UNIVERSIDADE FEDERAL DO PARÁ
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

TALISMAN CLÁUDIO DE QUEIROZ TEIXEIRA JÚNIOR

CLASSIFICAÇÃO FONÉTICA UTILIZANDO BOOSTING E SVM

DM 02/2006

UFPA/CT/PPGEE
Campus Universitário do Guamá
Belém - Pará - Brasil
2006

UNIVERSIDADE FEDERAL DO PARÁ
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

TALISMAN CLÁUDIO DE QUEIROZ TEIXEIRA JÚNIOR

CLASSIFICAÇÃO FONÉTICA UTILIZANDO BOOSTING E SVM

Dissertação submetida à Banca Examinadora do Programa de Pós - Graduação em Engenharia Elétrica da Universidade Federal do Pará para a obtenção do grau de Mestre em Engenharia Elétrica.

UFPA/CT/PPGEE
Campus Universitário do Guamá
Belém - Pará - Brasil
2006

UNIVERSIDADE FEDERAL DO PARÁ
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

TALISMAN CLÁUDIO DE QUEIROZ TEIXEIRA JÚNIOR

CLASSIFICAÇÃO FONÉTICA UTILIZANDO BOOSTING E SVM

Esta Dissertação de Mestrado foi julgada adequada para a obtenção do título de Mestre em Engenharia Elétrica, na área de Telecomunicações, e aprovada na sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Pará.

BANCA EXAMINADORA:

_____

Prof. Ph.D.Aldebaro Barreto da Rocha Klautau Júnior
(ORIENTADOR – DEEC – UFPA)

_____

Prof. Dr. Evaldo Gonçalves Pelaes
(CO – ORIENTADOR – DEEC – UFPA)

_____

Prof. Dr. Antônio Marcos de Lima Araújo
(MEMBRO – IESAM)

_____

Prof. Dr. Agostinho Luiz da Silva Castro
(MEMBRO – DEEC – UFPA)

VISTO:

_____

Prof. Dr. Evaldo Gonçalves Pelaes
COORDENADOR DO PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Para a minha mãe Conceição e minha esposa Adriana,
pelo suporte emocional nos momentos difíceis.

AGRADECIMENTOS

A Deus, em primeiro lugar, pelo dom da vida e do pensar;

A minha mãe, Conceição, pela educação que me deu, pelo amor durante toda a minha vida e por ter suportado sozinha a minha criação;

A minha esposa, Adriana, em especial, pela dedicação, compreensão, amor e alegria em todos os momentos;

Ao meu orientador Aldebaro (uma competência rara), pelos ensinamentos e pela compreensão nos momentos em que parecia que nada ia dar certo;

Ao meu co-orientador Pelaes, por me fornecer a base sólida que me permitiu entender este assunto e por me orientar nos primeiros anos deste curso;

A todos os professores, que durante a minha vida acadêmica, contribuíram de alguma forma na minha educação;

Ao professor Antônio Marcos, em especial, pela confiança em mim depositada e pelo incentivo em realizar este curso, dando todo o suporte necessário;

Aos colegas do LAPS, principalmente o Rafael, o Tales e o Murilo, que foram as minhas bússolas no Laboratório quando eu me encontrava perdido;

A todas as pessoas que de alguma forma contribuíram na minha formação.

RESUMO

Para compor um sistema de Reconhecimento Automático de Voz, pode ser utilizada uma tarefa chamada Classificação Fonética, onde a partir de uma amostra de voz decide-se qual fonema foi emitido por um interlocutor. Para facilitar a classificação e realçar as características mais marcantes dos fonemas, normalmente, as amostras de voz são pré-processadas através de um *front-end*. Um *front-end*, geralmente, extrai um conjunto de parâmetros para cada amostra de voz. Após este processamento, estes parâmetros são inseridos em um algoritmo classificador que (já devidamente treinado) procurará decidir qual o fonema emitido. Existe uma tendência de que quanto maior a quantidade de parâmetros utilizados no sistema, melhor será a taxa de acertos na classificação. A contrapartida para esta tendência é o maior custo computacional envolvido. A técnica de Seleção de Parâmetros tem como função mostrar quais os parâmetros mais relevantes (ou mais utilizados) em uma tarefa de classificação, possibilitando, assim, descobrir quais os parâmetros redundantes, que trazem pouca (ou nenhuma) contribuição à tarefa de classificação. A proposta deste trabalho é aplicar o classificador SVM à classificação fonética, utilizando a base de dados TIMIT, e descobrir os parâmetros mais relevantes na classificação, aplicando a técnica *Boosting* de Seleção de Parâmetros.

Palavras-Chave: **Fonemas, Classificação, SVM, Parâmetros, *Front-end*, Seleção de Parâmetros, *Boosting*, TIMIT.**

ABSTRACT

With the aim of setting up a Automatic Speech Recognition (ASR) system, a task named Phonetic Classification can be used. That task consists in, from a speech sample, deciding which phoneme was pronounced by a speaker. To ease the classification task and to enhance the most marked characteristics of the phonemes, the speech samples are usually pre-processed by a front-end. A front-end, as a general rule, extracts a set of features to each speech sample. After that, these features are inserted in a classification algorithm, that (already properly trained) will try to decide which phoneme was pronounced. There is a rule of thumb which says that the more features the system uses, the smaller the classification error rate will be. The disadvantage to that is the larger computational cost. Feature Selection task aims to show which are the most relevant (or more used) features in a classification task. Therefore, it is possible to discover which are the redundant features, that make little (or no) contribution to the classification task. The aim of this work is to apply SVM classificator in Phonetic Classification task, using TIMIT database, and discover the most relevant features in this classification using Boosting approach to implement Feature Selection.

Keywords: Phonemes, Classification, SVM, Features, Front-end, Feature Selection, Boosting, TIMIT.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AGC – *Automatic Gain Control*

AR – Autorregressivo

ASR – *Automatic Speech Recognition*

CD – *Context Dependent*

DCT – *Discrete Cosine Transform*

DCTC – *Discrete Cosine Transform Coefficient*

DFT – *Discrete Fourier Transform*

DTW – *Dynamic Time Warping*

ECOC – *Error-correcting Output Code*

ED – *Envelope Detector*

EUA – Estados Unidos da América

FFT – *Fast Fourier Transform*

GMM – *Gaussian Mixture Models*

HMM – *Hidden Markov Model*

IIR – *Infinite Impulse Response*

LDC – *Linguistic Data Consortium*

LFCC – *Linear-Frequency Cepstrum Coefficients*

LP – *Linear Predition*

LPC – *Linear Predition Coefficients*

MFCC – *Mel-Frequency Cepstrum Coefficients*

MIT – *Massachussets Institute of Technology*

NIST – *National Institute of Standards and Technology*

PLL – *Phase-Locked Loop*

PLP – *Perceptual Linear Predictive*

RASTA – *Relative Espectral*

RAV – Reconhecimento Automático de Voz

SD –*Synchrony Detector*

SVM – *Support Vector Machine*

TI – *Texas Instruments*

TIMIT – *Texas Instruments / Massachussets Institute of Technology*

SUMÁRIO

1 Introdução

A tarefa de reconhecimento automático de voz tem se tornado um dos assuntos mais importantes na área de Processamento Digital de Sinais e, principalmente, na área de Automação e Controle. A importância advém da própria ficção científica que vislumbra, para o futuro, uma interação homem-máquina através da voz.

O objetivo de uma máquina que executa a tarefa de reconhecimento automático de voz é, a partir de amostras do sinal de voz, tentar entender o conteúdo da sentença, tentando montar frases, que derivam de palavras, que por sua vez são compostas por fonemas.

Um dos problemas encontrados em Processamento Digital de Voz é a Classificação Fonética, que permite que uma máquina escolha dentre um conjunto de fonemas, qual o emitido. Existe uma probabilidade bem alta de que, se a taxa de erro for baixa em um sistema de Classificação Fonética, o será também em RAV.

Existem várias decisões que devem ser tomadas ao se projetar um sistema de Reconhecimento Automático de Voz (RAV). Algumas delas são: adota-se um único conjunto de parâmetros, ou vários (Bourlard et al, 1998); adota-se um conjunto homogêneo ou heterogêneo de parâmetros (Halberstadt, 1998); qual o classificador; etc. Estes tópicos serão discutidos nas seções posteriores, mas pode-se adiantar que este trabalho utilizou como classificador a técnica SVM (Cortes; Vapnik, 1995), com *front-end´s* heterogêneos.

Todo trabalho voltado para RAV tem como objetivo reduzir a taxa de erros no reconhecimento de um determinado conjunto de fonemas. Este trabalho não foge à regra. Mas, além disso, procura mostrar uma forma de se descobrir quais os parâmetros mais adequados para classificar pares específicos de fonemas, o que possibilitará reduzir o custo computacional dos sistemas futuros. Este método é conhecido como Seleção de Parâmetros (Hall, 1999) e será discutido com detalhes no Capítulo 3. Neste trabalho, foi sugerida a técnica conhecida como *Boosting* (Freund; Schapire, 1995), que também será descrita no mesmo capítulo.

A idéia de se utilizar Boosting para melhorar um classificador surgiu de Tieu;Viola (2000). Em sua publicação, foi utilizada a técnica *Boosting* para melhorar a taxa de acertos de um classificador de imagens. Este trabalho aproveitou a mesma técnica e aplicou em classificadores SVM para classificar fonemas.

1.1 Objetivos do Trabalho

Este trabalho tem como objetivos:

- Projetar um front-end heterogêneo utilizando SVM, objetivando chegar a uma taxa de erros compatível com publicações internacionais, independente do alto custo computacional
- Descobrir, utilizando Seleção de Parâmetros, quais os parâmetros que trazem mais informação consigo, ou, explicando de outra forma, quais aqueles parâmetros menos redundantes.
- Descobrir, para alguns pares de fonemas, quais os parâmetros mais adequados, ou mais relevantes, para a distinção entre os dois fonemas.

1.2 Organização da Dissertação

Esta dissertação está dividida em cinco capítulos. O segundo capítulo explica o conceito de *front-end*, além de detalhar a taxonomia envolvida. Neste capítulo também são descritos os diversos *front-end´s* utilizados neste trabalho. O terceiro capítulo aborda o classificador escolhido neste trabalho, bem como conceitua Seleção de Parâmetros e descreve a técnica *Boosting*. O quarto capítulo expõe os resultados obtidos. O quinto capítulo é reservado para as conclusões e possíveis trabalhos futuros.

2 Front-end´s

Em uma tarefa de Classificação Fonética, o objetivo central é observar uma amostra de voz e afirmar qual fonema foi emitido pelo interlocutor. Como toda tarefa de classificação, o sistema que pode auxiliar nesta tarefa é constituído de três blocos principais: Pré-processamento, Extração de Parâmetros e Classificação (Duda; Hart, 2000). Costuma-se chamar os dois primeiros blocos de *Front-end*, que tem como função reduzir o conjunto de dados a um conjunto específico de parâmetros (*features*) que minimizem o erro no processo de classificação. O bloco seguinte – conhecido como Classificador – tem a função de, a partir dos parâmetros apresentados, tomar uma decisão de qual fonema foi apresentado. Obviamente, este exemplo simplório omite o fato de que o classificador foi treinado anteriormente, através de conjuntos de amostras de treino, que serão úteis para que o classificador "aprenda" os parâmetros característicos para cada fonema. Independente se o sistema está em fase de treinamento ou de testes, o *front-end* está sempre presente, para obter os parâmetros que servirão de entrada para o classificador.

Este capítulo trata dos *front-end´*s mais comuns utilizados neste trabalho, possibilitando com que fique mais clara a interpretação dos resultados encontrados.

2.1 Taxonomia

Os *front-end´*s podem ser classificados de acordo com a forma como foram projetados: baseados no conhecimento ou de forma automática. O primeiro é o mais tradicional e aproveita toda a experiência adquirida em psico-acústica e no processo da fala. Podemos citar como exemplos, o MFCC (Davis; Merlmerstein, 1980) e o PLP (Hermansky, 1990). A outra forma se baseia em utilizar uma técnica conhecida como seleção de parâmetros, que é uma técnica automática que auxilia na escolha dos melhores parâmetros para o *front-end*, eliminando assim parâmetros ruidosos e/ou redundantes (Hall, 1999). Esta técnica é a base deste trabalho e será descrita com detalhes no Capítulo 3.

Outro critério de classificação é com relação à utilização de um conjunto homogêneo ou de um conjunto heterogêneo de parâmetros. Quando se utiliza um conjunto homogêneo, o *front-end* é invariável para qualquer tipo de fonema apresentado. Este método é mais fácil de ser implementado, mas, intuitivamente, percebe-se que a classificação tende a melhorar

quando o *front-end* é adaptado ao fonema apresentado. Por exemplo, se a intenção é fazer a distinção entre o fonema vocálico /i/ (como em l**i**vro) e a vogal /u/ (como em l**u**ta), o ideal seria utilizar a segunda formante como parâmetro de diferenciação (Huang et al., 2001). Mas, se o objetivo é distinguir as palavras *pata* e *bata*, mais interessante seria estudar a Probabilidade de Vozeamento, pois o que distingue estas duas palavras é que o fonema /b/ é uma consoante plosiva vozeada (emite uma pequena quantidade de energia de baixa freqüência através das paredes da garganta), enquanto que o fonema /p/ não é (Rabiner; Juang, 1993).

2.2 Formantes

Os primeiros parâmetros que serão abordados são as formantes. No contexto de produção de voz, são definidas como sendo as freqüências de ressonância do trato vocal (Rabiner; Schafer, 1978). De acordo com Peterson; Barney (1952), as formantes desempenham um papel importante na decisão entre os tipos de fonemas vocálicos, pois para cada vogal existe uma conformação do conjunto língua/boca, que altera as dimensões da caixa de ressonância do trato vocal, alterando, por sua vez, as freqüências de ressonância. Pode ser verificado na Figura 1 o triângulo fonético de Peterson e Barney, onde se percebe, por exemplo, que o fonema /i/ tem uma freqüência de segunda formante F_2 em torno de 2 kHz, enquanto que a vogal /u/ possui uma F_2 de aproximadamente 1 kHz. Neste trabalho, foram utilizadas as quatro primeiras formantes (F_0 a F_4).

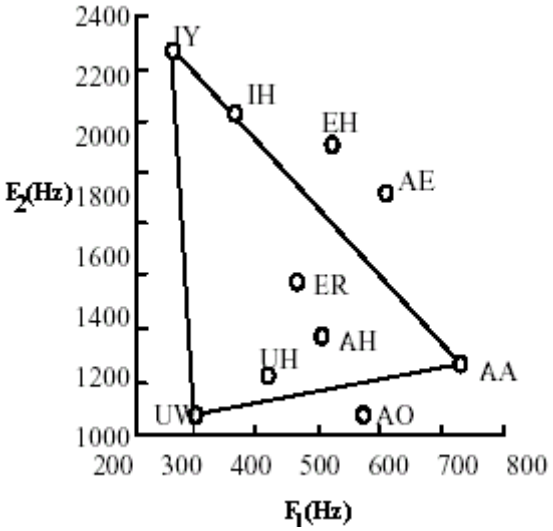

Figura 1 – O triângulo fonético
Fonte: Peterson; Barney (1952)

A extração das formantes foi implementada de acordo com o algoritmo desenvolvido por David Talkin, descrito em Talkin (1987, 1995). Estes algoritmos fazem parte do pacote Waves+, incorporados ao pacote Snack, que pode ser obtido em Snack (2006).

