)^2} + \left[\prod_{l=1}^{D} {\sigma^i_{jml}}^2 \right]^{-\frac{1}{2}} e^{-\frac{1}{2}\sum_{l=1}^{D}\left(\frac{x_{tl}-\mu^i_{jml}}{\sigma^i_{jml}}\right)^2} (-\frac{1}{2}) \frac{(x_{tl}-\mu^i_{jml})^2}{{\sigma^i_{jml}}^3} (-2) \right\}$$

$$\frac{\partial N(x_t;\mu^i_{jm},W^i_{jm})}{\partial \sigma^i_{jml}} = \frac{1}{2\pi^{\frac{D}{2}}} \left[\prod_{l=1}^{D} {\sigma^i_{jml}}^2 \right]^{-\frac{1}{2}} e^{-\frac{1}{2}\sum_{l=1}^{D}\left(\frac{x_{tl}-\mu^i_{jml}}{\sigma^i_{jml}}\right)^2} \left\{ - \left[\prod_{l=1}^{D} {\sigma^i_{jml}}^2 \right]^{-1} \sigma^i_{jml} \left[\prod_{l`=1 \; l`\neq l}^{D} {\sigma^i_{jml}}^2 \right] + \left(\frac{x_{tl}-\mu^i_{jml}}{\sigma^i_{jml}} \right)^2 \frac{1}{\sigma^i_{jml}} \right\}$$

$$\frac{\partial N(x_t;\mu^i_{jm},W^i_{jm})}{\partial \sigma^i_{jml}} = \frac{1}{2\pi^{\frac{D}{2}}}\left[\prod_{l=1}^{D}{\sigma^i_{jml}}^2\right]^{-\frac{1}{2}} e^{-\frac{1}{2}\sum_{l=1}^{D}\left(\frac{x_{tl}-\mu^i_{jml}}{\sigma^i_{jml}}\right)^2}\left\{-\left[\prod_{l=1}^{D}{\sigma^i_{jml}}^2\right]^{-1}\right.$$
$$\left.\sigma^i_{jml}\left[\prod_{l'=1\ l'\neq l}^{D}{\sigma^i_{jml}}^2\right]\frac{\sigma^i_{jml}}{\sigma^i_{jml}}+\left(\frac{x_{tl}-\mu^i_{jml}}{\sigma^i_{jml}}\right)^2\frac{1}{\sigma^i_{jml}}\right\}$$

$$\frac{\partial N(x_t;\mu^i_{jm},W^i_{jm})}{\partial \sigma^i_{jml}} = \frac{1}{2\pi^{\frac{D}{2}}}\left[\prod_{l=1}^{D}{\sigma^i_{jml}}^2\right]^{-\frac{1}{2}} e^{-\frac{1}{2}\sum_{l=1}^{D}\left(\frac{x_{tl}-\mu^i_{jml}}{\sigma^i_{jml}}\right)^2}\frac{1}{\sigma^i_{jml}}$$
$$\left\{-1+\left(\frac{x_{tl}-\mu^i_{jml}}{\sigma^i_{jml}}\right)^2\right\}$$

Chegando finalmente a

$$\frac{\partial N(x_t;\mu^i_{jm},W^i_{jm})}{\partial \sigma^i_{jml}} = N(x_t;\mu^i_{jm},W^i_{jm})\frac{1}{\sigma^i_{jml}}\left\{-1+\left(\frac{x_{tl}-\mu^i_{jml}}{\sigma^i_{jml}}\right)^2\right\} \tag{3.34}$$

Deste modo, pode-se reescrever a equação (3.33), a partir das equações (3.14), (3.15a), (3.15b), (3.24), (3.29), (3.34) como segue abaixo:

1. Para o modelo que está sendo treinado

$$\overline{\sigma}^i_{jml}(n+1) = \sigma^i_{jml}(n) + \varepsilon\gamma l_i(X;\Lambda)\{1-l_i(X;\Lambda)\}\sum_{t=1}^{T}\delta(q_t,j)\frac{1}{b^i_j(x_t)}$$
$$N(x_t;\mu^i_{jm},W^i_{jm})c^i_{jm}\frac{1}{\sigma_{jml}}\left\{-1+\left(\frac{x_{tl}-\mu_{jml}}{\sigma_{jml}}\right)^2\right\} \tag{3.35a}$$

2. Para os modelos concorrentes

$$\overline{\sigma}^i_{jml}(n+1) = \sigma^i_{jml}(n) - \varepsilon\gamma l_i(X;\Lambda)\{1 - l_i(X;\Lambda)\}\frac{e^{g_p(X|\lambda_p)\eta}}{\sum_{j\neq i} e^{g_j(X|\lambda_j)\eta}}\sum_{t=1}^{T}\delta(q_t,j)$$
$$\frac{1}{b^i_j(x_t)}N(x_t;\mu^i_{jm},W^i_{jm})c^i_{jm}\frac{1}{\sigma_{jml}}\left\{-1+\left(\frac{x_{tl}-\mu_{jml}}{\sigma_{jml}}\right)^2\right\} \tag{3.35b}$$

3.2.3 Normalização dos parâmetros

O algoritmo *Segmental GPD* é um processo de otimização dos parâmetros dos HMM's e, portanto, não obedecendo necessariamente às restrições dos parâmetros do HMM apresentadas no capítulo anterior (coeficiente de ponderação, probabilidade de transição e variância). Por isso é necessário, após cada iteração do processo de reestimação, realizar a normalização destes parâmetros. A seguir são mostrados os processos de normalização utilizados para cada um dos parâmetros reestimados.

Probabilidade de transição (a_{ij})

Como visto no Capítulo 2, a probabilidade de transição entre estados para um HMM deve seguir as seguintes restrições:

$$\sum_{j=1}^{N} a_{kj} = 1, \qquad \forall k$$

$$a_{kj} > 0, \qquad \forall k,j$$

Aplica-se então um processo de normalização linear ao parâmetro irrestrito $\overline{a}^i_{kj}$:

$$a^i_{kj} = \frac{\overline{a}^i_{kj}}{\sum_{k=1}^{Q}\overline{a}^i_{kj}}, \qquad k,j = 1,2,...,Q \tag{3.36}$$

onde Q é o número de estados do modelo HMM da palavra i.

Função densidade de probabilidade de emissão ($b^i_j(x_t)$)

$b^i_j(x_t)$ é uma função densidade de probabilidade constituída por uma mistura de M fdps gaussianas multidimensionais. De modo a garantir que seja realmente uma fdp válida, o parâmetro c_{jm} deve obedecer às seguintes restrições:

$$c_{jm} > 0, \qquad j = 1, \ldots, N \qquad e \qquad m = 1, \ldots, M$$

$$\sum_{m=1}^{M} c_{jm} = 1, \quad j = 1, \ldots, N$$

onde N é o número de estados do modelo e M, o número de gaussianas na mistura.

Aplicando o processo de normalização linear ao parâmetro irrestrito $\overline{c}^i_{jm}$ para satisfazer a condição (3.2.3), tem-se:

$$c^i_{jm} = \frac{\overline{c}^i_{jm}}{\sum_{j=1}^{Q} \overline{c}^i_{jm}}, \quad j = 1, 2, ..., Q \;\; m = 1, 2, ..., M. \tag{3.37}$$

Os parâmetros média μ^i_{jm} e variância σ^2_{jmd} não sofrem nenhuma restrição do ponto de vista estatístico, e desta forma poderiam ser usados sem normalização. Entretanto verifica-se na prática que valores de variância muito baixos podem levar a problemas numéricos quando são avaliados sinais que ocorrem muito longe da média. De forma a evitar este problema, coloca-se a seguinte restrição para a variância:

$$\sigma^2_{jmd} \geq \varepsilon_2, \qquad d = 1, \ldots, D \qquad e \qquad m = 1, \ldots, M \tag{3.38}$$

onde ε_2 é um número positivo pequeno, neste trabalho foi utilizado 0,00001.

3.3 Resumo do Algoritmo de Treinamento Discriminativo para Palavras Isoladas

O algoritmo de Treinamento Discriminativo para o caso de HMMs contínuos e modelos de palavras pode ser sintetizado nas seguintes etapas:

Inicialização

Para a inicialização do treinamento discriminativo geralmente são usados HMMs previamente treinados. Neste trabalho foram utilizados HMMs treinados através do algoritmo Baum-Welch (critério ML).