Outro parâmetro incluído foi a freqüência fundamental F_0, que está diretamente relacionada com a freqüência da excitação vocal, considerando o modelo fonte-sistema (Rabiner; Schafer, 1978). Esta excitação é gerada pela movimentação oscilatória (abertura e fechamento) das pregas vocais, basicamente na geração de fonemas vozeados (Huang et al., 2001). Pode ser observado na Figura 2, que representa a forma de onda da vogal /a/, a característica periódica que determina a freqüência fundamental.

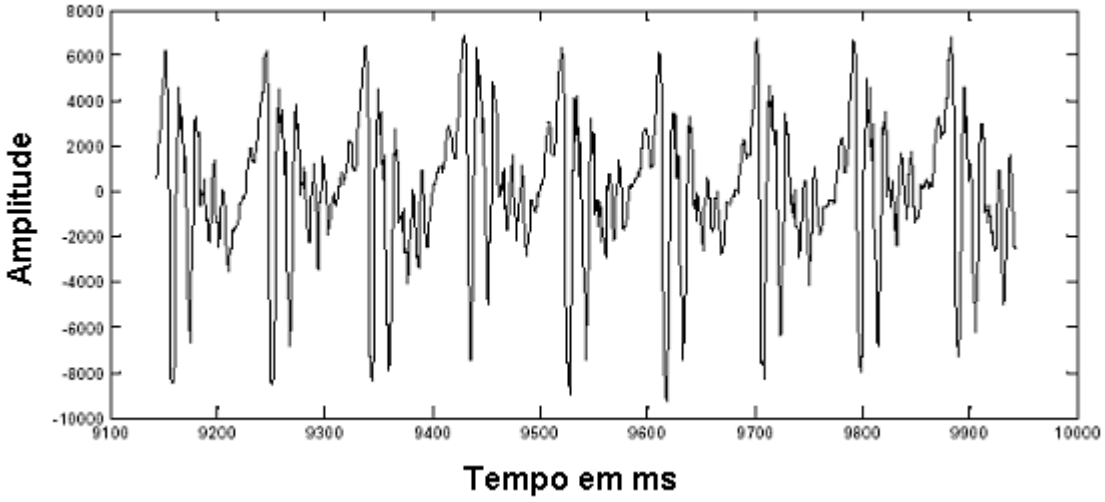

Figura 2 – Forma de onda da vogal /a/

Além das freqüências F_0 a F_4, outro parâmetro utilizado foi a Probabilidade de Vozeamento, que indica se o sinal é vozeado ou não, isto é, se o fonema gerado é baseado (ou possui uma componente) na vibração das cordas vocais. Basicamente, os fonemas vozeados são vogais, mas existem também consoantes consideradas vozeadas, como por exemplo, a fricativa vozeada /z/ e a plosiva vozeada /b/. Daí, a importância deste parâmetro para fazer distinção entre algumas classes de fonemas. Pode ser percebido, na Figura 3, que mostra o espectrograma da sílaba /bi/, onde se percebe uma freqüência baixa (vozeamento), antes da

liberação do ar (instante de tempo aproximadamente igual a 4 segundos) , característica das plosivas (Rabiner; Schafer, 1978). Como todo espectrograma, as frequências de maior

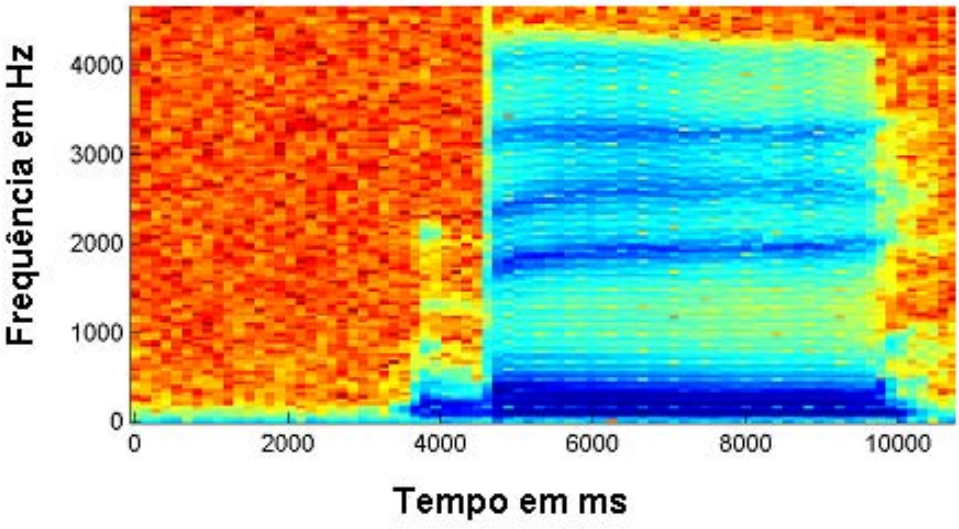

Figura 3 – Espectrograma da sílaba bi

amplitude são representadas pela cor azul, enquanto que as de menor amplitude são representadas pela cor vermelha.

2.3 *Perceptual Linear Predictive* (PLP)

O segundo grupo de parâmetros foi obtido com base no *front-end* denominado de PLP (*Perceptual Linear Predictive*), sendo que cada quadro é composto de 39 parâmetros (12 coeficientes, a energia, além de suas primeiras e segundas derivadas).

PLP é um front-end que aproveita três conceitos básicos da psico-acústica pra tentar modelar o sistema auditivo. Ele se utiliza de um modelo AR, constituído apenas de pólos, semelhante à LP, mas com a diferença que existe um pré-processamento psico-acústico em três etapas (Hermansky, 1990).

A primeira etapa faz um *warping* utilizando uma curva de transformação Bark-rad/s (Schroeder, 1977), conforme pode ser visto na Figura 4 e, em seguida, faz-se uma convolução entre o espectro de potência resultante com uma máscara de banda crítica, cujo formato pode

ser ilustrado na Figura 5 (Hermansky, 1990). Esta etapa equivale a implementar um banco de filtros, sendo que o formato dos filtros é aproximadamente constante na escala Bark.

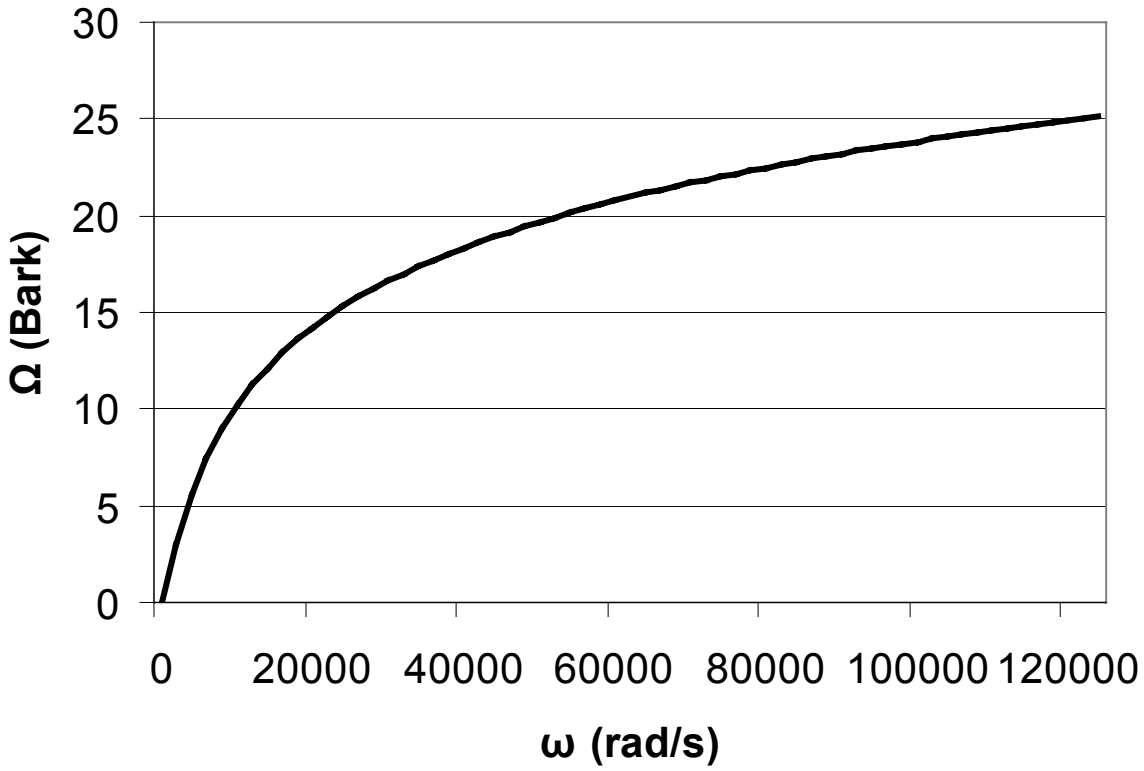

Figura 4 – Curva Bark versus rad/s

A segunda etapa consiste em fazer uma pré-ênfase através da curva de mesma audibilidade igual, que é uma aproximação da sensibilidade auditiva humana para diferentes freqüências. Neste front-end, a curva pode ser aproximada pela expressão

$$E(\omega) = \frac{\left(\left(\omega^2 + 56.8 \times 10^6\right)\omega^4\right)}{\left(\omega^2 + 6.3 \times 10^6\right)^2 \times \left(\omega^2 + 0.38 \times 10^9\right)} \qquad (1)$$

onde E é a saída do filtro de pré-ênfase e ω (igual a $2 \times \pi \times f$, onde f é a frequência em Hz) é a frequência angular em rad/s.

Esta expressão foi proposta por Makhoul; Cosell (1976) e pode ser vista com detalhes na Figura 6. Esta expressão é uma boa aproximação para freqüências até 5000 Hz. Para

freqüências acima de 5 kHz, a equação inclui um termo que aproxima a queda de sensibilidade da audição.

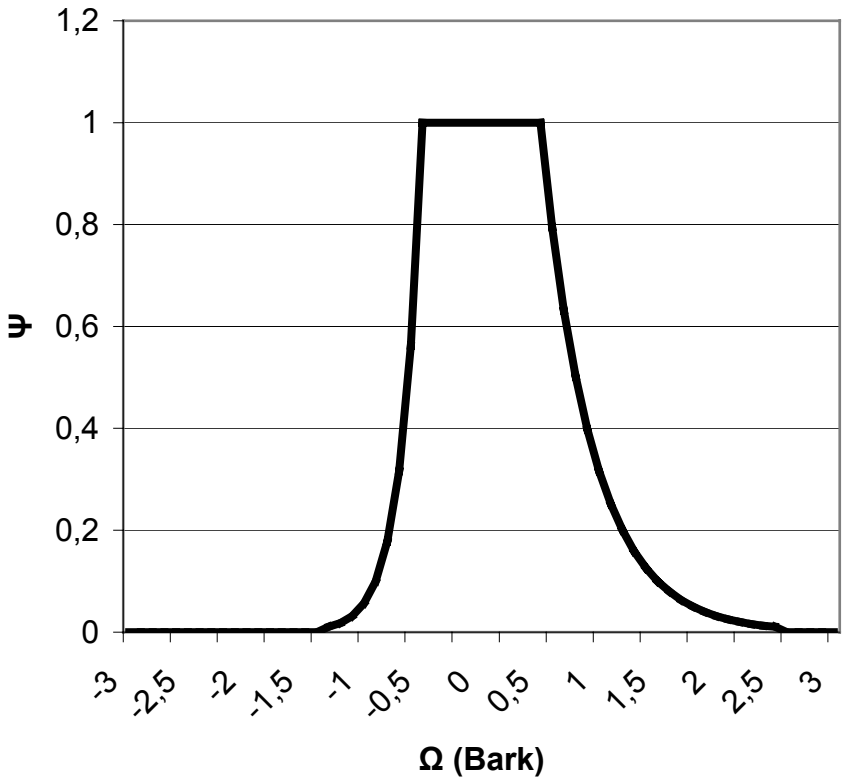

Figura 5 – Máscara de banda crítica

A última etapa de pré-processamento é a compressão de amplitude em forma de raiz cúbica:

$$\Phi(\Omega) = \Xi(\Omega)^{1/3} \qquad (2)$$

onde Ξ é a saída do bloco anterior e Φ é a saída do bloco de compressão, ambas em função da frequência Ω.

Esta expressão simula a relação não-linear entre a intensidade do som e a sua audibilidade percebida.

Depois da etapa de pré-processamento, o PLP consiste em uma modelagem *all-pole* usando o método da autocorrelação, gerando os coeficientes PLP.

Além desses coeficientes, chamados de coeficientes estáticos, são introduzidas também a suas primeira e a segunda derivadas para tentar caracterizar as variações dinâmicas, típicas em consoantes como as oclusivas, que segundo Rabiner; Juang (1993), são "sons **transientes,** produzidos pela formação de uma constrição total em algum lugar no trato oral e pela **súbita liberação** desse ar que estava preso, logo em seguida". Vários estudos citam melhorias devido à introdução das derivadas (também chamadas de coeficientes dinâmicos ou deltas) (Andreão, 2001).

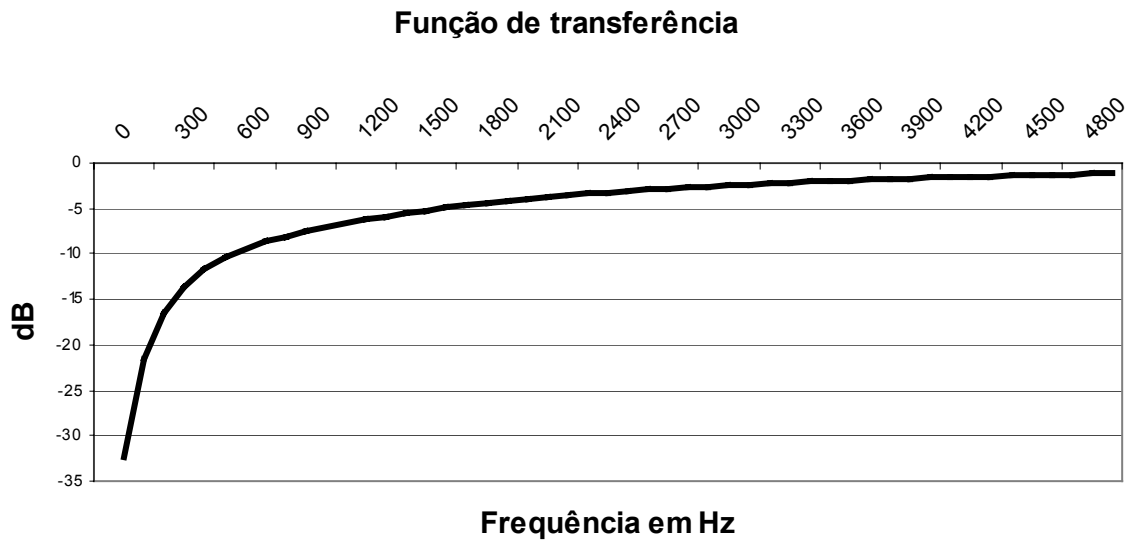

Figura 6 – Curva de pré-ênfase de Makhoul e Cosell

2.4 Modelo Auditivo de Seneff

O quarto conjunto de parâmetros utilizado é o relacionado ao Modelo Auditivo de Seneff (Seneff, 1984). Seu objetivo é extrair os parâmetros essenciais da cóclea em resposta a ondas de pressão sonora. A Figura 7 mostra o diagrama de blocos deste método.

Este modelo é composto de três estágios: um primeiro estágio de pré-processamento, onde o sinal é amostrado a 16 kHz, e passa por uma pré-filtragem para eliminar as componentes de freqüência muito altas e muito baixas. Esta pré-filtragem é implementada com um filtro com quatro pares de zeros complexos. Depois, o sinal é passado por um banco de filtros, composto por 40 filtros, que simulam a resposta da membrana basilar. A resposta

deste banco de filtros pode ser vista na Figura 8. A largura de banda de cada canal é de aproximadamente 0,5 Bark, que corresponde à largura de uma banda crítica.

O segundo estágio é conhecido como modelo de sinapse da célula capilar. É não-linear e tem como objetivo capturar parâmetros proeminentes de transformação de vibrações da membrana basilar, representadas pelas saídas do banco de filtros, a propriedades de respostas probabilísticas dos nervos auditivos. As saídas deste estágio representam a probabilidade de disparo como uma função do tempo para um conjunto de feixes de nervos similares agindo como um grupo. Quatro diferentes mecanismos neurais são modelados neste estágio não-linear.

Um retificador de meia-onda é aplicado ao sinal com o objetivo de simular a sensibilidade direcional presente nas células capilares. Este retificador é o primeiro componente deste estágio e é implementado através de uma não-linearidade do tipo saturação.

A taxa de descarga instantânea das fibras de nervos auditivos é quase sempre significantemente mais alta durante a primeira parte do estímulo acústico e diminui, logo em seguida, até que alcança um estado estacionário. O módulo de adaptação do transitório controla a dinâmica desta resposta a sinais transitórios (devido à liberação do neurotransmissor na região sináptica entre a célula capilar interna e as fibras de nervos conectadas).

A terceira unidade implementa a perda de sincronismo gradual observada no comportamento dos feixes de nervos, à medida que a freqüência do estímulo aumenta, e é implementada por um simples filtro passa-baixas.

A última unidade é chamada Adaptação Rápida e implementa o decaimento inicial muito rápido na taxa de descarga dos feixes de nervos auditivos ocorrendo imediatamente depois do início do estímulo, seguido pelo decaimento mais lento, devido à adaptação do transitório, levando a um estado estacionário. Este módulo utiliza um “Controle Automático de Ganho” na sua implementação.

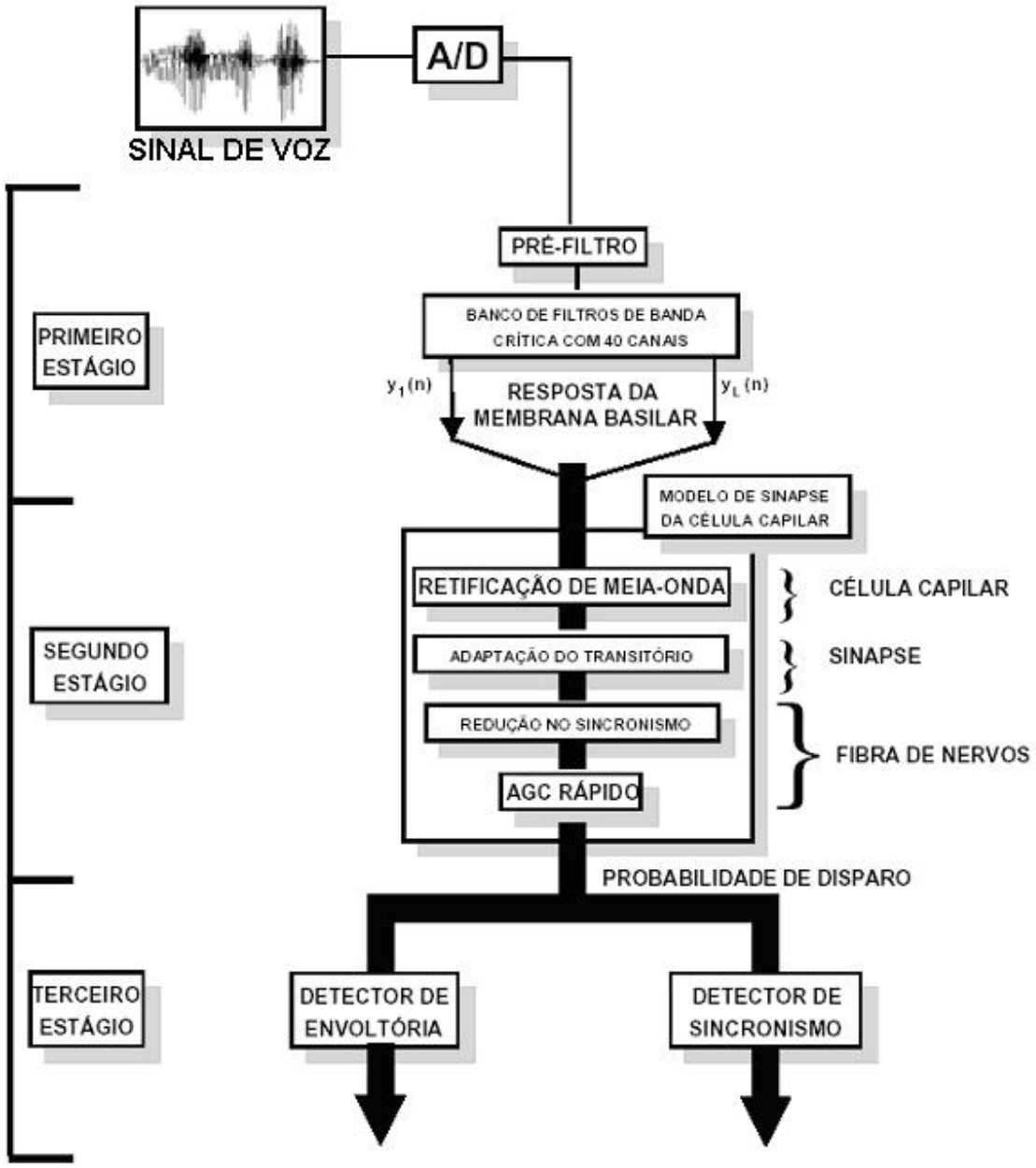

Figura 7 – Diagrama de blocos do Modelo Auditivo de Seneff
Fonte: Cosi (1993)

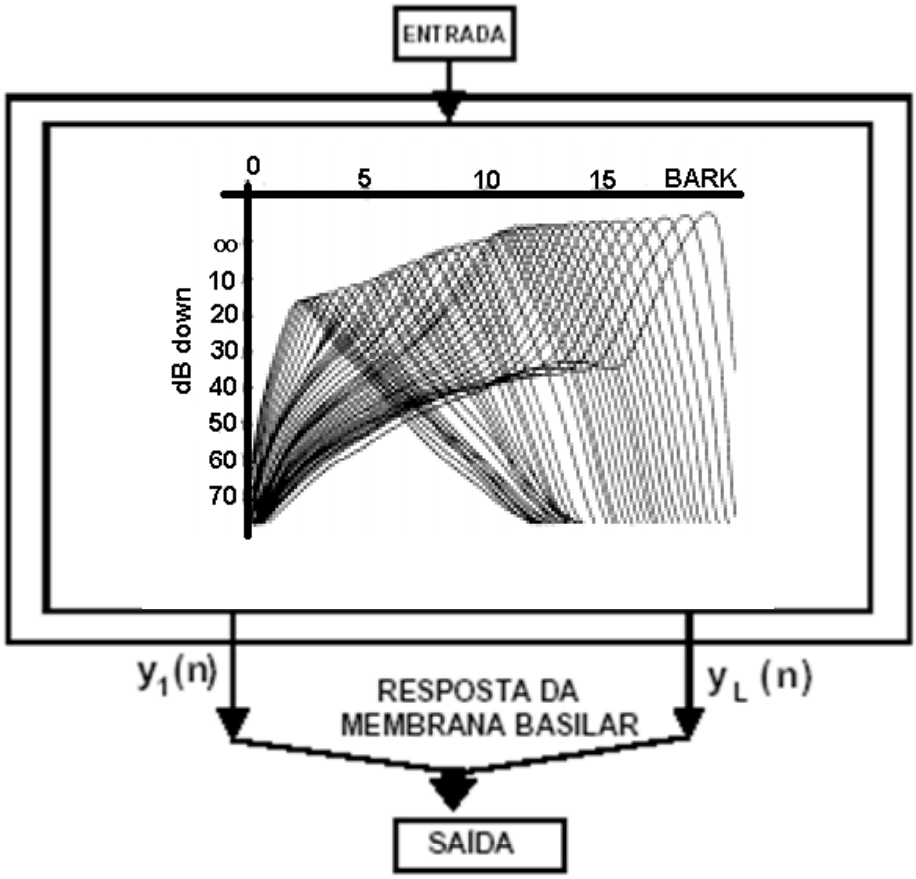

Figura 8 – Resposta em frequência do banco de filtros
Fonte: Cosi (1993)

O último estágio é constituído de dois blocos, que funcionam em paralelo. O primeiro deles, conhecido como Detector de Sincronismo (SD), implementa um PLL característico dos feixes de nervos. Este bloco tem como função ressaltar os picos espectrais devido às ressonâncias do trato vocal. O segundo é conhecido como Detector de Envoltória (ED) e tem como objetivo calcular a envoltória dos sinais na saída do estágio anterior, concentrando-se na variação dinâmica do sinal, detectando assim, mais facilmente, os sons transientes. É implementado através de um filtro-passa-baixas, suavizando e sub-amostrando as saídas do segundo estágio. Segundo Seneff (1988), este bloco é o mais indicado para localizar as transições entre fonemas, provendo uma base adequada para segmentação fonética. Maiores detalhes de implementação podem ser encontrados em Cosi (1993).

2.5 *Mel-Frequency Cepstrum Coefficients* (MFCC)

O cálculo dos parâmetros MFCC envolve basicamente duas etapas: pré-processamento e cálculo dos coeficientes. A Figura 9 mostra os blocos que compõem essas duas etapas.

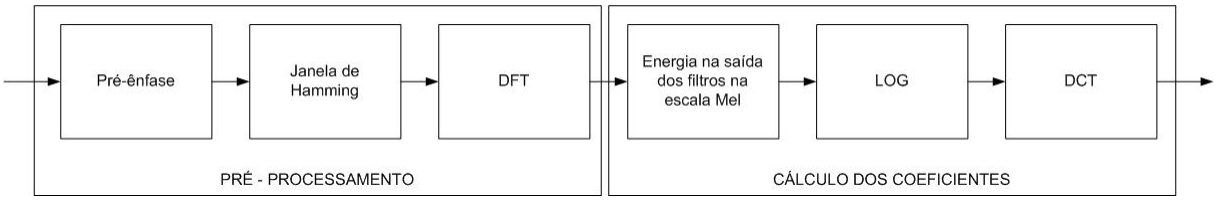

Figura 9 – Diagrama de blocos do *front-end* MFCC

O pré-processamento envolve as etapas de pré-ênfase, janelamento (janela de Hamming) e cálculo da DFT, já conhecidos pela utilização em LPC.

O cálculo dos coeficientes significa passar o sinal de voz por um banco de filtros distribuídos na freqüência, mas na escala Mel, diferente dos coeficientes LFCC, onde a escala de freqüências é linear. A fórmula de conversão da escala Mel é

$$MEL = 2595 \log_{10}\left(1 + \frac{f}{700}\right) \qquad (3)$$

onde f é a frequência original em Hz e MEL é o valor da nova frequência na escala Mel.

A curva de conversão de Hertz para Mel pode ser vista na Figura 10. Segundo Zbynik; Psutka (1999), o MFCC tem, como vantagem, a escala de freqüências corrigida segundo a escala Mel, que tenta se adequar à modelagem do sistema auditivo.

Neste caso, são 20 filtros, que podem ser observados na Figura 11. Os dez primeiros filtros são espaçados uniformemente até o valor de 1 kHz. Acima de 1 kHz, as faixas estão distribuídas segundo a escala Mel.

Calcula-se a energia na saída de cada filtro e aplica-se a função log, proveniente da teoria de Processamento Homomórfico, necessária para separar o trato vocal da excitação. Como o banco de filtros já eliminou a excitação, o único objetivo aqui é de suavizar o

espectro (Zbynik; Psutka, 1999), resultando no coeficiente que Davis; Mermelstein (1980) denominam de X_k, que é aplicado em

$$MFCC_i = \sum_{k=1}^{20} X_k \cos\left[i\left(k - \frac{1}{2} \right) \frac{\pi}{20} \right], \qquad \text{i} = 1,2,...,\text{M} \qquad (4)$$

que nada mais é que a fórmula da DCT, onde M é o número de coeficientes cepstrais (neste trabalho, 12; o décimo terceiro coeficiente seria a energia) e k = 1,2,...,20, especifica qual o filtro que está sendo utilizado.

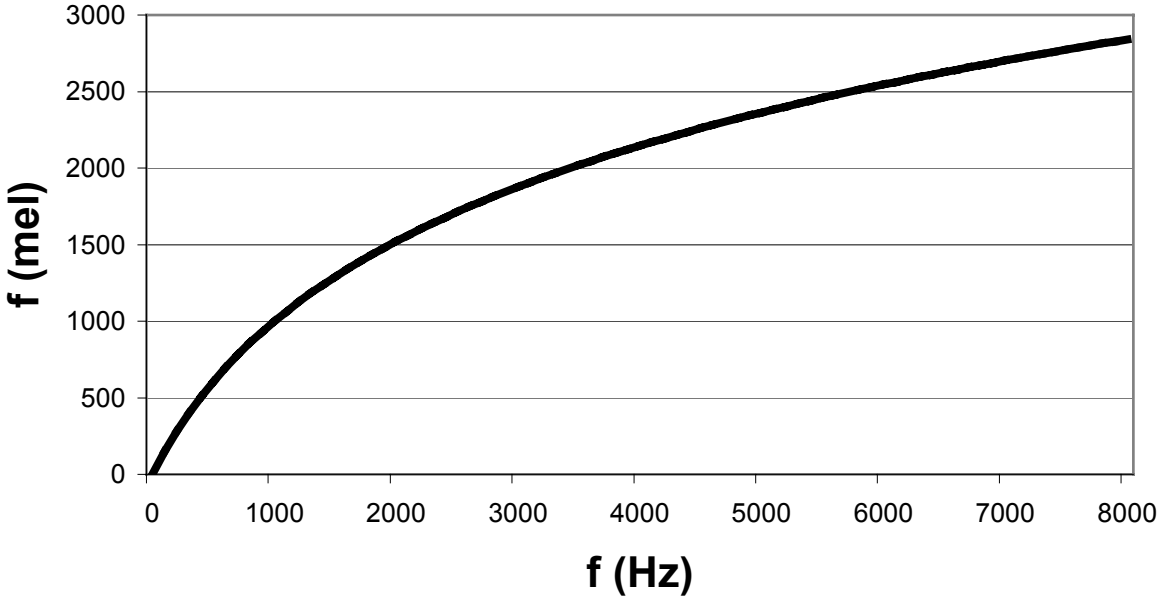

Figura 10 – Curva de conversão Hz vs Mel

De forma semelhante à técnica PLP, são introduzidos os coeficientes dinâmicos, totalizando assim 39 coeficientes (12 estáticos, mais a energia, e 26 dinâmicos).