Reestimação dos parâmeros dos HMMs

Inicialmente executa-se o algoritmo de Viterbi para segmentar cada elocução de treinamento e calcular a função custo. A partir daí cada um dos parâmetros é

reestimado segundo as equações a seguir:

Probabilidades de transição (a_{ij})

1. Para o modelo correspondente à locução de treinamento:

$$\overline{a}^i_{kj}(n+1) = a^i_{kj}(n) + \varepsilon\gamma l_i(X;\Lambda)\{1 - l_i(X;\Lambda)\}\sum_{t=2}^{T}\delta(q_{t-1},k)\delta(q_t,j)\frac{1}{a^i_{kj}}$$

2. Para os modelos concorrentes:

$$\overline{a}^i_{kj}(n+1) = a^i_{kj}(n) - \varepsilon\gamma l_i(X|\Lambda)\{1 - l_i(X|\Lambda)\}\frac{e^{g_p(X|\lambda_p)\eta}}{\sum_{j\neq i} e^{g_j(X|\lambda_j)\eta}}$$
$$\sum_{t=2}^{T}\delta(q_{t-1},k)\delta(q_t,j)\frac{1}{a^i_{kj}}$$

Coeficientes de ponderação c^i_{jm}

1. Para o modelo correspondente à locução de treinamento:

$$\overline{c}^i_{jm}(n+1) = c^i_{jm}(n) + \varepsilon\gamma l_i(X;\Lambda)\{1 - l_i(X;\Lambda)\}\sum_{t=1}^{T}\delta(q_t,j)\frac{1}{b^i_j(x_t)}N(x_t;\mu^i_{jm},W^i_{jm})$$

2. Para os modelos concorrentes:

$$\overline{c}^i_{jm}(n+1) = c^i_{jm}(n) - \varepsilon\gamma l_i(X;\Lambda)\{1 - l_i(X;\Lambda)\}\frac{e^{g_p(X|\lambda_p)\eta}}{\sum_{j\neq i} e^{g_j(X|\lambda_j)\eta}}$$
$$\sum_{t=1}^{T}\delta(q_t,j)\frac{1}{b^i_j(x_t)}N(x_t;\mu^i_{jm},W^i_{jm})$$

Médias das gaussianas μ^i_{jm}

1. Para o modelo correspondente à locução de treinamento:

$$\overline{\mu}^i_{jml}(n+1) = \mu^i_{jml}(n) + \varepsilon\gamma l_i(X;\Lambda)\{1 - l_i(X;\Lambda)\}\sum_{t=1}^{T}\delta(q_t,j)\frac{1}{b^i_j(x_t)}$$
$$N(x_t;\mu^i_{jm},W^i_{jm})c^i_{jm}\frac{(x_{tl} - \mu_{jml})}{\sigma^2_{jml}}$$

2. Para os modelos concorrentes:

$$\overline{\mu}^i_{jml}(n+1) = \mu^i_{jml}(n) - \varepsilon\gamma l_i(X;\Lambda)\{1 - l_i(X;\Lambda)\}\frac{e^{g_p(X|\lambda_p)\eta}}{\sum_{j\neq i} e^{g_j(X|\lambda_j)\eta}}$$
$$\sum_{t=1}^{T}\delta(q_t,j)\frac{1}{b^i_j(x_t)}N(x_t;\mu^i_{jm},W^i_{jm})c^i_{jm}\frac{(x_{tl}-\mu_{jml})}{\sigma^2_{jml}}$$

Desvio padrão das gaussianas σ^i_{jm}

1. Para o modelo correspondente à locução de treinamento:

$$\overline{\sigma}^i_{jml}(n+1) = \sigma^i_{jml}(n) + \varepsilon\gamma l_i(X;\Lambda)\{1 - l_i(X;\Lambda)\}\sum_{t=1}^{T}\delta(q_t,j)\frac{1}{b^i_j(x_t)}$$
$$N(x_t;\mu^i_{jm},W^i_{jm})c^i_{jm}\frac{1}{\sigma_{jml}}\left\{-1+\left(\frac{x_{tl}-\mu_{jml}}{\sigma_{jml}}\right)^2\right\}$$

2. Para os modelos concorrentes

$$\overline{\sigma}^i_{jml}(n+1) = \sigma^i_{jml}(n) - \varepsilon\gamma l_i(X;\Lambda)\{1 - l_i(X;\Lambda)\}\frac{e^{g_p(X|\lambda_p)\eta}}{\sum_{j\neq i} e^{g_j(X|\lambda_j)\eta}}\sum_{t=1}^{T}\delta(q_t,j)$$
$$\frac{1}{b^i_j(x_t)}N(x_t;\mu^i_{jm},W^i_{jm})c^i_{jm}\frac{1}{\sigma_{jml}}\left\{-1+\left(\frac{x_{tl}-\mu_{jml}}{\sigma_{jml}}\right)^2\right\}$$

Normalização dos parâmetros

Finalmente, realizam-se as operações de normalização dos parâmetros para que estes atendam às restrições impostas pela teoria de probabilidade e pelos problemas de precisão numérica.

a) Probabilidade de transição (a_{ij})

$$a^i_{kj} = \frac{\overline{a}^i_{kj}}{\sum_{k=1}^{Q}\overline{a}^i_{kj}}, \qquad k,j = 1,2,...,Q$$

b) Coeficientes de ponderação das gaussianas (c^i_{jm})

$$c^i_{jm} = \frac{\overline{c}^i_{jm}}{\sum_{j=1}^{Q}\overline{c}^i_{jm}}, \qquad j = 1,2,...,Q \;\; m = 1,2,...,M.$$

c) Variância das gaussianas (σ^2_{jmd})

$$\sigma^2_{jmd} \geq \varepsilon_2, \qquad d = 1, \ldots, D \qquad e \qquad m = 1, \ldots, M$$

onde ε_2 é um número positivo pequeno.

Término do treinamento

O critério de parada utilizado para finalizar o TD foi baseado na taxa de acertos obtida no material de teste: o sistema foi treinado enquanto esta taxa estivesse subindo.

A rigor este teste deveria ser realizado em um conjunto de validação, diferente do conjunto de locuções de teste, mas isto exigiria uma base de dados bastante grande, recurso não disponível quando da realização deste trabalho. Entretanto, este fato não afeta a validade dos resultados obtidos.

Capítulo 4

Sistema Implementado

4.1 Introdução

Para testar o ganho obtido ao se utilizar o treinamento discriminativo foi implementado um sistema de reconhecimento de palavras isoladas e vocabulário pequeno, baseado em modelos ocultos de Markov contínuos com modelamento por palavras. Este sistema tomou como base o trabalho do Dr. José Antônio Martins [10], desenvolvido durante sua tese de doutorado.

Neste capítulo será apresentado o sistema desenvolvido, bem como as bases de dados utilizadas nos testes. Também serão tecidos alguns comentários sobre como foram resolvidos alguns problemas numéricos encontrados durante a implementação.

4.2 Bases de Dados

A avaliação do sistema HMM implementado foi realizada com o uso de duas bases de dados: a primeira composta por locuções gravadas no laboratório de Pós-Graduação do Instituto Nacional de Telecomunicações - INATEL, e a segunda gerada pelo Dr. José Antônio Martins e se encontra alojada no Laboratório de Processamento Digital de Fala do Departamento de Comunicações da Universidade Estadual de Campinas - UNICAMP [10], ambas multilocutores e gravadas em ambiente de escritório.

4.2.1 Base de dados do INATEL

Esta base de dados foi gravada no laboratório de Pós-Graduação do Instituto Nacional de Telecomunicações, sendo composta por 10 palavras distintas corres-

pondentes ao nome de alguns aplicativos do Windows®:

- calculadora, excel, explorer, freecell, internet, matlab, notepad, paint, powerpoint e word.

Os locutores, 10 homens e 2 mulheres, são alunos de mestrado do Inatel, apresentando faixa etária entre 23 e 30 anos de idade. Todos nasceram ou residem há mais de 10 anos na região sudeste, mais exatamente no Sul e Zona da Mata de Minas Gerais, São Paulo e Rio de Janeiro.

Cada locutor pronunciou três vezes cada palavra, resultando um total de 36 locuções de cada palavra. Desta forma, esta base de dados é composta por um total de 360 locuções. Para o treinamento foram selecionados 10 locutores (9 homens e 1 mulher), e os dois locutores restantes foram usados para os testes.

A gravação da base foi feita em ambiente de escritório, com pouco ruído, usando um microfone direcional de boa qualidade e placa de som SoundBlaster AWE 64, a uma freqüência de amostragem de 11,025 kHz e resolução de 16 bits. Os dados foram armazenados em formato Windows PCM (wav).

4.2.2 Base de dados do DECOM-UNICAMP

Esta base de dados apresenta um vocabulário de 50 palavras organizadas em 6 grupos, como mostrado abaixo:

Dígitos : zero, um, dois, três, quatro, cinco, seis, sete, oito, nove, meia;

Comandos : sim, não, terminar, repetir, continuar, voltar, avançar, certo, errado, opções, ajuda;

Regiões : norte,nordeste, sul, sudeste, centro-oeste;

Signos : áries, touro, câncer, leão, gêmeos, virgem, libra, escorpião, capricórnio, sagitário, aquário, peixes;

Opções : horóscopo, dólar, real, tempo, esportes;

Organizações : departamento, divisão, seção, coordenação, imagem, voz.

O conjunto de locutores para esta base de dados consiste de 69 pessoas adultas, sendo 43 do sexo masculino e 26 do sexo feminino. Cada locutor pronunciou cada palavra 3 vezes, e deste modo tem-se $50 \times 69 \times 3 = 10350$ locuções nesta base de dados. 22 homens e 13 mulheres constituem o material de treinamento, e os demais locutores fazem parte do grupo de teste.

A aquisição desta base de dados foi realizada em um ambiente com ruído de escritório através de uma placa DSP-16 *Data Aquisition Processor*, fabricada pela

Ariel, acoplada a um microcomputador. As gravações foram realizadas em 16 bits, a uma frequência de amostragem de 8 kHz, com o sinal analógico pré-filtrado na faixa entre 100 Hz e 3400 Hz.