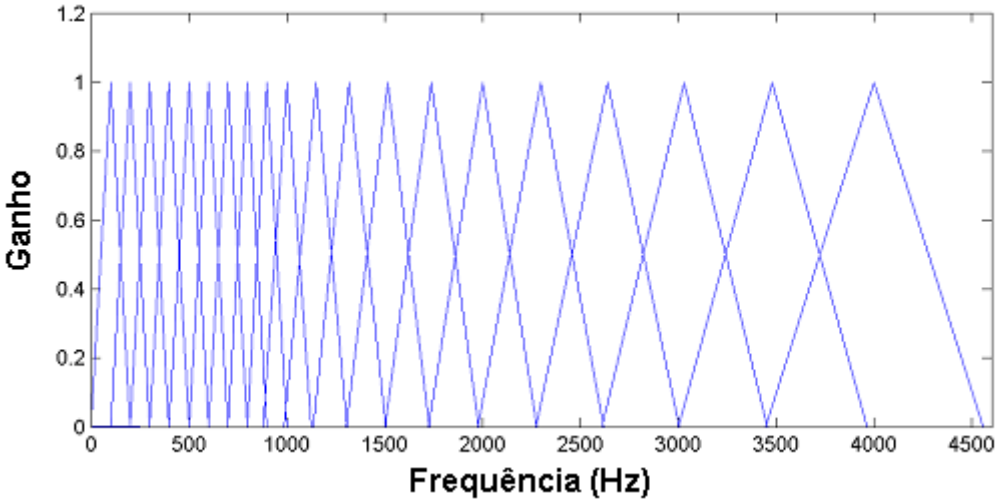

Figura 11 – Banco de filtros utilizado em MFCC

2.6 RASTA

A sigla RASTA vem das palavras ***Rela****tive* ***Spectra****l* (Espectro Relativo). Nada mais é que uma modificação da técnica PLP (Hermansky; Morgan, 1994). Seguindo os passos da técnica PLP, descritos na Seção 2.3, depois da primeira etapa (processamento de banda crítica), esta técnica insere três novas etapas, antes das próximas etapas do PLP.

As três novas etapas consistem em: 1) Fazer uma transformação na amplitude, através de uma técnica de compressão, como por exemplo, o logaritmo, 2) Filtrar o sinal através de um filtro IIR, com a função de transferência:

$$H(z) = 0.1z^4 \frac{2 + z^{-1} - z^{-3} - 2z^{-4}}{1 - 0.98z^{-1}} \quad (5)$$

e 3) Realizar a transformação inversa da operação 1), que seria neste o caso o antilogaritmo.

Como se trata de uma pequena modificação do PLP, gera a mesma quantidade de parâmetros, que no caso serão 39 parâmetros a cada quadro. O diagrama de blocos do sistema pode ser visto na Figura 12.

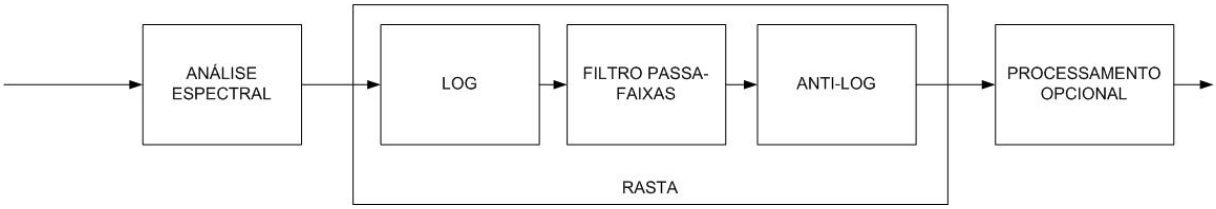

Figura 12 – Diagrama de blocos do *front-end* RASTA

O RASTA surgiu para tentar eliminar a dependência do reconhecimento com variações de freqüência decorrentes de alterações no canal ou alterações de microfones, isto é, variações constantes ou muito lentas. É uma tentativa de simular uma característica humana de não prestar muita atenção a mudanças lentas nas características de freqüência do ambiente de comunicações ou de não se deixar influenciar com ruído de fundo (Hermansky; Morgan, 1994).

Assim, a expressão (5) corresponde a um filtro IIR passa-faixas. Como a taxa de mudança das componentes não-linguísticas na voz quase sempre reside fora da taxa típica de mudança do formato do trato vocal, este filtro elimina as componentes espectrais que variam mais lentamente ou mais rapidamente que a faixa típica de variação da voz.

A mudança de domínio da filtragem (log) é necessária para que as distorções resultantes de mudança de canal ou de microfone apareçam como uma constante aditiva no espectro. Logicamente, após a filtragem é necessário voltar ao domínio anterior, utilizando uma função exponencial (operação inversa do log).

2.7 Banco de Filtros (*Filter Bank*)

O último *front-end* utilizado é conhecido como Banco de filtros e consiste em passar o sinal de voz por um banco de filtros semelhante ao descrito na seção 2.5, com a diferença de que são utilizados 24 filtros espaçados de acordo com a escala mel, em vez de 20, como utilizados no MFCC. A escolha do número de 24 está relacionada diretamente com os experimentos de banda crítica (Moore; Glasberg, 1983). Estes parâmetros correspondem à energia na saída de cada filtro.

Além dos 24 parâmetros gerados pelos filtros, acrescenta-se também a energia normalizada do quadro e a primeira derivada dos 25 parâmetros, totalizando assim 50 parâmetros, no total.

3 Classificação e Seleção de Parâmetros

3.1 Classificação

Conforme mencionado no Capítulo 1, a tarefa de classificação é parte do processo de reconhecimento de padrões. Independente da máquina que está sendo implementada (reconhecimento de voz, impressões digitais ou mesmo classificação de frutas em um processo industrial de fabricação de sucos), a tarefa de classificação é considerada um ponto chave do reconhecimento de padrões. A tarefa de classificar um objeto em um pré-especificado conjunto de categorias ou classes é uma característica da inteligência humana, que é objeto de estudos da inteligência artificial. É notório que o poder de classificar ajuda a mente humana no processo de aprendizado. Classificando, os homens podem se especializar em uma classe, ou estudar uma classe de cada vez. Ou até mesmo, podem tomar decisões. Sabendo a classe de um determinado objeto, pode-se escolher que o que fazer com aquele objeto.

Para distinguir um objeto de outro, normalmente, o homem armazena parâmetros (atributos) marcantes (característicos) de uma determinada classe. Por exemplo, uma laranja é composta de gomos. Se alguém precisa verificar se é uma laranja, imediatamente verifica se a fruta tem gomos. Esse é um parâmetro marcante da laranja. É óbvio que somente este parâmetro não é suficiente para afirmar que a fruta é uma laranja, mas é um parâmetro relevante.

Existem duas formas de se treinar um classificador: o aprendizado supervisionado e o aprendizado não-supervisionado. No primeiro, é fornecido um rótulo para cada classe durante a fase de treinamento, para ajudar o classificador a distinguir as características de cada classe. No caso das frutas, é apresentada uma fruta e é informada a sua classificação (se é laranja, uva, etc.), durante a fase de treinamento. No segundo, apenas os exemplos de treinamento são disponibilizados ao classificador. De acordo com o exemplo anterior, não se informa que tipo de fruta existe na fase de treinamento. Assim, ele forma agrupamentos naturais, chamados *clusters*, dos padrões de entrada (Duda; Hart, 2000). Estes *clusters* podem ser considerados como classes que o próprio classificador identificou. É óbvio que neste caso, se houver imprecisões nos dados de entrada, o desempenho da classificação será prejudicado.

Para se implementar uma tarefa de classificação, torna-se necessária uma base de dados de exemplos, para possibilitar o treinamento do classificador e cada exemplo é caracterizado por uma série de parâmetros que descrevem suas características.

A seguir, será descrito o SVM, um dos classificadores utilizados neste trabalho.

3.2 SVM (*Support Vector Machines*)

SVM é uma técnica que pode ser usada para classificação de padrões e regressão linear, proposta por Vapnik em Boser et al (1992). Ela se tornou famosa quando, usando mapas de *pixels* como entrada, obteve resultados competitivos em tarefas de reconhecimento de palavras escritas à mão (Cortes; Vapnik,1995).

SVM´s tentam resolver um problema de classificação binária, que pode ser definido como:

1 - Dado um conjunto de dados D, composto por N amostras, $<x_1, y_1>$, $<x_2, y_2>$, ...,$<x_N, y_N>$. Cada amostra é composta de um conjunto de exemplos de treinamento x_i de comprimento M, com os elementos $x_i = <x_1, x_2, ...,x_M>$ e um identificador da classe $y_i \in \{-1, +1\}$.

2 – O objetivo é encontrar um classificador com uma função de decisão $f(x)$ tal que $f(x_i) = y_i$, $\forall <x_i, y_i> \in D$.

Considerando um problema binário (classificar em uma de duas classes possíveis), o objetivo é separar as duas classes por uma função que é deduzida dos exemplos disponíveis. Pode ser observado na Figura 13 que existem várias funções que conseguem separar as duas classes, mas só existe uma que maximiza a margem (distância entre a função e o ponto mais próximo).

O desempenho desse classificador pode ser medido em termos do erro de classificação definido na equação

$$erro(f(\mathbf{x}), y) = \begin{cases} 0 & \text{se } f(\mathbf{x}) = y \\ 1 & \text{para outros casos} \end{cases} \quad (6)$$

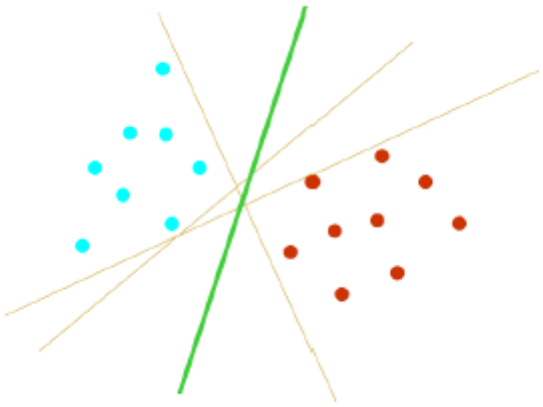

Figura 13 – Funções separando duas classes em problema binário
Fonte: Gunn (1998)

3.2.1 Caso Separável de SVM

Existem dois casos a se considerar em termos de classificadores lineares: o caso onde a separação entre as duas classes é alcançada de forma perfeita e a outra, onde este requisito não é satisfeito. Neste trabalho, será considerado o primeiro caso.

Na Figura 14, pode-se observar um arranjo de pontos que pode ser utilizado em um problema de classificação binário. Os círculos representam os exemplos positivos (y_i = +1), enquanto que os quadrados representam os exemplos negativos (y_i = -1).

Um mapeamento possível que pode separar as duas classes é:

$$f(\mathbf{x}, y) = sign(\mathbf{w}.\mathbf{x} + b) \qquad (7)$$

onde **w** é um vetor de ponderação e b é a distância à origem.

Dado este mapeamento, o hiperplano

$$\mathbf{w}.\mathbf{x} + b = 0 \qquad (8)$$

define o limite de decisão entre a classe positiva e a classe negativa. Os dois conjuntos de dados são chamados de linearmente separáveis pelo hiperplano se puder ser escolhido um par $\{w, b\}$ tal que o mapeamento na equação (8) seja perfeito.

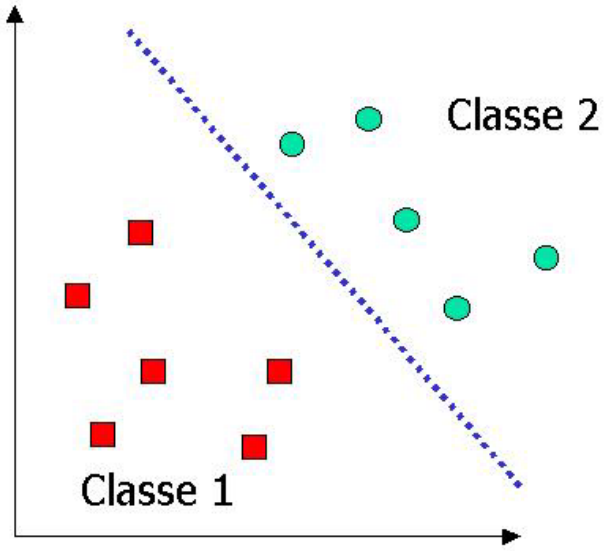

Figura 14 – Exemplo de arranjo de pontos, utilizado em classificação binária

Várias linhas podem ser traçadas entre os dois conjuntos de pontos que podem separar com facilidade os dois grupos. A questão aqui é encontrar qual a melhor fronteira de decisão. O classificador SVM encontra o único hiperplano que maximiza a margem entre os dois conjuntos. Por esse motivo é que o SVM é descrito como um classificador de margem máxima. Nesse caso, o conjunto de pontos é chamado de otimamente separado. Na Figura 15, pode ser percebida a definição de margem, representada pela variável *m*. Pode ser facilmente percebido que maximizar a margem significa diminuir o erro esperado.

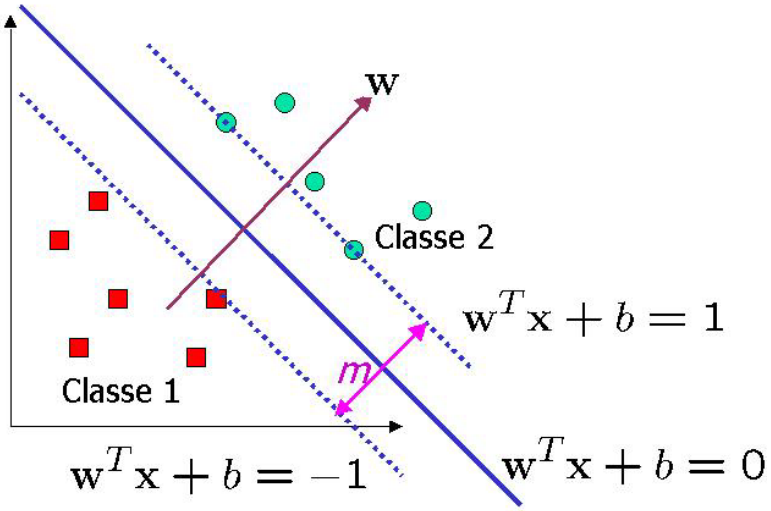

Figura 15 – Definição de Margem em SVM

Este trabalho não tem como objetivo detalhar a teoria de SVM, mas utilizá-la como ferramenta classificadora, juntamente com a técnica de seleção de parâmetros. É importante frisar que as SVM´s são bastante adequadas quando os *front-end´s* são heterogêneos (Klautau et al, 2004).

Considerando que o objetivo é classificar qual o fonema pronunciado, cada fonema se torna uma classe. Pode-se então criar uma SVM que informe se aquele trecho de voz é um fonema de uma classe *A*, por exemplo, ou de uma classe *B*. Então, um problema que se mostrava como sendo de classificação entre várias classes possíveis, se torna um problema de classificação para duas classes. Logicamente, pode-se perceber que, neste caso, tornam-se necessárias várias SVM´s de duas classes, para que se possa compor a solução do problema. Na verdade, este esquema é conhecido como *Error-Correcting Output Code* (ECOC) com a matriz *all-pairs* (todos-os-pares) e decodificação de Hamming (Klautau et al, 2004), sendo este um possível arranjo de ECOC.

Neste método, treina-se uma SVM para cada par de fones. Assim, o número de SVM´s treinadas é a combinação dois a dois de *K* fones, onde *K* é a quantidade total de fones possíveis. Para escolher a classe correta, cada classificador apostará em uma das *K* classes (fones). Aquela classe que obtiver a maior quantidade de "votos" é a classe escolhida.

3.3 Seleção de Parâmetros

Hoje em dia, existem diversas formas de coletar dados para vários sistemas. A grande questão é: como analisá-los. Da mesma forma, um sinal de voz pode ser transformado em um conjunto de parâmetros através de diversos tipos de *front-end´s*. Esta grande quantidade de *front-end´s* fornece uma quantidade equivalente de parâmetros que podem aumentar a quantidade de informação relacionada à análise, mas também pode aumentar em muito o custo computacional. A técnica conhecida como Seleção de Parâmetros identifica os parâmetros mais relevantes em um determinado exemplo, eliminando dados redundantes (com isso, reduzindo o custo computacional, acelerando o processo) e, em alguns casos, até melhorando o desempenho de algoritmos de aprendizado. Hall (1999) identifica exemplos de classificadores extremamente sensíveis a parâmetros irrelevantes ou imprecisos.

Existem duas grandes classes de métodos de seleção de parâmetros: os filtros e os *wrappers*. Os filtros calculam o valor dos parâmetros usando heurísticas baseadas nas características gerais dos dados, enquanto que os *wrappers* calculam o valor dos parâmetros usando o algoritmo de aprendizado que é aplicado aos dados. Os filtros implementam um

ranking para cada parâmetro, eliminando (*backward elimination*) ou acrescentando (*forward selection*) esse parâmetro no conjunto. Os *wrappers* utilizam o algoritmo classificador de interesse como uma caixa preta para pontuar subconjuntos de parâmetros de acordo com o seu poder de predição. Os *wrappers* conseguem resultados melhores que os filtros porque eles são otimizados para o algoritmo de aprendizado utilizado. A grande vantagem dos filtros é que eles conseguem uma generalização inerente ao próprio método, enquanto que os *wrappers* precisam ser refeitos quando o algoritmo de treinamento é substituído por um outro qualquer. Além disso, são mais rápidos que os *wrappers* (Hall, 1999). Pelas vantagens demonstradas, neste trabalho foi utilizado um filtro para seleção de parâmetros, que será descrito na próxima seção.

3.4 *Boosting*

Em um classificador, é muito comum encontrar um parâmetro (ou um conjunto de parâmetros) que gera um resultado razoável em termos de taxa de acertos – pelo menos, melhor que jogar uma moeda. Um classificador alimentado por estes parâmetros é considerado uma hipótese fraca. *Boosting* é uma técnica utilizada para obter uma hipótese dita "forte" a partir de várias hipóteses fracas.

O algoritmo chama o classificador fraco repetidamente, sendo que em cada iteração, ele concentra-se nos exemplos de treinamento classificados erradamente. Cada vez que ele é chamado, o classificador gera uma nova regra de classificação, sendo que o classificador forte é o resultado da somatória ponderada desses classificadores fracos (Schapire, 2002).

Pode ser observada a Figura 16, citada em Meir; Ratsch (2003), onde são mostrados, no eixo horizontal, um parâmetro e no eixo vertical outro parâmetro, um exemplo de *Boosting*. Os exemplos azuis são considerados de uma classe e os vermelhos são de outra classe. A linha verde mostra a linha de decisão do classificador combinado. Na primeira iteração, pode ser observado que existem alguns exemplos classificados erradamente. Na segunda iteração, o diâmetro desses exemplos foi aumentado (o diâmetro do círculo representa o peso desse exemplo no treinamento). O algoritmo tenta se concentrar nestes exemplos, que foram classificados erradamente na primeira iteração. Pode-se observar que na segunda iteração a curva foi corrigida para considerar estes exemplos.

O algoritmo conhecido como *AdaBoost* foi introduzido em 1995 por Freund; Schapire (1995). Ele tem como entrada um conjunto de exemplos de treinamento $(x_1, y_1),...., (x_m, y_m)$

onde cada x_i pertence a algum domínio ou espaço de exemplos X, e cada rótulo y_i está em algum conjunto de rótulos Y.

Neste trabalho, considerou-se $Y = \{-1, +1\}$, um problema de classificação binário. Mas, isto não chega a ser um problema, pois, como já explicado, cada SVM vai trabalhar com duas classes apenas. O *AdaBoost* chama um classificador fraco repetidamente em uma série de rodadas $t = 1, ...,T$. Uma das principais idéias do algoritmo é manter uma distribuição ou conjunto de pesos sobre o conjunto de treinamento. O peso desta distribuição no exemplo de treinamento i na rodada t é denotado $D_t\,(i)$.

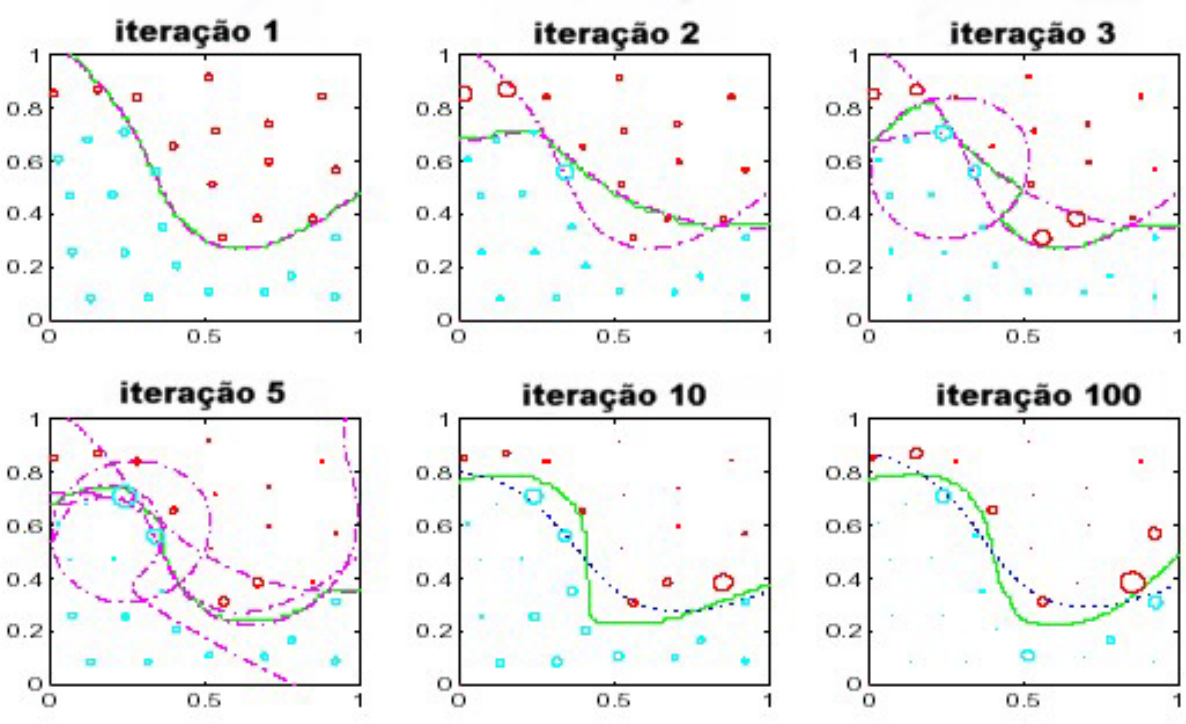

Figura 16 – Exemplo de aplicação do algoritmo Boosting
Fonte: Meir e Ratsch (2003)

Inicialmente, todos os pesos são considerados iguais ($D_1(i) = 1/m$), mas em cada rodada, os pesos dos exemplos classificados incorretamente são aumentados de forma que o classificador base é forçado a se focar nos exemplos difíceis. A alteração dos pesos é feita de acordo com a fórmula abaixo:

$$D_{t+1}(i) = \frac{D_t(i)\exp(\alpha_t y_i h_t(x_i))}{Z_t} \qquad (9)$$

onde Z_t é um fator de normalização para garantir que a soma dos pesos seja igual a 1. Além disso, α_t é um parâmetro que intuitivamente mede a importância dedicada ao classificador h_t. Tipicamente,

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right). \quad (10)$$

onde ε_t é o erro calculado quando o classificador fraco é alimentado com os m exemplos.

Considerando que α é sempre positivo, caso a predição $h_t(x_i)$ seja correta, o peso $D_t(i)$ é diminuído, já que y_i e $h_t(x_i)$ teriam o mesmo sinal e o argumento da exponencial seria negativo. Caso contrário, em caso de erro, o argumento seria positivo. Isso aumentaria o peso D_t (i).

O classificador combinado H é definido como

$$H(x) = sign\left(\sum_{t=1}^{T}\alpha_t h_t(x)\right). \quad (11)$$

A figura 17 lista o pseudo-código do algoritmo AdaBoost.

Pode-se provar que o erro combinado tende a diminuir com o passar das iterações. Deve ser frisado que o cálculo do erro em cada etapa é encontrado, passando o próprio conjunto de treino através do classificador encontrado. Considerando todos os eventos equiprováveis, de (9), vem

$$D_2(i) = \frac{1}{m}\frac{\exp(-\alpha_1 y_i h_1(x_i))}{Z_1} \quad (12)$$

$$D_3(i) = D_2(i)\frac{\exp(-\alpha_2 y_i h_2(x_i))}{Z_2} = \frac{1}{m}\frac{\exp(-\alpha_1 y_i h_1(x_i))}{Z_1}\frac{\exp(-\alpha_2 y_i h_2(x_i))}{Z_2} \quad (13)$$

Generalizando a expressão, encontra-se

Dado: $(x_1, y_1), \ldots, (x_m, y_m)$ onde $x_i \in X, y_i \in Y = \{-1, +1\}$

Inicializar: $D_1(i) = 1/m$.

Para $t = 1, \ldots, T$:

- Treinar a hipótese fraca utilizando a distribuição D_t;
- Utilizar a hipótese fraca $h_t : X \rightarrow \{-1, +1\}$ com erro

$$\varepsilon_t = \Pr_{i \sim D_t}\left[h_t(x_i) \neq y_i\right]$$

- Calcular

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$$

- Atualizar os pesos:

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} \text{ se } h_t(x_i) = y_i \\ e^{\alpha_t} \text{ se } h_t(x_i) \neq y_i \end{cases}$$

$$= \frac{D_t(i) e^{-\alpha_t y_i h_t(x_i)}}{Z_t}$$

onde Z_t é um fator de normalização (escolhido de forma tal que D_{t+1} seja uma distribuição de probabilidades)

Saída do classificador final:

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

Figura 17 – Pseudo-código do algoritmo *AdaBoost*

$$D_{t+1}(i) = \frac{1}{m}\frac{\prod_t \exp(-\alpha_t y_i h_t(x_i))}{\prod_t Z_t} = \frac{1}{m}\frac{\exp\left(\sum_t -\alpha_t y_i h_t(x_i)\right)}{\prod_t Z_t} =$$

$$= \frac{1}{m}\frac{\exp\left(-y_i \sum_t \alpha_t h_t(x_i)\right)}{\prod_t Z_t} = \frac{1}{m}\frac{\exp(-y_i f(x_i))}{\prod_t Z_t} \quad (14)$$

onde

$$f(x_i) = \sum_t \alpha_t h_t(x_i) \quad (15)$$

Se $H(x_i) \neq y_i$ então $y_i f(x_i) \leq 0$, o que implica que $\exp(-y_i f(x_i)) \geq 1$. Por conseguinte,

$$[H(x_i) \neq y_i] \leq e^{-y_i f(x_i)} \quad (16)$$

Fazendo a somatória em i em ambos os lados e multiplicando por $1/m$, vem

$$\frac{1}{m}\sum_i [H(x_i) \neq y_i] \leq \frac{1}{m}\sum_i \exp(-y_i f(x_i)) \quad (17)$$

Mas, de (14), pode-se extrair

$$\exp(-y_i f(x_i)) = m D_{t+1}(i) \prod_t Z_t \quad (18)$$

Fazendo a somátoria em i nos dois lados da expressão, encontra-se

$$\sum_i \exp(-y_i f(x_i)) = \sum_i \left(m D_{t+1}(i) \prod_t Z_t \right) \quad (19)$$

As variáveis m e ΠZ_t podem ser colocadas para fora da somatória, resultando em

$$\sum_i \exp(-y_i f(x_i)) = m \prod_t Z_t \sum_i (D_{t+1}(i)) \tag{20}$$

Mas, considerando que

$$\sum_i (D_{t+1}(i)) = 1 \tag{21}$$

Então,

$$\sum_i \exp(-y_i f(x_i)) = m \prod_t Z_t \tag{22}$$

Substituindo (22) em (17), encontra-se

$$\frac{1}{m} \sum_i [H(x_i) \neq y_i] \leq \frac{1}{m} \sum_i \exp(-y_i f(x_i)) = \prod_t Z_t \tag{23}$$

Percebe-se em (23) que o erro de treinamento pode ser reduzido de forma mais rápida se se escolhe α_t e h_t em cada rodada para minimizar

$$Z_t = \sum_i D_t(i) \exp(-\alpha_t y_i h_t(x_i)) \tag{24}$$

Considerando classificadores binários (classes –1 e +1), então $y_i h_t(x_i)$ será igual a 1 se o classificador acertar a classe e será igual a –1 se errar a classe. Então Z_t pode ser redefinido como

$$Z_t = (1 - \varepsilon_t) \exp(-\alpha_t) + \varepsilon_t \exp(\alpha_t) \tag{25}$$

Para minimizar Z_t. deriva-se a parte direita da expressão e iguala-se a 0. Assim,

$$\frac{dZ_t}{d\alpha_t} = -(1-\varepsilon_t)\exp(-\alpha_t) + \varepsilon_t \exp(\alpha_t) = 0 \tag{26}$$

Eliminando os parênteses,

$$-\exp(-\alpha_t) + \varepsilon_t \exp(-\alpha_t) + \varepsilon_t \exp(\alpha_t) = 0 \tag{27}$$

Multiplicando a expressão por exp ($2\alpha_t$)

$$-1 + \varepsilon_t + \varepsilon_t \exp(2\alpha_t) = 0 \tag{28}$$

Isolando exp ($2\alpha_t$)

$$\exp(2\alpha_t) = \frac{1-\varepsilon_t}{\varepsilon_t} \tag{29}$$

Aplicando o logaritmo neperiano dos dois lados e isolando α_t

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) \tag{30}$$

que corresponde à equação (10).

Substituindo (30) em (25), encontra-se

$$Z_t = (1-\varepsilon_t)\exp\left(-\frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)\right) + \varepsilon_t \exp\left(\frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)\right) =$$

$$= (1-\varepsilon_t)\exp\left(\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)^{-\frac{1}{2}}\right) + \varepsilon_t \exp\left(\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)^{\frac{1}{2}}\right) \tag{31}$$

Cancelando as funções exp e ln, resulta em

$$Z_t = (1-\varepsilon_t)\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)^{-\frac{1}{2}} + \varepsilon_t \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)^{\frac{1}{2}} = 2\sqrt{\varepsilon_t(1-\varepsilon_t)} \tag{32}$$

Definindo

$$\gamma_t = \frac{1}{2} - \varepsilon_t \tag{33}$$

Então

$$Z_t = \sqrt{4\left(\frac{1}{2}-\gamma_t\right)\left(\frac{1}{2}+\gamma_t\right)} = \sqrt{4\left(\frac{1}{4}-\gamma_t^{\,2}\right)} = \sqrt{1-4\gamma_t^{\,2}} \tag{34}$$

Fazendo o produtório de *t* em ambos os lados

$$\prod_t Z_t = \prod_t \sqrt{1-4\gamma_t^{\,2}} \tag{35}$$

A partir da desigualdade deduzida a partir da Série de Taylor Binomial

$$\sqrt{1+x} \le 1 + \frac{\mathrm{x}}{2} \tag{36}$$

Têm-se

$$\prod_t Z_t = \prod_t \sqrt{1-4\gamma_t^{\,2}} \le \prod_t \left(1-2\gamma_t^{\,2}\right) \tag{37}$$

A partir da Série de Taylor Exponencial

$$\exp(x) \ge 1 + \mathrm{x} \tag{38}$$

Substituindo (38) em (37)

$$\prod_t Z_t = \prod_t \sqrt{1-4\gamma_t^{\,2}} \le \prod_t \exp\left(-2\gamma_t^{\,2}\right) = \exp\left(-2\sum_t \gamma_t^{\,2}\right) \tag{39}$$

Então, se cada classificador base é levemente melhor que o experimento de jogar uma moeda de tal forma que $\gamma_t \geq \gamma$ para algum $\gamma > 0$, então o erro de treinamento cai exponencialmente em T, visto que o limite de (39) é em $\exp(-2T\gamma^2)$.