4.3 Sistema Desenvolvido

O sistema desenvolvido é formado por dois módulos:

- Módulo de Treinamento
- Módulo de Reconhecimento

Os dois módulos do sistema foram implementados em linguagem C++ para a plataforma Windows®. Na implementação, teve-se o cuidado de criar um código estruturado e extensamente documentado, de forma que outros pesquisadores possam desenvolver as suas idéias a partir deste sistema.

4.3.1 Modelos de Markov para as palavras do vocabulário

Para este sistema utilizou-se 3 gaussianas por estado, sendo estas de dimensão 12, e o modelo de Bakis (*left-rigth*), com número de estados igual ao número de fones de cada palavra mais 3, e $\triangle = 1$ ou seja, só são permitidas transições para o mesmo estado e para o estado imediatamente posterior, como mostrado na Figura 4.1.

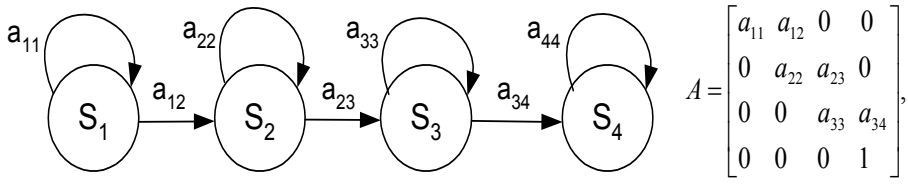

Figura 4.1: *Modelo left-to-right utilizado no sistema.*

Na tabela 4.1 tem-se o número de estados utilizados para cada palavra da base de dados do Departamento de Comunicações da Universidade Estadual de Campinas - UNICAMP, e na tabela 4.2, número de estados para cada palavra da base de dados do INATEL.

4.3.2 Parâmetros

Como vetores acústicos foram usados 12 parâmetros mel-cepstrais, calculados a partir de janelas de 20 ms, atualizadas a cada 10 ms (superposição de 50%).

Tabela 4.1: *Número de estados representando cada palavra da base de dados do DECOM-UNICAMP.*

Palavra	Número de estados	Palavra	Número de estados
zero	7	nordeste	11
um	5	sul	6
dois	7	sudeste	10
três	7	centro-oeste	14
quatro	9	esportes	11
cinco	8	departamento	15
seis	7	divisão	10
sete	7	seção	8
oito	7	coordenação	14
nove	7	imagem	9
meia	7	voz	6
sim	6	áries	8
não	6	touro	8
terminar	11	câncer	9
repetir	10	leão	7
continuar	12	gêmeos	9
voltar	9	virgem	9
avançar	10	libra	11
certo	8	escorpião	12
errado	8	capricórnio	14
opções	9	sagitário	12
dólar	8	aquário	10
real	7	peixes	9
tempo	8	horóscopo	12
norte	8	ajuda	8

Tabela 4.2: *Número de estados representando cada palavra da base de dados do INATEL.*

Palavra	Número de estados	Palavra	Número de estados
calculadora	11	excel	8
explorer	11	freecel	9
internet	11	matlab	9
notpad	9	paint	8
powerpoint	13	word	7

Antes da extração, o sinal é submetido a alguns pré-processamentos: pré-ênfase com um filtro passa altas $H(z) = 1 - 0,95z^{-1}$, e janelamento através de uma janela de Hamming.

4.3.3 Módulo de Treinamento

O treinamento é realizado em duas etapas: inicialmente utiliza-se o algoritmo Baum-Welch para gerar os modelos iniciais. Este bloco foi desenvolvido tomando-se como referência o trabalho do Dr. José Martins [10], e um diagrama em blocos do mesmo é mostrado na Figura 4.2.

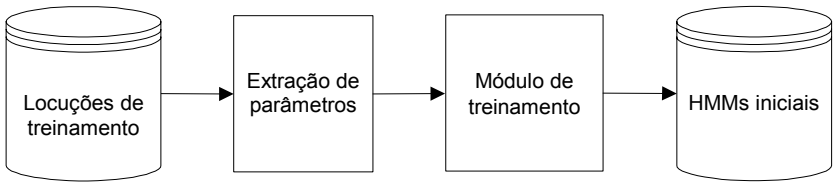

Figura 4.2: *Diagrama em blocos do módulo de treinamento via Baum-Welch.*

Com os modelos treinados a partir do algoritmo Baum-Welch (critério ML), passa-se então à etapa de treinamento discriminativo. Nesta, cada locução atualiza não apenas os parâmetros do modelo correspondente mas também os dos outros modelos (concorrentes). Um diagrama em blocos deste sistema é mostrado na Figura 4.3

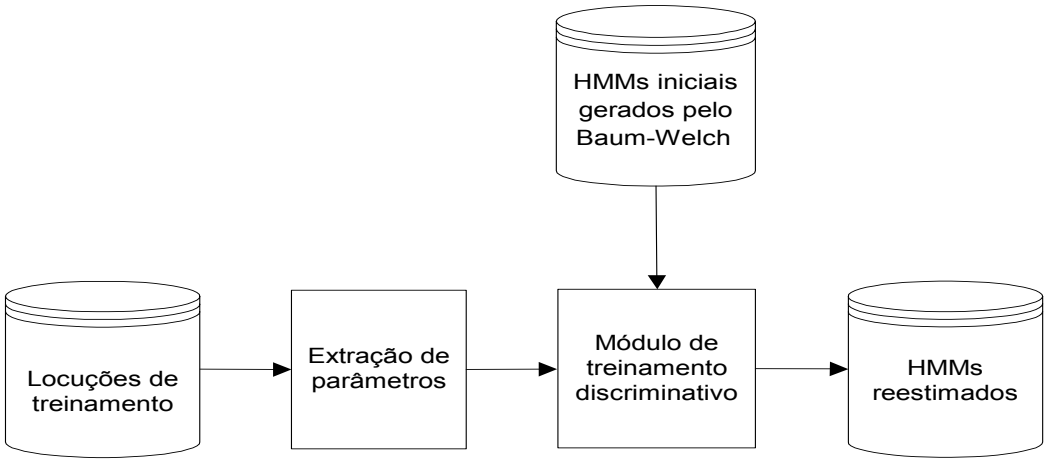

Figura 4.3: *Diagrama em blocos do módulo de treinamento discriminativo.*

4.3.4 Módulo de Reconhecimento

Para o módulo de reconhecimento optou-se pelo algoritmo de Viterbi ao invés do método exato (algoritmo Forward), uma vez que o primeiro é mais eficiente computacionalmente e apresenta virtualmente o mesmo desempenho em termos de taxa de acertos. Um diagrama de blocos para este sistema é mostrado na Figura 4.4.

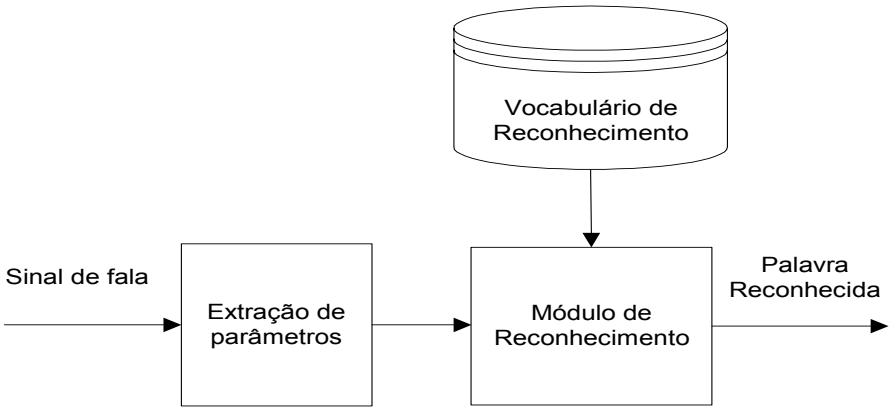

Figura 4.4: *Diagrama em blocos do módulo de reconhecimento*

4.4 Problemas numéricos devido às constantes de normalização

Nos testes realizados observou-se que o algoritmo é bastante instável numericamente. O uso de funções exponenciais em algumas equações leva facilmente as variáveis a assumirem valores extremamente altos ou baixos, excedendo facilmente a sua faixa de representação, mesmo utilizando precisão dupla. Mais especificamente, as equações problemáticas são a que define a função de erro de classificação (3.6) e a equação da função custo (3.7). Felizmente estes problemas são facilmente resolvidos com o uso de constantes de normalização, como mostrado abaixo:

- A equação que define a função de erro de classificação é

$$d_i(X|\Lambda) = -g_i(X|\lambda_i) + \ln\left[\frac{1}{W-1}\sum_{j\neq i} e^{g_j(X|\lambda_j)\eta}\right]^{\frac{1}{\eta}}$$

  O termo $g_j(X|\lambda_j)$ é a log-verossimilhança da locução dado o modelo, e portanto uma grandeza negativa. Na prática, para o problema em questão,

foram observados valores de verossimilhança variando na faixa de -1000 a -3000, o que faz com que a exponencial seja igual a zero para toda a faixa, levando a $\ln(0) = -\infty$. Para evitar este problema, devemos ter $0 < \eta < 1$ afim de diminuir a faixa dinâmica destes valores. Um valor que se mostrou adequado neste caso foi $\eta = 0,001$.

- A equação da função custo é

$$l_i(X|\Lambda) = l_i(d_i(X|\Lambda)) = \frac{1}{1 + e^{-\gamma d_i(X|\Lambda)}}$$

Observou-se que na prática $d_i(X|\Lambda)$ assume valores no intervalo (-750,+750) aproximadamente. Para $d_i(X|\Lambda) = -750$ tem-se $e^{-(-750)} = \infty$, e para $d_i(X|\Lambda) = 750$ tem-se $e^{-750} = 0$. A exemplo do caso anterior, se fizermos $0 < \gamma < 1$ este problema está resolvido. Após vários testes optou-se por $\gamma = 0,001$.

No capítulo a seguir serão descritos os testes realizados para a avaliação do desempenho do sistema treinado através do método convencional (Baum-Welch) e do método discriminativo (Segmental GPD).

Capítulo 5

Testes e análise dos resultados

5.1 Testes iniciais

Inicialmente fez-se um teste inicial com os modelos treinados apenas com o algoritmo Baum-Welch. O objetivo desta etapa é estabelecer um padrão de comparação, a partir do qual serão julgados os resultados obtidos no treinamento discriminativo.

Os modelos foram treinados até que a diferença entre a verossimilhança média entre a época atual e a época anterior ficasse abaixo de um determinado limiar. O critério usado foi o de que a medida de distorção, dada por

$$d = \frac{\overline{P}(O|\Lambda)_{\text{atual}} - \overline{P}(O|\Lambda)_{\text{anterior}}}{\overline{P}(O|\Lambda)_{\text{atual}}} \tag{5.1}$$

ficasse abaixo de um limiar ϵ. Para este trabalho foi escolhido $\epsilon = 10^{-5}$.

Foram treinados modelos para cada uma das bases de dados, e os resultados são mostrados na Tabela abaixo:

Tabela 5.1: *Resultados dos testes iniciais.*

Base de dados	Taxa de acertos
Inatel	91,6667%
Unicamp	90,177%

Realizados estes testes, partiu-se para o treinamento discriminativo dos HMMs. Os resultados destas investigações são mostrados nas seções a seguir.

5.2 Determinação do passo de aprendizagem

Como visto no Capítulo 3, a fórmula para reestimação dos parâmetros segundo o algoritmo Segmental GPD é

$$\Lambda_{n+1} = \Lambda_n - \varepsilon \nabla l(X_n|\Lambda n)$$

O gradiente $\nabla l(X_n|\Lambda n)$ indica a *direção* em que os parâmetros devem ser alterados, e o passo de aprendizagem ε, o *quanto* se deve caminhar na direção indicada pelo gradiente.

Evidentemente, como a superfície de erro é desconhecida, um passo de aprendizagem muito grande pode levar a um ponto onde o erro é eventualmente maior que o atual, enquanto que um passo de aprendizagem muito pequeno leva a uma convergência muito lenta do treinamento.

Na literatura de redes neurais, usa-se um passo de aprendizagem na faixa de $0,5$ a $0,8$, mas este resultado não pode ser aproveitado por causa de uma particularidade do algoritmo Segmental GPD: a constante de normalização γ. Em todas as equações de reestimação, o passo de aprendizagem aparece multiplicando esta constante. Assim, para γ muito pequeno, ε deve assumir valores maiores do que uma situação onde se tem γ um pouco maior.

Optou-se então por fazer uma varredura para vários valores de ε para tentar encontrar o valor mais adequado. O intervalo inicial invertigado foi $\varepsilon \in [1,8]$, com passo 1. Como os melhores resultados foram obtidos para passos pequenos, fez-se também um teste para $\varepsilon = 0,1$. A seguir são mostrados os resultados dos testes realizados tanto para a base da Unicamp como para a base do Inatel.

Base de Dados da UNICAMP

Os resultados apresentados nas figuras 5.1 a 5.13 mostram o desempenho do sistema para vários valores do passo de aprendizagem ε, e para $\gamma = \eta = 0,001$.