3.5 Exemplo de Classificador com *Boosting*

Para exemplificar como *Boosting* pode melhorar a taxa de acertos de um classificador, nesta seção será detalhado o passo-a-passo do treinamento de um classificador baseado em classificadores "fracos" do tipo *Naive Bayes* (Duda; Hart, 2000). Este classificador foi escolhido por ser de fácil compreensão.

O classificador *Naive Bayes* considera que o efeito do valor de uma variável em uma dada classe é independente dos valores de outra variável, isto é,

$$P(x_1, x_2 \mid y) = P(x_1 \mid y)P(x_2 \mid y) \tag{40}$$

Isto é feito para simplificar o cálculo e é por esse motivo que este classificador é chamado de *naive* (ingênuo, em português).

Para se determinar qual a classe do evento observado, utiliza-se o Teorema de Bayes:

$$P(X \mid Y) = \frac{P(Y \mid X)P(X)}{P(Y)} \tag{41}$$

que diz que a probabilidade a posteriori é igual a verossimilhança, multiplicada pela probabilidade a priori, dividida pela evidência (Duda; Hart, 2000). Em outras palavras, sabendo a probabilidade a priori de uma classe e conhecendo a probabilidade de um atributo assumir um determinado valor, dada a ocorrência de uma determinada classe (dados extraídos do conjunto de treinamento), calcula-se a probabilidade de um evento ser de uma classe, dada a ocorrência de um determinado valor de atributo.

Os passos para se treinar um classificador *Naive Bayes* são: calcular as probabilidades a priori de cada classe; calcular as verossimilhanças e a partir daí, calcular as probabilidades de cada classe. Para cada exemplo, vence a classe que exibir a maior probabilidade a posteriori.

Por exemplo, considere-se um conjunto de treinamento como o exibido na Tabela 1. Ela mostra nove eventos possíveis observados em uma experiência onde um jogador de futebol chutava a gol de várias distâncias. Existem duas classes possíveis: a bola entra no gol (+1) ou não entra (-1). Nos nove eventos foram observados dois atributos: o primeiro mostra a distância do jogador em relação ao gol (perto, médio ou longe) e o segundo mostra a forma como o jogador chuta a bola (lado esquerdo, centro ou lado direito). A penúltima coluna mostra o número de ocorrências que aparecem no conjunto de treinamento, enquanto que a última mostra a probabilidade de ocorrência de cada evento, calculada dividindo-se o número de ocorrências de cada evento pelo número total de ocorrências que é 80.

Considerando o número de ocorrências, então a probabilidade a priori da classe ser +1 é igual a 0,925, que é igual a soma dos números da última coluna apenas para os eventos 1, 2, 3, 5, 6 e 8 e a probabilidade da classe ser -1 é igual a 0,075, que é igual a soma dos números da última coluna apenas para os eventos 4, 7 e 9.

A seguir, calculam-se as verossimilhanças:

$$\begin{array}{ll} P(A_1 = perto \mid Classe = +1) = 0{,}5838 & P(A_2 = esquerdo \mid Classe = +1) = 0{,}1946 \\ P(A_1 = perto \mid Classe = -1) = 0 & P(A_2 = esquerdo \mid Classe = -1) = 0{,}6667 \\ P(A_1 = médio \mid Classe = +1) = 0{,}2216 & P(A_2 = centro \mid Classe = +1) = 0{,}5838 \\ P(A_1 = médio \mid Classe = -1) = 0{,}3333 & P(A_2 = centro \mid Classe = -1) = 0 \\ P(A_1 = longe \mid Classe = +1) = 0{,}1946 & P(A_2 = direito \mid Classe = +1) = 0{,}2216 \\ P(A_1 = longe \mid Classe = -1) = 0{,}6667 & P(A_2 = direito \mid Classe = -1) = 0{,}3333 \end{array} \quad (42)$$

Aplicando o Teorema de Bayes, desconsiderando o denominador que é igual em todos os casos (probabilidade de ocorrer um determinado valor para um atributo), não influenciando na comparação sobre qual a classe com a maior probabilidade, tem-se

Tabela 1 – Conjunto de treinamento utilizado no Exemplo de *Boosting*

Evento	Atributo 1: Distância	Atributo 2: Região de contato do pé com a bola	CLASSE	Número de ocorrências no conjunto de treino	Probabilidade da ocorrência de cada evento
1	PERTO	ESQUERDO	É GOL (+1)	36	0,18
2	PERTO	CENTRO	É GOL (+1)	36	0,18
3	PERTO	DIREITO	É GOL (+1)	36	0,18
4	MÉDIO	ESQUERDO	NÃO É GOL (-1)	5	0,025
5	MÉDIO	CENTRO	É GOL (+1)	36	0,18
6	MÉDIO	DIREITO	É GOL (+1)	5	0,025
7	LONGE	ESQUERDO	NÃO É GOL (-1)	5	0,025
8	LONGE	CENTRO	É GOL (+1)	36	0,18
9	LONGE	DIREITO	NÃO É GOL (-1)	5	0,025

$$\begin{aligned} &P(Classe=+1 \mid A_1=perto, A_2=esquerdo) = P(A_1=perto \mid Classe=+1) \\ &\times P(A_2=esquerdo \mid Classe=+1) \times P(Classe=+1) = 0{,}5838 \times 0{,}1946 \times 0{,}925 = 0{,}1051 \\ &P(Classe=-1 \mid A_1=perto, A_2=esquerdo) = P(A_1=perto \mid Classe=-1) \\ &\times P(A_2=esquerdo \mid Classe=-1) \times P(Classe=-1) = 0 \times 0{,}6667 \times 0{,}075 = 0 \end{aligned} \quad (43)$$

Observa-se neste caso que, se o atributos A_1 for igual a perto e A_2 for igual a esquerdo, então se decide pela classe +1, pois é a que apresenta a maior probabilidade a posteriori. Repete-se o mesmo processo para as outras oito combinações possíveis de atributos:

$$\begin{aligned} &P(Classe=+1 \mid A_1=perto, A_2=centro)=P(A_1=perto \mid Classe=+1) \\ &\times P(A_2=centro \mid Classe=+1)\times P(Classe=+1)=0{,}5838\times 0{,}5838\times 0{,}925=0{,}3152 \\ &P(Classe=-1 \mid A_1=perto, A_2=centro)=P(A_1=perto \mid Classe=-1) \\ &\times P(A_2=centro \mid Classe=-1)\times P(Classe=-1)=0\times 0\times 0{,}075=0 \end{aligned} \quad (44)$$

$$\begin{aligned} &P(Classe=+1 \mid A_1=perto, A_2=direito)=P(A_1=perto \mid Classe=+1) \\ &\times P(A_2=direito \mid Classe=+1)\times P(Classe=+1)=0{,}5838\times 0{,}2216\times 0{,}925=0{,}1197 \\ &P(Classe=-1 \mid A_1=perto, A_2=direito)=P(A_1=perto \mid Classe=-1) \\ &\times P(A_2=direito \mid Classe=-1)\times P(Classe=-1)=0\times 0{,}3333\times 0{,}075=0 \end{aligned} \quad (45)$$

$$\begin{aligned} &P(Classe=+1 \mid A_1=médio, A_2=esquerdo)=P(A_1=médio \mid Classe=+1) \\ &\times P(A_2=esquerdo \mid Classe=+1)\times P(Classe=+1)=0{,}2216\times 0{,}1946\times 0{,}925=0{,}0399 \\ &P(Classe=-1 \mid A_1=médio, A_2=esquerdo)=P(A_1=médio \mid Classe=-1) \\ &\times P(A_2=esquerdo \mid Classe=-1)\times P(Classe=-1)=0{,}3333\times 0{,}6667\times 0{,}075=0{,}0167 \end{aligned} \quad (46)$$

$$\begin{aligned} &P(Classe=+1 \mid A_1=médio, A_2=centro)=P(A_1=médio \mid Classe=+1) \\ &\times P(A_2=centro \mid Classe=+1)\times P(Classe=+1)=0{,}22216\times 0{,}5838\times 0{,}925=0{,}1197 \\ &P(Classe=-1 \mid A_1=médio, A_2=centro)=P(A_1=médio \mid Classe=-1) \\ &\times P(A_2=centro \mid Classe=-1)\times P(Classe=-1)=0{,}3333\times 0\times 0{,}075=0 \end{aligned} \quad (47)$$

$$\begin{aligned} &P(Classe=+1 \mid A_1=médio, A_2=direito)=P(A_1=médio \mid Classe=+1) \\ &\times P(A_2=direito \mid Classe=+1)\times P(Classe=+1)=0{,}2216\times 0{,}2216\times 0{,}925=0{,}0454 \\ &P(Classe=-1 \mid A_1=médio, A_2=direito)=P(A_1=médio \mid Classe=-1) \\ &\times P(A_2=direito \mid Classe=-1)\times P(Classe=-1)=0{,}3333\times 0{,}3333\times 0{,}075=0{,}0083 \end{aligned} \quad (48)$$

$$\begin{aligned} &P(Classe=+1 \mid A_1=longe, A_2=esquerdo)=P(A_1=longe \mid Classe=+1) \\ &\times P(A_2=esquerdo \mid Classe=+1)\times P(Classe=+1)=0{,}1946\times 0{,}1946\times 0{,}925=0{,}035 \\ &P(Classe=-1 \mid A_1=longe, A_2=esquerdo)=P(A_1=longe \mid Classe=-1) \\ &\times P(A_2=esquerdo \mid Classe=-1)\times P(Classe=-1)=0{,}6667\times 0{,}6667\times 0{,}075=0{,}0333 \end{aligned} \quad (49)$$

$$
\begin{aligned}
&P(Classe = +1 \mid A_1 = longe, A_2 = centro) = P(A_1 = longe \mid Classe = +1) \\
&\times P(A_2 = centro \mid Classe = +1) \times P(Classe = +1) = 0{,}1946 \times 0{,}5838 \times 0{,}925 = 0{,}1051 \\
&P(Classe = -1 \mid A_1 = longe, A_2 = centro) = P(A_1 = longe \mid Classe = -1) \\
&\times P(A_2 = centro \mid Classe = -1) \times P(Classe = -1) = 0{,}6667 \times 0 \times 0{,}075 = 0
\end{aligned}
\qquad (50)
$$

$$
\begin{aligned}
&P(Classe = +1 \mid A_1 = longe, A_2 = direito) = P(A_1 = longe \mid Classe = +1) \\
&\times P(A_2 = direito \mid Classe = +1) \times P(Classe = +1) = 0{,}1946 \times 0{,}2216 \times 0{,}925 = 0{,}0399 \\
&P(Classe = -1 \mid A_1 = longe, A_2 = direito) = P(A_1 = longe \mid Classe = -1) \\
&\times P(A_2 = direito \mid Classe = -1) \times P(Classe = -1) = 0{,}6667 \times 0{,}3333 \times 0{,}075 = 0{,}0167
\end{aligned}
\qquad (51)
$$

Em todas as situações decide-se pela classe +1, pelo fato de apresentar a maior probabilidade a posteriori. A Tabela 2 compara a classificação correta de cada evento com a classificação extraída a partir das probabilidades a posteriori.

Tabela 2 – Comparação entre o conjunto de treinamento e o resultado da primeira iteração de *Boosting*

EVENTO	A_1	A2	CLASSE CORRETA	CLASSE DE ACORDO COM O CLASSIFICADOR NAIVE BAYES
1	PERTO	ESQUERDO	+1	+1
2	PERTO	CENTRO	+1	+1
3	PERTO	DIREITO	+1	+1
4	MÉDIO	ESQUERDO	-1	+1
5	MÉDIO	CENTRO	+1	+1
6	MÉDIO	DIREITO	+1	+1
7	LONGE	ESQUERDO	-1	+1
8	LONGE	CENTRO	+1	+1
9	LONGE	DIREITO	-1	+1

Comparando-se cada evento, percebe-se que este classificador só errou os eventos 4, 7 e 9, isto é, errou 3 × 5 das 200 ocorrências, ou uma taxa de erro de 15/200 (7,5%). Este classificador conseguiu neste caso um resultado razoável. Para melhorar a taxa de erro, será aplicada a técnica *Boosting*.

Considera-se este classificador como sendo a primeira iteração de *Boosting*. Para um erro de 7,5%, calcula-se o valor de α_1:

$$\alpha_1 = 0{,}5 \ln\left(\frac{1-\varepsilon_1}{\varepsilon_1}\right) = 0{,}5 \times \ln\left(\frac{1\text{-}\,0{,}075}{0{,}075}\right) = 1{,}2561 \qquad (52)$$

A partir do valor de α_1, podem ser calculadas as novas probabilidades de cada evento, que serão utilizadas na segunda iteração. Para os eventos 4, 7 e 9, onde a classificação foi errada, o novo peso será

$$D_2 = D_1 \exp\left(-\alpha_1 y h_1(x)\right) = 0{,}025 \times \exp\left(-1{,}2561 \times (-1) \times 1\right) = 0{,}0878 \qquad (53)$$

Percebe-se que o peso destes exemplos foi aumentado, pois na próxima iteração o algoritmo vai se concentrar nestes eventos, que são os mais difíceis. Agora, serão calculados os pesos para os eventos classificados corretamente:

$$D_2 = D_1 \exp\left(-\alpha_1 y h_1(x)\right) = 0{,}025 \times \exp\left(-1{,}2561 \times (1) \times 1\right) = 0{,}0071 \qquad (54)$$

$$D_2 = D_1 \exp\left(-\alpha_1 y h_1(x)\right) = 0{,}18 \times \exp\left(-1{,}2561 \times (1) \times 1\right) = 0{,}0512 \qquad (55)$$

Nestes casos, os pesos foram diminuídos pelo mesmo motivo anterior. Para fazer com que a soma das probabilidades seja 1, estes pesos são normalizados. A Tabela 3 mostra os pesos encontrados e os pesos depois de normalizados.