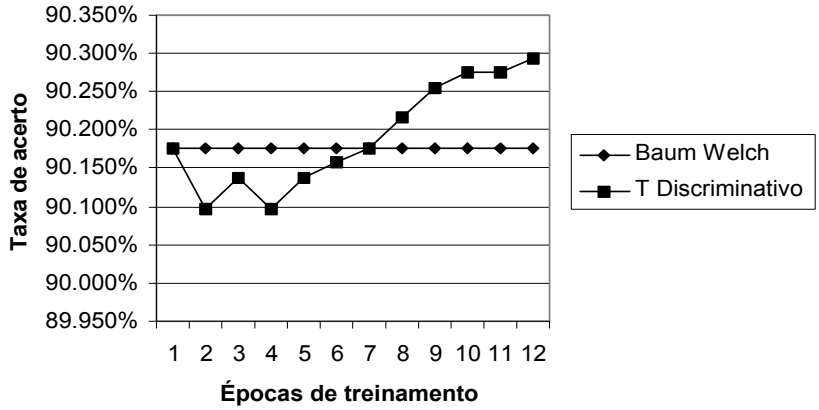

Figura 5.1: *Desempenho do sistema com passo de aprendizagem 0.1.*

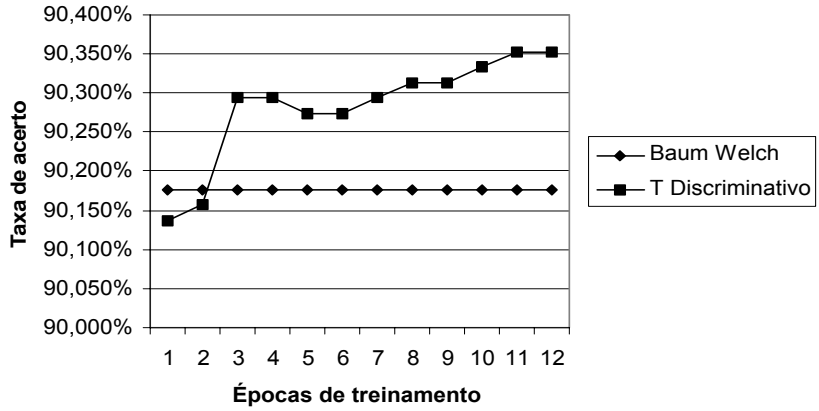

Figura 5.2: *Desempenho do sistema com passo de aprendizagem 0.3.*

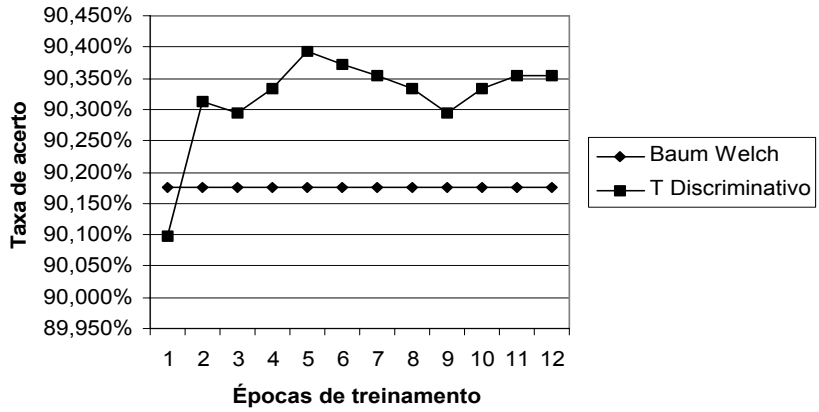

Figura 5.3: *Desempenho do sistema com passo de aprendizagem 0.5.*

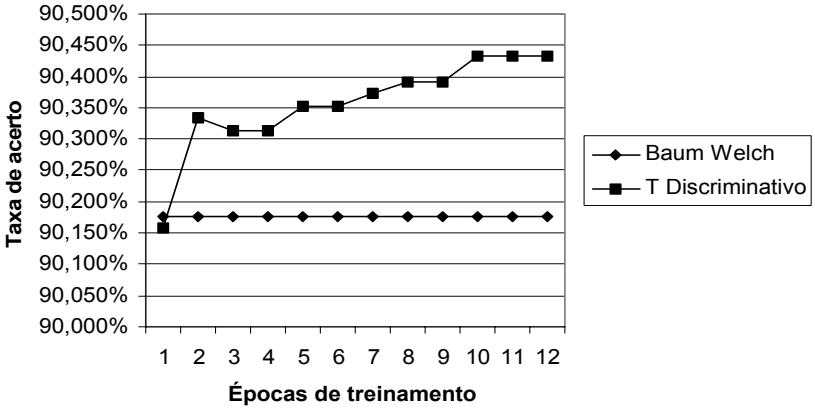

Figura 5.4: *Desempenho do sistema com passo de aprendizagem 0.7.*

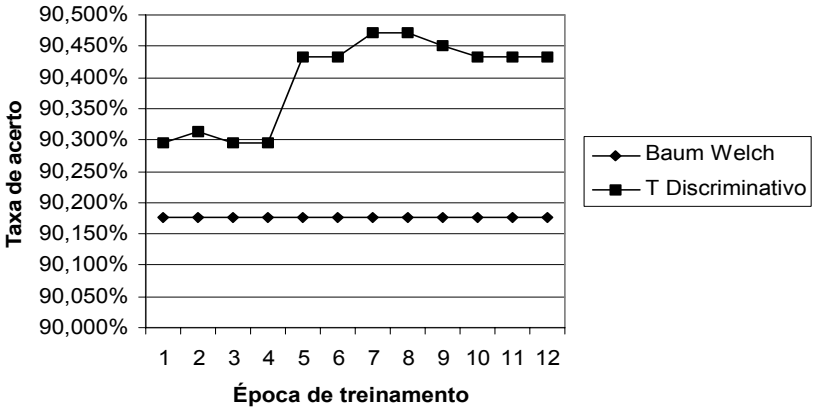

Figura 5.5: *Desempenho do sistema com passo de aprendizagem 0.9.*

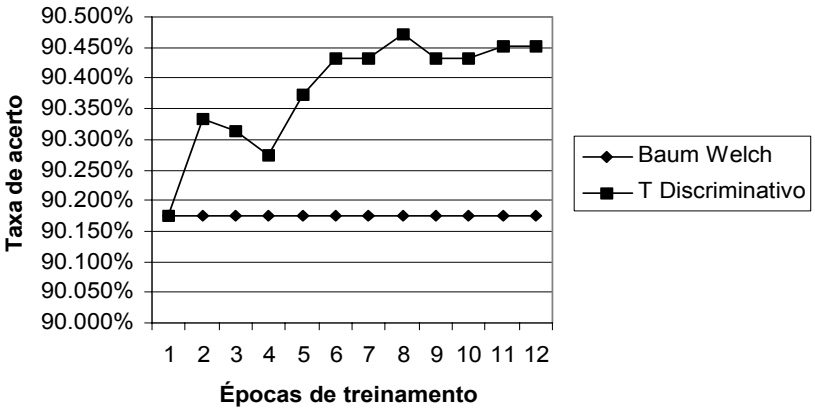

Figura 5.6: *Desempenho do sistema com passo de aprendizagem 1.*

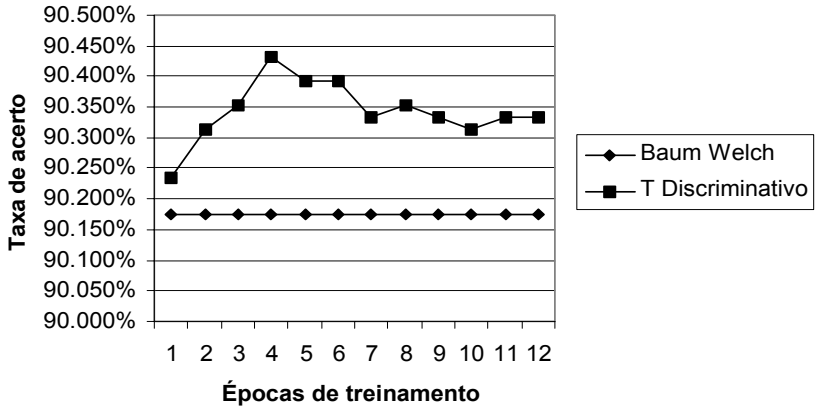

Figura 5.7: *Desempenho do sistema com passo de aprendizagem 2.*

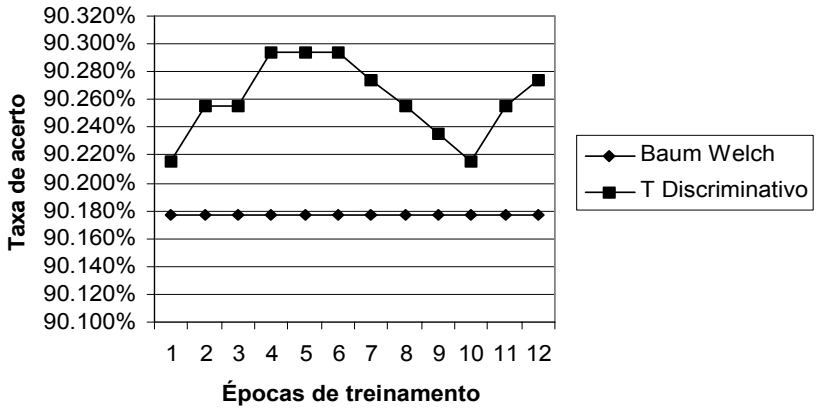

Figura 5.8: *Desempenho do sistema com passo de aprendizagem 3.*

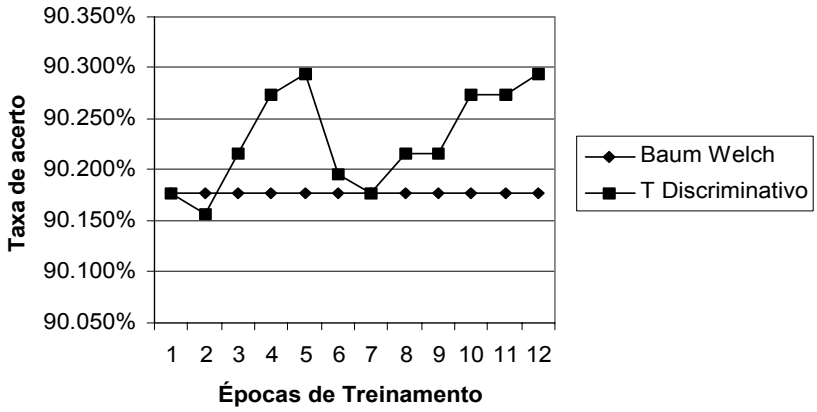

Figura 5.9: *Desempenho do sistema com passo de aprendizagem 4.*

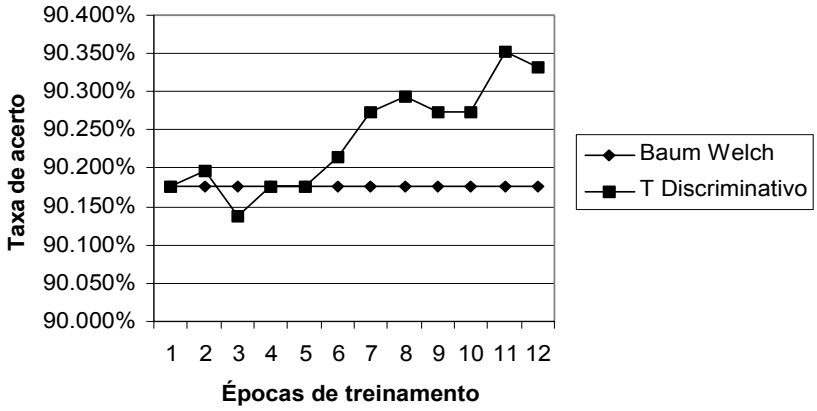

Figura 5.10: *Desempenho do sistema com passo de aprendizagem 5.*

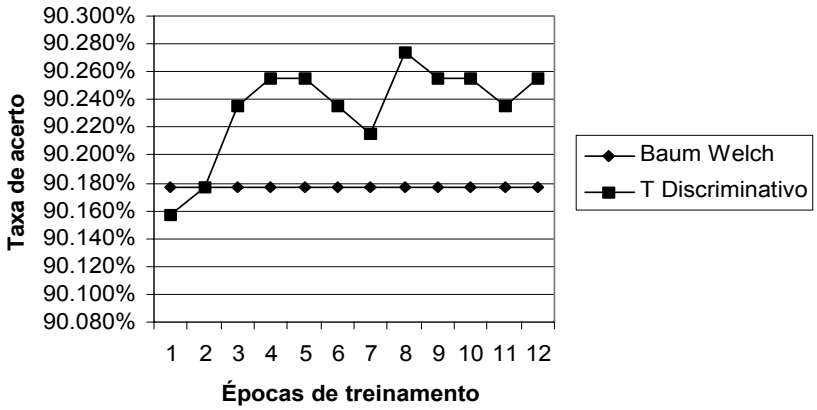

Figura 5.11: *Desempenho do sistema com passo de aprendizagem 6.*

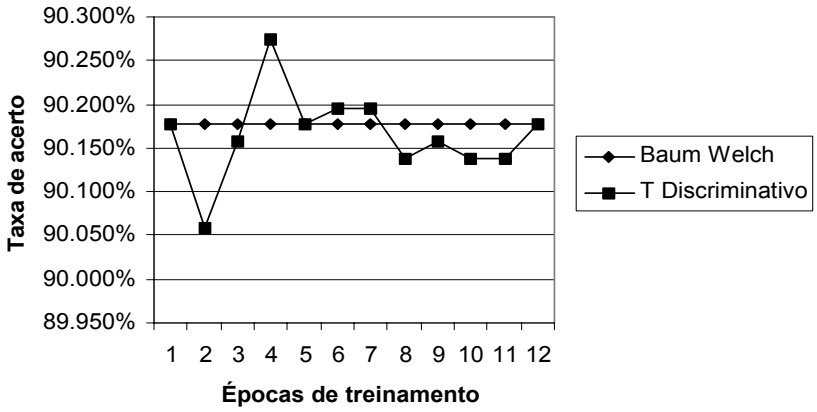

Figura 5.12: *Desempenho do sistema com passo de aprendizagem 7.*

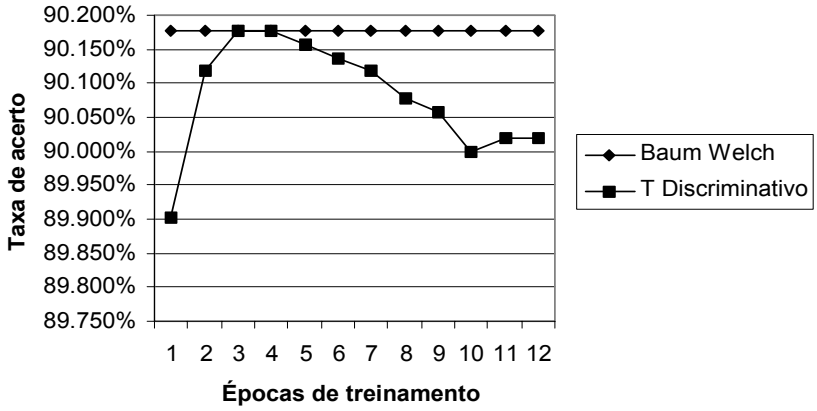

Figura 5.13: *Desempenho do sistema com passo de aprendizagem 8.*

Base de Dados do INATEL

Como nos testes anteriores, os resultados apresentados nas figuras 5.14 a 5.22 mostram o desempenho do sistema para vários valores do passo de aprendizagem ε, e para $\gamma = \eta = 0,001$.

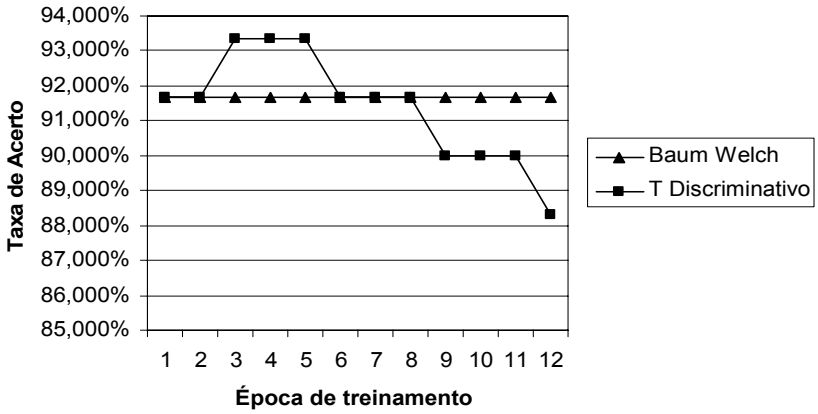

Figura 5.14: *Desempenho do sistema com passo de aprendizagem 0.1.*

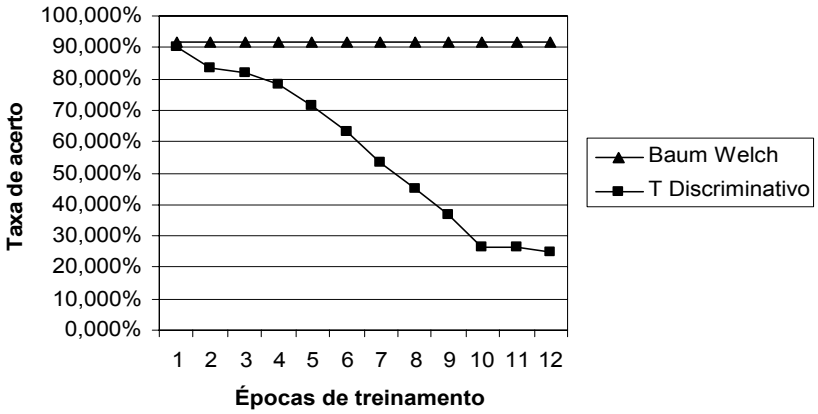

Figura 5.15: *Desempenho do sistema com passo de aprendizagem 1.*

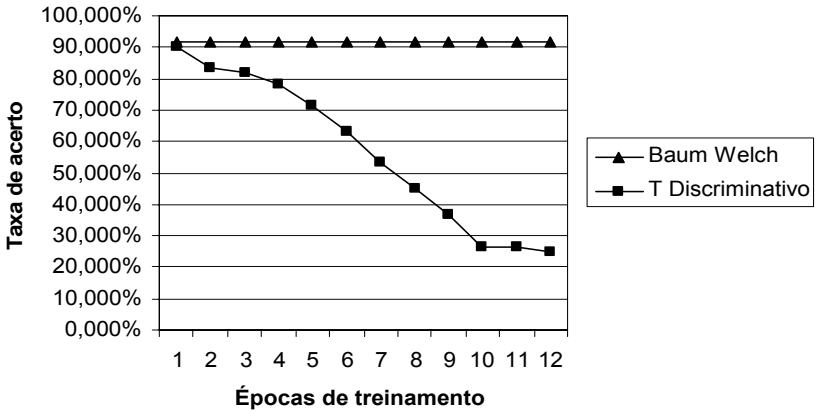

Figura 5.16: *Desempenho do sistema com passo de aprendizagem 2.*

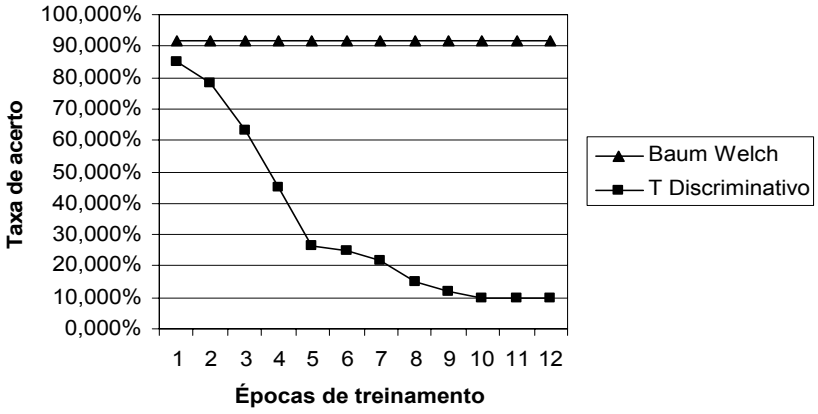

Figura 5.17: *Desempenho do sistema com passo de aprendizagem 3.*

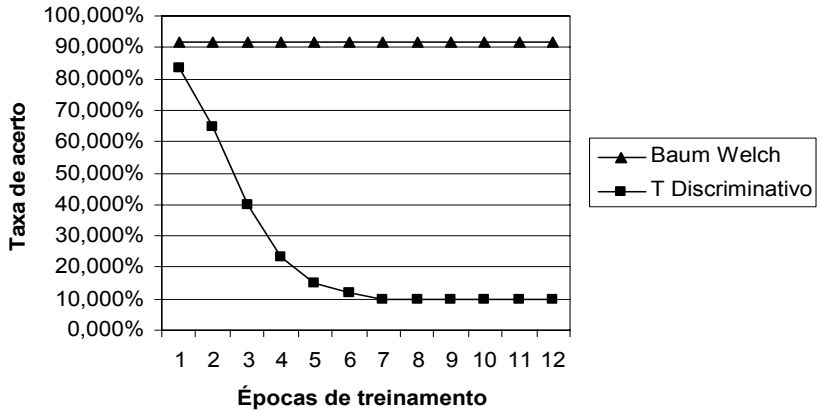

Figura 5.18: *Desempenho do sistema com passo de aprendizagem 4.*

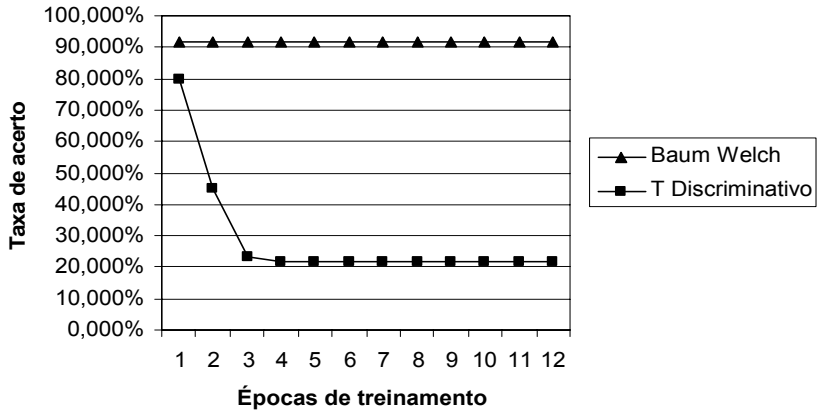

Figura 5.19: *Desempenho do sistema com passo de aprendizagem 5.*

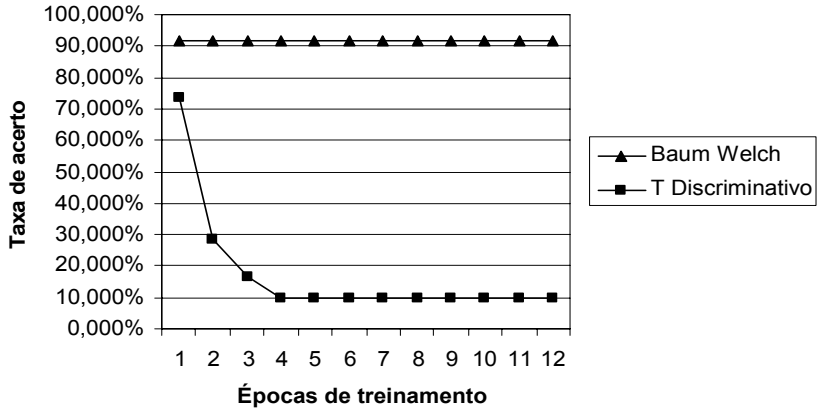

Figura 5.20: *Desempenho do sistema com passo de aprendizagem 6.*

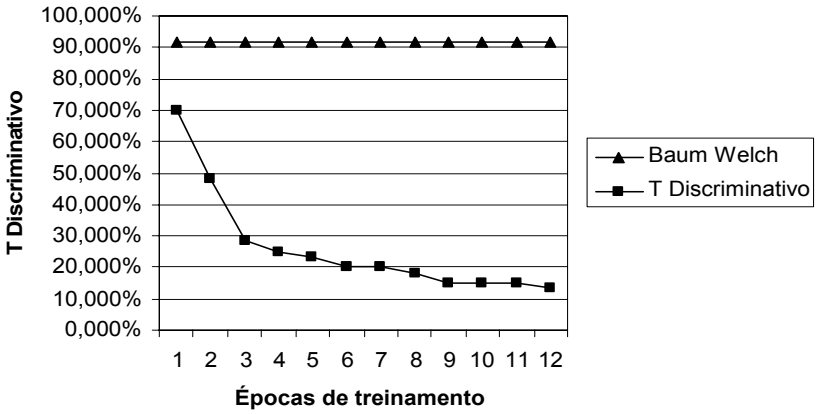

Figura 5.21: *Desempenho do sistema com passo de aprendizagem 7.*

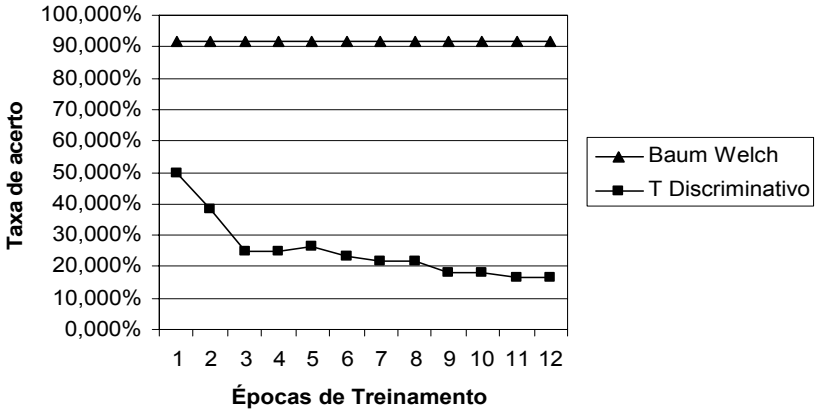

Figura 5.22: *Desempenho do sistema com passo de aprendizagem 8.*

Análise dos resultados

Para a base de dados da Unicamp observa-se que houve um pequeno ganho utilizando o treinamento discriminativo, tendo a taxa de acertos subido de 90,177% (obtida com o sistema treinado pelo algoritmo Baum-Welch) para 90,471%. Este ganho foi obtido com passo de aprendizagem igual a 1, após 8 épocas de treinamento. Observou-se também que o aumento do passo de aprendizagem não deteriora de forma significativa o desempenho do sistema (ver Figuras 5.1 a 5.13). Somente com passo a partir de 8 observou-se uma piora no desempenho em relação à taxa de acertos.

Já para a base de dados do Inatel observou-se um comportamento diferente: a taxa de acertos subiu de 91,667% (Baum-Welch) para 93,333% com o uso do

treinamento discriminativo. O melhor desempenho foi obtido com passo de aprendizagem igual 0,1, na terceira época. Entretanto, ao contrário do que aconteceu com a base de dados da Unicamp, observou-se uma degradação bastante acentuada na taxa de acertos para passos de aprendizagem já a partir de 1 (ver Figuras 5.15 a 5.22).

Estes resultados parecem ser coerentes, pois como o passo de aprendizagem ótimo para a base de dados da Unicamp foi uma ordem de grandeza maior que o passo ótimo para a base de dados do Inatel, era de se esperar que o aumento deste fosse mais sentido para a base de dados do Inatel.

5.3 Determinação do número de épocas de treinamento

A observação das Figuras 5.1 a 5.22 mostra que, diferentemente do treinamento convencial via Baum-Welch, o treinamento discriminativo não garante que a verossimilhança média sempre aumente: dependendo do passo de aprendizagem, o treinamento pode até divergir. Além disso observou-se que, a exemplo do que ocorre com as redes neurais, depois de certo tempo a taxa de acertos começa a cair, indicando que o sistema está ficando "viciado" nos dados de treinamento.

Optou-se então por utilizar um critério de parada similar ao utilizado no treinamento de redes neurais: o treinamento continua até que a taxa de acertos, verificada em um conjunto de validação cruzada, pare de aumentar. Devido à escassez de material de treinamento, optou-se por fazer a validação cruzada utilizando o próprio conjunto de testes. Este embora não seja um procedimento correto, não invalida as conclusões aqui apresentadas.

5.4 Ordem de apresentação das locuções

O algoritmo Segmental GPD apresenta grande similaridade com o algoritmo Back-Propagation utilizado para o treinamento de redes neurais. Um dos problemas do Back Propagation é a tendência de se viciar nos dados de treinamento, como discutido na seção anterior. Outra questão que se apresenta neste sentido é que o Back Propagation se vicia não somente nos dados de treinamento, como também na ordem em que estes são apresentados.