Tabela 3 – Pesos que serão utilizados na segunda iteração

EVENTO	A_1	A2	CLASSE CORRETA	PESOS	PESOS NORMALIZADOS
1	PERTO	ESQUERDO	+1	0,0512	0,0973
2	PERTO	CENTRO	+1	0,0512	0,0973
3	PERTO	DIREITO	+1	0,0512	0,0973
4	MÉDIO	ESQUERDO	-1	0,0878	0,1667
5	MÉDIO	CENTRO	+1	0,0512	0,0973
6	MÉDIO	DIREITO	+1	0,0071	0,0135
7	LONGE	ESQUERDO	-1	0,0878	0,1667
8	LONGE	CENTRO	+1	0,0512	0,0973
9	LONGE	DIREITO	-1	0,0878	0,1667

Utilizando esta nova tabela, encontram-se as probabilidades a priori. A probabilidade da classe ser +1 é igual a 0,5 e a probabilidade da classe ser –1 é igual a 0,5. A seguir, recalculam-se as verossimilhanças para a segunda iteração:

$$\begin{gathered} P(A_1 = perto \mid Classe = +1) = (0{,}0973 + 0{,}0973 + 0{,}0973)/0{,}5 = 0{,}5838 \\ P(A_1 = perto \mid Classe = -1) = 0 \\ P(A_1 = médio \mid Classe = +1) = (0{,}0973 + 0{,}0135)/0{,}5 = 0{,}2216 \\ P(A_1 = médio \mid Classe = -1) = 0{,}1667/0{,}5 = 0{,}3333 \\ P(A_1 = longe \mid Classe = +1) = 0{,}0973/0{,}5 = 0{,}1946 \\ P(A_1 = longe \mid Classe = -1) = (0{,}1667 + 0{,}1667)/0{,}5 = 0{,}6667 \end{gathered} \tag{56}$$

$$\begin{gathered} P(A_2 = esquerdo \mid Classe = +1) = 0{,}0973/0{,}5 = 0{,}1946 \\ P(A_2 = esquerdo \mid Classe = -1) = (0{,}1667 + 0{,}1667)/0{,}5 = 0{,}6667 \\ P(A_2 = centro \mid Classe = +1) = (0{,}0973 + 0{,}0973 + 0{,}0973)/0{,}5 = 0{,}5838 \\ P(A_2 = centro \mid Classe = -1) = 0 \\ P(A_2 = direito \mid Classe = +1) = (0{,}0973 + 0{,}0135)/0{,}5 = 0{,}2216 \\ P(A_2 = direito \mid Classe = -1) = 0{,}1667/0{,}5 = 0{,}3333 \end{gathered} \tag{57}$$

Aplicando o Teorema de Bayes na segunda iteração, desconsiderando o denominador, tem-se

$$\begin{aligned} &P(Classe=+1 \mid A_1=perto, A_2=esquerdo) = P(A_1=perto \mid Classe=+1) \\ &\times P(A_2=esquerdo \mid Classe=+1) \times P(Classe=+1) = 0{,}5838 \times 0{,}1946 \times 0{,}5 = 0{,}0568 \\ &P(Classe=-1 \mid A_1=perto, A_2=esquerdo) = P(A_1=perto \mid Classe=-1) \\ &\times P(A_2=esquerdo \mid Classe=-1) \times P(Classe=-1) = 0 \times 0{,}6667 \times 0{,}5 = 0 \end{aligned} \quad (58)$$

Observa-se neste caso que, se o atributos A_1 for igual a perto e A_2 for igual a esquerdo, então se decide pela classe +1, pois é a que apresenta a maior probabilidade a posteriori. Repete-se o mesmo processo para as outras oito combinações possíveis de atributos:

$$\begin{aligned} &P(Classe=+1 \mid A_1=perto, A_2=centro) = P(A_1=perto \mid Classe=+1) \\ &\times P(A_2=centro \mid Classe=+1) \times P(Classe=+1) = 0{,}5838 \times 0{,}5838 \times 0{,}5 = 0{,}1704 \\ &P(Classe=-1 \mid A_1=perto, A_2=centro) = P(A_1=perto \mid Classe=-1) \\ &\times P(A_2=centro \mid Classe=-1) \times P(Classe=-1) = 0 \times 0 \times 0{,}5 = 0 \end{aligned} \quad (59)$$

$$\begin{aligned} &P(Classe=+1 \mid A_1=perto, A_2=direito) = P(A_1=perto \mid Classe=+1) \\ &\times P(A_2=direito \mid Classe=+1) \times P(Classe=+1) = 0{,}5838 \times 0{,}2216 \times 0{,}5 = 0{,}0647 \\ &P(Classe=-1 \mid A_1=perto, A_2=direito) = P(A_1=perto \mid Classe=-1) \\ &\times P(A_2=direito \mid Classe=-1) \times P(Classe=-1) = 0 \times 0{,}3333 \times 0{,}5 = 0 \end{aligned} \quad (60)$$

$$\begin{aligned} &P(Classe=+1 \mid A_1=médio, A_2=esquerdo) = P(A_1=médio \mid Classe=+1) \\ &\times P(A_2=esquerdo \mid Classe=+1) \times P(Classe=+1) = 0{,}2216 \times 0{,}1946 \times 0{,}5 = 0{,}0216 \\ &P(Classe=-1 \mid A_1=médio, A_2=esquerdo) = P(A_1=médio \mid Classe=-1) \\ &\times P(A_2=esquerdo \mid Classe=-1) \times P(Classe=-1) = 0{,}3333 \times 0{,}6667 \times 0{,}5 = 0{,}1111 \end{aligned} \quad (61)$$

$$\begin{aligned} &P(Classe=+1 \mid A_1=médio, A_2=centro) = P(A_1=médio \mid Classe=+1) \\ &\times P(A_2=centro \mid Classe=+1) \times P(Classe=+1) = 0{,}22216 \times 0{,}5838 \times 0{,}5 = 0{,}0647 \\ &P(Classe=-1 \mid A_1=médio, A_2=centro) = P(A_1=médio \mid Classe=-1) \\ &\times P(A_2=centro \mid Classe=-1) \times P(Classe=-1) = 0{,}3333 \times 0 \times 0{,}5 = 0 \end{aligned} \quad (62)$$

$$\begin{aligned} &P(Classe=+1 \mid A_1=médio, A_2=direito) = P(A_1=médio \mid Classe=+1) \\ &\times P(A_2=direito \mid Classe=+1) \times P(Classe=+1) = 0{,}2216 \times 0{,}2216 \times 0{,}5 = 0{,}0245 \\ &P(Classe=-1 \mid A_1=médio, A_2=direito) = P(A_1=médio \mid Classe=-1) \\ &\times P(A_2=direito \mid Classe=-1) \times P(Classe=-1) = 0{,}3333 \times 0{,}3333 \times 0{,}5 = 0{,}0556 \end{aligned} \quad (63)$$

$$\begin{aligned} &P(Classe=+1 \mid A_1=longe, A_2=esquerdo) = P(A_1=longe \mid Classe=+1) \\ &\times P(A_2=esquerdo \mid Classe=+1) \times P(Classe=+1) = 0{,}1946 \times 0{,}1946 \times 0{,}5 = 0{,}0189 \\ &P(Classe=-1 \mid A_1=longe, A_2=esquerdo) = P(A_1=longe \mid Classe=-1) \\ &\times P(A_2=esquerdo \mid Classe=-1) \times P(Classe=-1) = 0{,}6667 \times 0{,}6667 \times 0{,}5 = 0{,}2222 \end{aligned} \quad (64)$$

$$\begin{aligned} &P(Classe=+1 \mid A_1=longe, A_2=centro) = P(A_1=longe \mid Classe=+1) \\ &\times P(A_2=centro \mid Classe=+1) \times P(Classe=+1) = 0{,}1946 \times 0{,}5838 \times 0{,}5 = 0{,}0568 \\ &P(Classe=-1 \mid A_1=longe, A_2=centro) = P(A_1=longe \mid Classe=-1) \\ &\times P(A_2=centro \mid Classe=-1) \times P(Classe=-1) = 0{,}6667 \times 0 \times 0{,}5 = 0 \end{aligned} \quad (65)$$

$$\begin{aligned} &P(Classe=+1 \mid A_1=longe, A_2=direito) = P(A_1=longe \mid Classe=+1) \\ &\times P(A_2=direito \mid Classe=+1) \times P(Classe=+1) = 0{,}1946 \times 0{,}2216 \times 0{,}5 = 0{,}0216 \\ &P(Classe=-1 \mid A_1=longe, A_2=direito) = P(A_1=longe \mid Classe=-1) \\ &\times P(A_2=direito \mid Classe=-1) \times P(Classe=-1) = 0{,}6667 \times 0{,}3333 \times 0{,}5 = 0{,}1111 \end{aligned} \quad (66)$$

Nos eventos 4, 6, 7 e 9 decide-se pela classe -1, pelo fato de apresentar a maior probabilidade a posteriori. Pelo mesmo motivo, o classificador decide pela classe +1 nas outras situações. A Tabela 4 compara a classificação correta de cada evento com a classificação extraída a partir das probabilidades a posteriori.

Comparando-se cada evento, percebe-se que este classificador só errou o evento 6, isto é, errou 5 das 200 ocorrências, ou uma taxa de erro de 5/200 (2,5%). Considera-se este classificador como sendo a segunda iteração de *Boosting*.

Para um erro de 2,5%, calcula-se o valor de α_2:

$$\alpha_2 = 0{,}5 \ln\left(\frac{1-\varepsilon_2}{\varepsilon_2}\right) = 0{,}5 \times \ln\left(\frac{1\text{-}0{,}025}{0{,}025}\right) = 2{,}1452 \quad (67)$$

Tabela 4 – Comparação entre o conjunto de treinamento e o resultado da segunda iteração de *Boosting*

EVENTO	A_1	A2	CLASSE CORRETA	CLASSE DE ACORDO COM O CLASSIFICADOR NAIVE BAYES
1	PERTO	ESQUERDO	+1	+1
2	PERTO	CENTRO	+1	+1
3	PERTO	DIREITO	+1	+1
4	MÉDIO	ESQUERDO	-1	-1
5	MÉDIO	CENTRO	+1	+1
6	MÉDIO	DIREITO	+1	-1
7	LONGE	ESQUERDO	-1	-1
8	LONGE	CENTRO	+1	+1
9	LONGE	DIREITO	-1	-1

Com este valor de α_2, é possível achar o classificador resultante da combinação dos dois classificadores individuais, de acordo com a expressão (11). Essa expressão mostra que para se achar o classificador combinado deve-se multiplicar o primeiro classificador por α_1, o segundo classificador por α_2 e somar os dois resultados. A Tabela 5 mostra este processo.

Para se compor o classificador final deve-se somar as últimas colunas de cada um dos classificadores da Tabela 5. A Tabela 6 mostra o resultado final do processo.

Este classificador final errou todos os exemplos dos eventos 6 e 9, gerando um erro de 2 × 5 = 10/200 = 5 %. Isto comprova o funcionamento da técnica, melhorando um classificador fraco. Continuando o processo em mais uma iteração, o classificador combinado consegue uma taxa de erro de 0 %, por se tratar de um exemplo simples.

Tabela 5 – Resumo da classificação dos dois classificadores fracos

CLASSIFICADOR 1			
EVENTO	CLASSIFICAÇÃO	α_1	CLASSIFICAÇÃO × α_1
1	1	1,2561	1,2561
2	1	1,2561	1,2561
3	1	1,2561	1,2561
4	1	1,2561	1,2561
5	1	1,2561	1,2561
6	1	1,2561	1,2561
7	1	1,2561	1,2561
8	1	1,2561	1,2561
9	1	1,2561	1,2561
CLASSIFICADOR 2			
EVENTO	CLASSIFICAÇÃO	α_2	CLASSIFICAÇÃO × α_2
1	1	2,1452	2,1452
2	1	2,1452	2,1452
3	1	2,1452	2,1452
4	-1	2,1452	-2,1452
5	1	2,1452	2,1452
6	-1	2,1452	-2,1452
7	-1	2,1452	-2,1452
8	1	2,1452	2,1452
9	-1	2,1452	-2,1452

Tabela 6 – Classificador Final

CLASSIFICADOR FINAL			
EVENTO	SOMATÓRIO	CLASSIFICAÇÃO	CLASSES REAIS
1	3,4014	1	1
2	3,4014	1	1
3	3,4014	1	1
4	-0,8891	-1	-1
5	3,4014	1	1
6	-0,8891	-1	1
7	-0,8891	-1	-1
8	6,8028	1	1
9	2,5123	1	-1

3.6 Seleção de Parâmetros Utilizando *Boosting*

Conforme percebido no item anterior, o AdaBoost não é uma técnica de seleção de parâmetros. Mas, com uma pequena modificação, pode-se alcançar este objetivo. Basta com que se escolha um algoritmo fraco que gere a cada iteração de Boosting, hipóteses que utilizem apenas um subconjunto de parâmetros.

Para esse fim, foi escolhido o algoritmo *Decision Stumps* (Iba; Langley, 1992), que pode ser definido como uma árvore de decisão de um único nó. O limiar de decisão deste classificador é sempre uma reta que depende única e exclusivamente de apenas um dos parâmetros. Na Figura 18, pode ser visto um conjunto de exemplos, sendo que o eixo horizontal representa um parâmetro qualquer e o eixo vertical, um segundo parâmetro. A reta horizontal que atravessa a área é o limiar. Acima desta reta, fica situada a classe +1 e abaixo, a classe –1 (por exemplo).

Este algoritmo pode ser utilizado em Seleção de Parâmetros, pois a cada iteração de *Boosting*, apenas um parâmetro é escolhido como sendo o parâmetro que melhor identifica a separação entre as duas classes. Logicamente, a melhor identificação é determinada pelo erro, ao se considerar um determinado parâmetro com um determinado valor.

Assim, a cada iteração de *Boosting*, o parâmetro selecionado é retirado do conjunto de parâmetros e na próxima iteração, o algoritmo de *Decision Stumps* escolhe um novo parâmetro que melhor separa as duas classes.

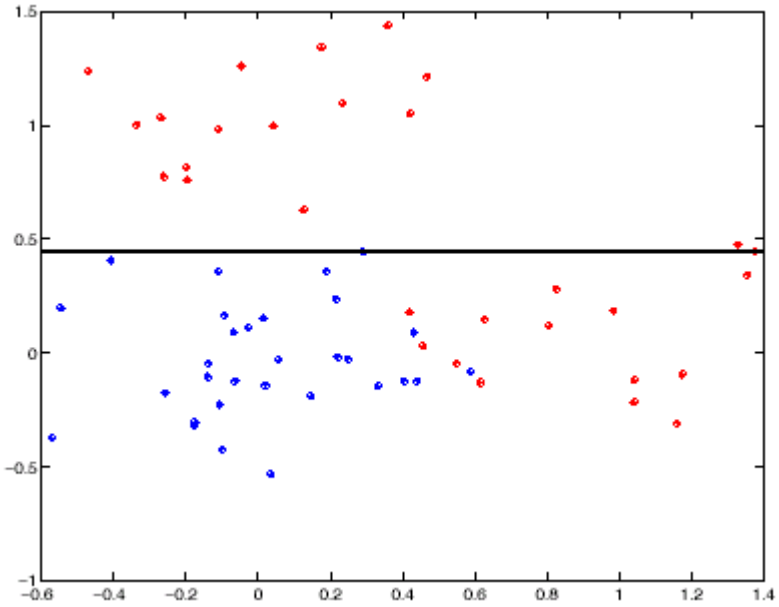

Figura 18 – Exemplo de aplicação do algoritmo *Decision Stumps*

3.7 Exemplo de Classificação Utilizando *Boosting*, *Decision Stumps* e Ganho de Informação

O exemplo a seguir aplica *Boosting* a um classificador *Decision Stumps*, utilizando como critério para criação da árvore de decisão a linha que gerar o maior Ganho de Informação.

A definição de Ganho de Informação vem da Teoria da Informação e é também conhecida como Informação Mútua (Haykin, 2001). A expressão que o define é

$$I(Y;X) = H(Y) - H(Y \mid X) \qquad (68)$$

onde $H(Y)$ é definido como a entropia do conjunto e $H(Y \mid X)$ como sendo a Entropia Condicional do conjunto Y dada a ocorrência de uma condição X.

Então, para realizar a classificação utilizando o algoritmo *Decision Stumps* escolhe-se uma linha que separe o conjunto de treinamento em duas classes (no caso de classificação

binária, -1 e +1). A partir daí, calcula-se $H(Y)$ que é a entropia do conjunto original baseado na informação antes do treinamento. A fórmula da Entropia é

$$H(Y) = \sum_{k=0}^{K-1} p_k \log_2\left(\frac{1}{p_k}\right) \qquad (69)$$

onde p_k é a probabilidade da k-ésima classe e K é o número de classes no conjunto. Neste trabalho, utiliza-se classificadores binários. Assim, $K = 2$.

Com a divisão do conjunto em duas áreas, analisa-se, a partir daí, a entropia em cada área, da seguinte forma: para a classe rotulada como +1 pela linha (por exemplo), devem existir alguns eventos classificados acertadamente e outros erroneamente. Aplica-se a expressão (69) neste subconjunto e para o subconjunto rotulado como –1. Achando a média ponderada entre esses dois valores, encontra-se o valor de $H(Y|X)$.

De posse dos valores de $H(Y)$ e $H(Y|X)$, basta substituir na expressão (68), para encontrar o Ganho de Informação. Repete-se este processo para todas as linhas utilizadas como hipótese. A que obtiver o maior Ganho de Informação é a escolhida pelo classificador.