Para verificar este ponto realizou-se um teste com os exemplos de treinamento sendo apresentados sempre na mesma ordem, e outro, com os exemplos reordenados de forma aleatória a cada época de treinamento, e constatou-se que,

diferentemente do algoritmo Back Propagation, o algoritmo Segmental GDP não sofre deste tipo de problema.

5.5 Determinação do conjunto de locuções para o treinamento discriminativo

O algoritmo Segmental GPD é extremamente custoso em termos computacionais, pois para cada locução de treinamento atualiza os parâmetros de todos os modelos do vocabulário. Como o vocabulário do sistema é fixo, uma possível redução no custo computacional do treinamento seria fazê-lo não com todo o conjunto de treinamento, mas apenas com parte deste.

O problema a ser resolvido então é o de determinar qual subconjunto das locuções de treinamento deveria ser utilizado. Uma escolha que pareceu ser bastante razoável foi selecionar as locuções de treinamento que o sistema reconheceu de forma errada após cada época de treinamento. Estas seriam as locuções *problemáticas*, e o retreinamento do sistemas apenas com elas faria com que o sistema "aprendesse" a lidar com elas.

Realizou-se então um teste que consistia em fazer o treinamento dicriminativo usando toda a base de dados de treinamento (TD1), e outro treinamento utilizando apenas as locuções de treinamento que o sistema reconheceu de forma errada (TD2). Para estes testes foi utilizada apenas a base de dados da Unicamp, e os resultados podem ser vistos na Tabela 5.2.

Tabela 5.2: *Resultados dos testes para verificação do conjunto de locutores para o treinamento discriminativo.*

Baum-Welch	TD1	TD2
90,177%	90,471%	90,255%

A análise da Tabela 5.2 mostra que realmente a utilização de apenas um subconjunto das locuções de treinamento leva a uma melhora na taxa de acertos, mas a utilização de todas as locuções leva a resultados melhores.

Isto confirma a teoria: o algoritmo Segmental GPD aumenta a probabilidade do modelo correto emitir a locução de treinamento e diminui a probabilidade dos modelos concorrentes emitirem a mesma. Desta forma, quanto maior o material de treinamento, mais informações o algoritmo tem para realizar esta discriminação, levando a resultados melhores.

Observação: Tempo de processamento

Os resultados apresentados utilizando a base de dados da UNICAMP o tempo médio de processamento do treinamento discriminativo foi de 3 horas enquanto que o tempo gasto realizando o treinamento convencional foi de 30 minutos, e para a base de dados do INATEL o tempo médio de processamento do treinamento discriminativo foi de 10 horas enquanto que o tempo gasto realizando o treinamento convencional foi de 1 minuto. Estes tempos, foi obtido utilizando um computador com processador pentium 4, com 256Mbytes de memória.

Capítulo 6

Conclusões

Tradicionalmente utiliza-se para o treinamento de HMMs o algoritmo Baum-Welch, baseado no critério de máxima verossimilhança que procura, a cada época, aumentar a probabilidade do modelo gerar a locução a ele correspondente.

O Segmental GPD é um algoritmo de treinamento discriminativo que procura, a cada época, não apenas maximizar a probabilidade do modelo correto gerar a locução de treinamento, mas também minimizar a probabilidade dos modelos concorrentes (incorretos) gerarem a mesma. Isto geraria um poder de discriminação maior, levando a uma consequente melhora na taxa de acertos do sistema.

Neste trabalho foi realizado um estudo teórico detalhado, com a dedução de todas as expressões de reestimação, e a implementação prática do algoritmo Segmental GPD para treinamento de HMMs contínuos em um contexto de reconhecimento de palavras isoladas, com modelamento por palavras.

Os testes foram realizados sobre duas bases de dados, ambas multilocutores: a primeira com 10 palavras no vocabulário e 12 locutores, gerada no Inatel, e a segunda, com 50 palavras e 69 locutores, gerada pelo Dr. José Antônio Martins e se encontra alojada no LPDF-UNICAMP. Como vetores acústicos, foram utilizados apenas 12 parâmetros mel-cepstrais, calculados em janelas de 20 ms, atualizadas a cada 10 ms. Antes do cálculo dos parâmetros, os sinais de voz foram pré-enfatizados através de um filtro passa-altas com função de transferência $H(z) = 1 - 0,95z^{-1}$ e janelados com uma janela de Hamming.

Com o algoritmo Baum-Welch conseguiu-se uma taxa de acertos de 90,177% e 91,667% para as bases de dados da Unicamp e do Inatel, respectivamente. Já utilizando o treinamento discriminativo conseguiu-se aumentar a taxa de acertos para 90,471% e 93,333% respectivamente.

É importante ressaltar também alguns fatos notados quando da implementação prática do treinamento discriminativo, fatos estes que embora não constituam uma contribuição científica, fazem toda a diferença na hora da implementação:

- O algoritmo Segmental GPD é bastante instável numericamente, de forma que as constantes de normalização γ e η devem ser cuidadosamente escolhidas. Neste trabalho utilizou-se $\gamma = \eta = 0,001$.

  O passo de aprendizagem ε aparece sempre multiplicando o termo γ, de forma que o valor ótimo para este depende do valor atribuído a γ. Desta forma, para a sua determinação, foi feita uma varredura com valores entre 0,1 e 12. Para a base de dados da Unicamp, o valor mais adequado foi $\varepsilon = 1$, e para a base de dados do Inatel, $\varepsilon = 0,1$.

O algoritmo Segmental GPD tem fortes semelhanças com o algoritmo Back Propagation, utilizado para a atualização dos pesos sinápticos de uma rede neural. Desta forma, foram feitos dois testes para verificar se o Segmental GPD também apresenta um problema bastante conhecido do Back Propagation, que é o de o sistema ficar "viciado" nos dados de treinamento, efeito conhecido como *overtraining*[26]:

- Verificou-se em todos os testes (Figuras 5.1 a 5.22), que a taxa de acertos sobe inicialmente, atinge um pico, e depois tende a cair, com o número de épocas, o que caracteriza o *overtrainig*. Desta forma, utilizou-se como critério de parada para o treinamento a verificação da taxa de acertos em um conjunto de validação. Como as bases de dados utilizadas são muito pequenas, o conjunto de validação utilizado foi o próprio conjunto de testes, um procedimento teoricamente inadequado, mas que não invalida o procedimento.

- O Back Propagation fica viciado não somente nos dados de treinamento, mas também na *ordem* em que os mesmos são apresentados ao sistema. Para verificar se este efeito também acontece com o Semental GPD, fez-se um treinamento apresentando-se os dados sempre na mesma ordem, e outro, reordenando-os de forma aleatória a cada época. Não foi observada nenhuma diferença de desempenho entre as duas estratégias de forma que, pelo menos para a configuração testada, este tipo de problema não ocorre.

Um questionamento final foi que, como o algoritmo de treinamento discriminativo é extremamente custoso em termos computacionais, poderia-se conseguir uma redução no tempo de treinamento utilizando-se apenas uma porção do conjunto de treinamento. A escolha natural seria utilizar apenas as locuções de treinamento em que o sistema errase ao reconhecer, pois estas poderiam conter informações ainda não assimiladas pelo sistema.

Os resultados obtidos mostram que realmente a utilização de apenas um subconjunto das locuções de treinamento leva a uma melhora na taxa de acertos,

mas a utilização de todas as locuções leva a resultados melhores. Isto confirma os resultados teóricos, que mostram que o algoritmo Segmental GPD realmente utiliza tanto os erros como os acertos para atualizar os parâmetros do sistema.

6.1 Sugestões para Trabalhos Futuros

Como sugestões para trabalhos futuros, pode-se citar:

- estudo e implementação da função erro utilizando apenas a diferença entre o modelo correto e o modelo mais próximo. Neste trabalho foi utilizada a diferença entre o modelo correto e a média dos concorrentes na etapa de treinamento,
- estudo e implementação do treinamento discriminativo para sistemas de reconhecimento de fala contínua utilizando modelos de subunidades fonéticas,
- determinação de um conjunto de equações de reestimação usando multiplicadores de Lagrange para garantir que os parâmetros reestimados atendam automaticamente às restrições probabilísticas impostas pelos HMMs.
- Uso do momentum nas equações de reestimação, como parâmetro adicional ao passo de aprendizagem, como costuma se usar no algoritmo Back Propagation [26].

Referências Bibliográficas

[1] Disponível na Internet via WWW. URL: <http://www.timaster.com.br/revista/artigos/main_artigo.asp?codigo=435>, (15/06/2003).

[2] Disponível na Internet via WWW. URL: <http://www.telecomweb.com.br/noticias/keyword.asp?key=portaisdevoz>, (15/06/2003).

[3] F. Valente, Tecnologia IBM fornece aos novos automóveis Honda um sistema de navegação ViaVoiceh, disponível na Internet via WWW. URL: <http://www.estado.estadao.com.br/jornal/suplem/info/99/11/22/info014.html>, (15/06/2003).

[4] Disponível na Internet via WWW. URL: <http://www-3.ibm.com/software/speech/dev/demo.html>, (15/06/2003).

[5] Disponível na Internet via WWW. URL: <www.telemigcelular.com.br>, (15/06/2003)

[6] Disponível na Internet via WWW. URL: <www.lucent.com>, (15/06/2003)

[7] Disponível na Internet via WWW. URL: <http://intervox.nce.ufrj.br/motrix/>, (15/06/2003)

[8] Disponível na Internet via WWW. URL: <http://jbonline.terra.com.br/jb/papel/cadernos/internet/2002/07/28/> jorinf20020728007.html¿, (15/06/2003)

[9] Juang, B. H., Fellow, W. C. and Chin-Hui, L., Minimum Classification Error Rate Methods for Speech Recognition, IEEE, 257-265, 1997.

[10] Martins, J. A. Avaliação de Diferentes Técnicas Para Reconhecimento de Fala, Tese de Doutorado, Universidade de Campinas, 1997.

[11] Scavone,A. P. R., Reconhecimento de Palavras Por Modelos Ocultos de Markov, Tese de Mestrado, Universidade de São Paulo, 1996.

[12] Rabiner, L. and Juang, B. H., Fundamentals of Speech Recognition, Prentice Hall Signal Processing Series, 1993.

[13] Dias, R. S. F., Normalização de Locutor Em Sistema de Reconhecimento de Fala, Tese de Mestrado, Universidade Estadual de Campinas, 2000.

[14] Ynoguti, C. A., Reconhecimento de Fala Contínua Usando Modelos Ocultos de Markov, Tese de Doutorado, Universidade Estadual de Campinas, 1999.

[15] Morais, E. S., Reconhecimento Automático de Fala Contínua Empregando Modelos Híbridos ANN + HMM, Universidade Estadual de Campinas, 1997.

[16] McDermott, E., Discriminative Training for Speech Recognition,Waseda University, 1997.

[17] Shigeru, K., Chin, H. L., and Biing H. J. , New Discriminative Training Algorithms Based on the Generalized Probabilistic Descent Method, IEEE, 299-308, 1991.

[18] Chi L.S., Chin H. L., Bing, H. J. and Aoron, E. R., Speaker Recognition Based on Minimum Error Discriminative Training, IEEE, 325-328, 1994.

[19] Jung, K. C. and Frank, K. S., An N-Best Candidates-Based Discriminative Training for Speech Recognition Aplications, IEEE, 206-216, 1994.

[20] Chow W., Juang, B. H. and Lee, C. H., Segmental GPD Training of HMM Based Speech Recognizer, IEEE, 473-476, 1993.

[21] Figueredo, F. L., Segmentação Automática e Treinamento Discriminativo Aplicados a Um Sistema de Reconhecimento de Dígitos Conectados, Universidade Estadual de Campina, 1999.

[22] Qiang, H. and Chorkin, C., The Gradient Projection Method for the Training of Hidden Markov Models, Speech Communication, 307-313, 1993.

[23] Chou W., Lee, C.H. and Juang, B. H. , Minimum Error Rate Training Based on N-Best String Models, IEEE, 652-655, 1993.

[24] Schluter, R., Macherey, W., Kanthak, S., Ney, H. and Welling, L., Comparison of Optimization Methods for Discriminative Training Criteria, University of Technology.

[25] Picone, J.W., Signal Modeling Techiques in Speech Recognition, IEEE, 1215-1247, 1993.

[26] HAYKIN, Simon. *Neural Networks: a comprehensive foundation.* Prentice Hall. 1999.