O conjunto de treinamento utilizado neste exemplo é detalhado na Tabela 7 e mostrado na Figura 19. Observe que a classe +1 é representada por um × e a classe –1 por um O e existem dois atributos (parâmetros) discretos p_1 e p_2, que podem assumir os valores 0, 1 ou 2.

Tabela 7 – Conjunto Utilizado no Exemplo de *Decision Stumps*

Evento	**Classe**	**Probabilidade**
(0,0)	+1	0,0864
(0,1)	+1	0,2917
(0,2)	+1	0,1144
(1,0)	-1	0,0968
(1,1)	+1	0,09
(1,2)	+1	0,0383
(2,0)	-1	0,1411
(2,1)	+1	0,0618
(2,2)	-1	0,0794

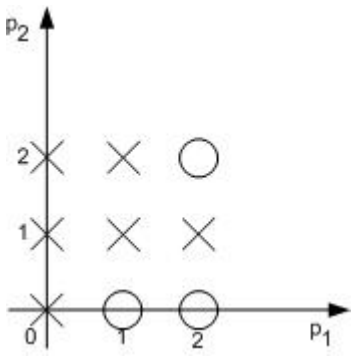

Figura 19 – Exemplo de conjunto de treinamento com 9 eventos possíveis

O primeiro passo é escolher as linhas que gerarão as separações entre as classes. Para o caso do eixo horizontal, devem ser criadas linhas que interceptem o eixo de p_1 em pontos intermediários às abscissas das amostras. No caso do conjunto de exemplo, as linhas podem ser p_1 = -0,5, p_1 = 0,5, p_1 = 1,5 e p_1 = 2,5.

Para calcular a entropia desse conjunto, torna-se necessário inicialmente descobrir a probabilidade da classe +1 e da classe –1, respectivamente. Basta somar as probabilidades dos eventos de cada classe. Assim, a probabilidade ser +1 é igual a 0,6826, enquanto a probabilidade da classe ser –1 é igual a 0,3174. A entropia desse conjunto é calculada de acordo com (69):

$$H(Y)=\sum_{k=0}^{K-1} p_k \log_2\left(\frac{1}{p_k}\right)=0{,}6826\log_2\left(\frac{1}{0{,}6826}\right)+0{,}3174\log_2\left(\frac{1}{0{,}3174}\right)=0{,}9015 \qquad (70)$$

A Figura 20 mostra a linha p_1 = 0,5 desenhada no conjunto de exemplo, como uma linha pontilhada. Adota-se a classe +1 para a região esquerda e a classe –1 para a região à direita da linha. Percebe-se que neste caso, quatro eventos seriam classificados errados: (1, 0), (1, 1), (1, 2) e (2, 1).

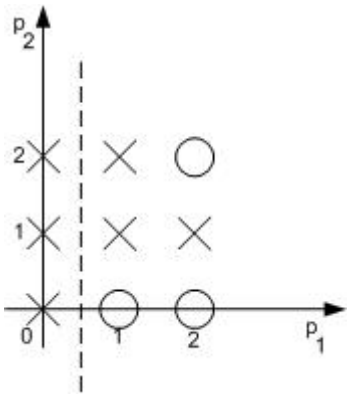

Figura 20 – Exemplo de conjunto de treinamento com uma linha em $p_1 = 0,5$ criando duas classes

A partir daí, pode ser calculada a entropia condicional, dada a linha de separação. No lado esquerdo da linha, a classe foi escolhida como +1. Nesse subconjunto, a probabilidade da classe original ser +1 é igual a 1 e a probabilidade da classe original ser –1 é igual a 0. Assim, percebe-se claramente que a entropia aqui é igual a 0.

Para a região da direita, a probabilidade da classe original ser +1 é igual a (0,0383+0,1411+0,0794) / (0,0968+0,09+0,0383+0,1411+0,0618+0,0794) = 0,3746 e a probabilidade da classe original ser –1 é igual a 1-0,3746 = 0,6254. Assim, a entropia resulta em 0,9541. Finalmente, aplica-se a média ponderada em ambos as regiões:

$$H(Y \mid X) = 0 \times 0,4925 + 0,9541 \times 0,5075 = 0,4842 \qquad (71)$$

Basta, agora, encontrar o Ganho de Informação a partir da expressão (68):

$$I(Y;X) = 0,9015 - 0,4842 = 0,4173 \qquad (72)$$

Repete-se este mesmo procedimento para as outras linhas em p_1 e para as linhas em p_2. A Tabela 8 mostra os resultados encontrados para este exemplo.

Tabela 8 – Ganho de Informação para cada linha de separação na primeira iteração de Boosting

Linhas de separação	Ganho de Informação
p_1 = -0,5	0
p_1 = 0,5	0,4173
p_1 = 1,5	0,2778
p_1 = 2,5	0
p_2 = -0,5	0
p_2 = 0,5	0,2775
p_2 = 1,5	0,0006
p_2 = 2,5	0

Observando a Tabela 8, pode-se perceber que a linha que gerou o maior Ganho de Informação foi a definida como p_1 = 0,5. A Figura 20 mostra o conjunto utilizado neste exemplo com a linha em pontilhado.

Neste caso, pode-se perceber que a melhor escolha é determinar que a região à esquerda da linha seja definida como a classe +1 e que à direita corresponda à classe –1. Nesta situação, haverá erro em apenas três eventos: (1, 1), (1, 2), (2,1). Somando as probabilidades, o erro deste classificador é de apenas 0,09+0,0383+0,0618 = 0,1901. A técnica de *boosting* permite reduzir essa taxa de erro, com algumas iterações.

A partir do erro ε_1 de 0,1901, pode-se calcular o valor de α_1:

$$\alpha_1 = 0{,}5\ln\left(\frac{1-\varepsilon_1}{\varepsilon_1}\right) = 0{,}5\times\ln\left(\frac{1\text{-}0{,}1901}{0{,}1901}\right) = 0{,}7247 \qquad (73)$$

O evento (1, 0) tinha peso igual a 0,0864 e foi um evento onde o classificador acertou. Assim, seu novo peso será:

$$D_2 = D_1\exp\left(-\alpha_1 y h_1(x)\right) = 0{,}0864\times\exp\left(-0{,}7247\times1\times1\right) = 0{,}0419 \qquad (74)$$

O evento (1, 1) tinha peso igual a 0,09 e foi um evento onde o classificador errou. Assim, seu novo peso será:

$$D_2 = D_1 \exp\left(-\alpha_1 y h_1(x)\right) = 0{,}09 \times \exp\left(-0{,}7247 \times (-1) \times 1\right) = 0{,}1857 \qquad (75)$$

A Tabela 9 mostra os valores de cada peso, já devidamente alterados para a segunda iteração. Para iniciar a segunda iteração de *Boosting*, torna-se necessário normalizar o conjunto para que a soma dos pesos seja igual a 1, para que os pesos possam ser chamados de probabilidades. A última coluna mostra os pesos já normalizados, prontos para serem utilizados na segunda iteração.

Tabela 9 – Probabilidade de cada evento após a primeira iteração de *Boosting*

Evento	Probabilidades na Primeira Iteração	Pesos na Segunda Iteração	Probabilidades na Segunda Iteração
(0, 0)	0,0864	0,0419	0,0583
(0, 1)	0,2917	0,1413	0,1968
(0, 2)	0,1144	0,0554	0,0772
(1, 0)	0,0968	0,1857	0,2586
(1, 1)	0,09	0,0791	0,1102
(1, 2)	0,0383	0,0186	0,0258
(2, 0)	0,1411	0,1275	0,1776
(2, 1)	0,0618	0,0299	0,0417
(2, 2)	0,0794	0,0385	0,0536

Com os novos valores de probabilidade, são refeitos os cálculos de Entropia Condicional e Ganho de informação. A Tabela 10 ilustra os valores obtidos na segunda iteração.

Tabela 10 – Ganho de Informação para cada linha de separação na segunda iteração de Boosting

Linhas de separação	Ganho de Informação
$p_1 = -0,5$	0
$p_1 = 0,5$	0,4415
$p_1 = 1,5$	0,1486
$p_1 = 2,5$	0
$p_2 = -0,5$	0
$p_2 = 0,5$	0,4944
$p_2 = 1,5$	0,0119
$p_2 = 2,5$	0

Neste caso é escolhida a linha $p_2 = 0,5$, devido gerar um maior Ganho de Informação ao conjunto. A Figura 21 mostra o conjunto de exemplo com a linha $p_2 = 0,5$ em pontilhado.

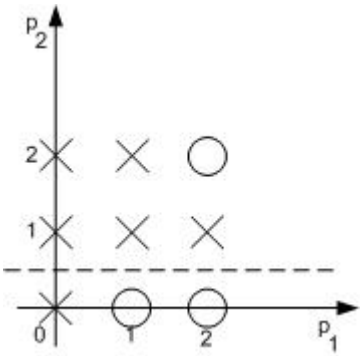

Figura 21 – Exemplo de conjunto de treinamento com uma linha em $p_2 = 0,5$ criando duas classes

Neste caso, abaixo da linha considera-se classe –1 e acima classe +1. Assim, haverá erro nos eventos (0, 0) e (2, 2). O erro desta iteração será a soma das probabilidades destes dois eventos, que é igual a 0,1119. Recalculando o α para a segunda iteração:

$$\alpha_2 = 0{,}5 \ln\left(\frac{1-\varepsilon_2}{\varepsilon_2}\right) = 0{,}5 \times \ln\left(\frac{1 - 0{,}1119}{0{,}1119}\right) = 1{,}0358 \qquad (76)$$

De posse do valor de α_1 e α_2, pode ser montado o classificador combinado das duas iterações de classificadores dito "fracos". Assim, a Tabela 10 mostra os dois classificadores.

Para se compor o classificador final deve-se somar as últimas colunas de cada um dos classificadores da Tabela 11. A Tabela 12 mostra o classificador final.

Neste caso, houve erro apenas no primeiro e no último evento. A soma das suas probabilidades resulta em uma taxa de erro de 0,1658. Houve uma redução em relação à primeira iteração onde a taxa de erro foi de 0,1901, comprovando a eficiência do algoritmo.

Tabela 11 – Resumo dos dois classificadores fracos

CLASSIFICADOR 1			
EVENTO	CLASSIFICAÇÃO	α_1	CLASSIFICAÇÃO × α_1
1	1	0,7247	0,7247
2	1	0,7247	0,7247
3	1	0,7247	0,7247
4	-1	0,7247	-0,7247
5	-1	0,7247	-0,7247
6	-1	0,7247	-0,7247
7	-1	0,7247	-0,7247
8	-1	0,7247	-0,7247
9	-1	0,7247	-0,7247
CLASSIFICADOR 2			
EVENTO	CLASSIFICAÇÃO	α_2	CLASSIFICAÇÃO × α_2
1	-1	1,0358	-1,0358
2	1	1,0358	1,0358
3	1	1,0358	1,0358
4	-1	1,0358	-1,0358
5	1	1,0358	1,0358
6	1	1,0358	1,0358
7	-1	1,0358	-1,0358
8	1	1,0358	1,0358
9	1	1,0358	1,0358

Tabela 12 – Classificador final com *Decision Stumps*

CLASSIFICADOR FINAL			
EVENTO	SOMATÓRIO	CLASSIFICAÇÃO	CLASSES REAIS
1	-0,3111	-1	1
2	1,7605	1	1
3	1,7605	1	1
4	-1,7605	-1	-1
5	0,3111	1	1
6	0,3111	1	1
7	-1,7605	-1	-1
8	0,3111	1	1
9	0,3111	1	-1

4 Simulações e Resultados

4.1 Simulações

Esta seção detalha o ambiente montado para a implementação do Classificador, descrevendo, inclusive, a base de dados e os front-end´s utilizados.

4.1.1 Base de dados utilizada

As simulações deste trabalho utilizaram a base de dados TIMIT, que é o *corpus* mais popular dentre os distribuídos pelo LDC (*Linguistic Data Consortium*) (LDC, 2006). É um *corpus* de voz contínua de alta qualidade, com interlocutores norte-americanos, acompanhado da transcrição fonética e ortográfica, em arquivos anexos. Ele foi gravado pela *Texas Instruments* (TI), transcrito no *Massachussets Institute of Technology* (MIT), e preparado para distribuição pelo *National Institute of Standards and Technology* (NIST) dos EUA. A transcrição fonética, indicando onde inicia e termina cada alofone demandou cerca de 100 a 1000 horas de trabalho para cada hora de voz, demonstrando o grau de precisão desta base de dados (Klautau et al, 2003).

O TIMIT contém voz de 630 interlocutores, sendo 438 homens e 192 mulheres. O *corpus* foi dividido em dois conjuntos: de treino e de teste. Os interlocutores que estão em um conjunto não estão em outro. Cada interlocutor gravou 10 sentenças "ricas" foneticamente. No conjunto de treino existem 30132 palavras, sendo que a sentença mais longa é a que possui 18 palavras. Por outro lado, existe uma sentença com apenas duas palavras. A maior parte das sentenças contém de 4 a 13 palavras. A palavra que possui o maior número de alofones é a que possui 18 alofones. A palavra mais longa tem uma duração de 1,63 segundos. O conjunto de treino possui 142910 alofones. Maiores detalhes sobre a base de dados TIMIT podem ser encontrados em Klautau et al (2003) e Halberstadt (1998).

Conforme Lee; Hon (1989), os 61 símbolos existentes nas transcrições fonéticas do TIMIT são divididos em 39 classes. Estas 39 classes foram as classes utilizadas neste trabalho. Além disso, foram excluídas as duas sentenças *sa*, que equivalem às sentenças de dialeto. Elas foram incluídas no TIMIT para representar as variações dialéticas dos interlocutores. Os testes foram realizados com um subconjunto do conjunto de teste, chamado

Core Test composto de 192 sentenças = 8 sentenças multiplicado por 24 interlocutores (já excluídas as duas dialéticas).

Para efeito de ilustração, a Tabela 13 apresenta alguns resultados de taxa de erro, encontrados na literatura, utilizando a base de dados TIMIT. GMM refere-se ao classificador que usa uma mistura de Gaussianas, o qual pode ser visto como uma HMM contínua de um único estado. Ressalta-se que os trabalhos mencionados não utilizaram exatamente a mesma metodologia, e os números devem ser interpretados a partir de uma leitura atenta dos artigos. Por exemplo, alguns autores descartam completamente a oclusiva glotal /ʔ/, enquanto outros não. Outro ponto de divergência é que alguns trabalhos utilizam todo o conjunto de teste e outros, apenas o *Core Test*. Os dois primeiros trabalhos utilizaram HMM´s dependentes do contexto (CD).

Tabela 13 – Taxa de erro para classificação fonética usando TIMIT

Autor (es) (Referência)	Erro	Observação
Chengalvarayan e Deng (Chengalvarayan; Deng, 1997)	18,5	CD HMM
Chengalvarayan e Deng (Chengalvarayan; Deng, 1998)	16,5	CD HMM
Chengalvarayan e Deng (Chengalvarayan; Deng, 1997)	31,8	HMM
Salomon, Simon e Osborne (Salomon et al, 2002)	28,6	SVM
Chengalvarayan e Deng (Chengalvarayan; Deng, 1998)	27,6	HMM
Zahorian, Silsbee e Wang (Zahorian et al, 1997)	23,0	Redes Neurais
Hazen e Halberstadt (Hazen; Halberstadt, 1998)	20,2	GMM
Halberstadt (Halberstadt, 1998)	18,3	GMM

4.1.2 Classificador (SVM)

Como as SVM´s foram criadas para resolver problemas binários, neste trabalho foi utilizado o esquema ECOC com a matriz *all-pairs*, e decodificação de Hamming, para distinguir a classe fonética correta dentre as 39 possíveis. Foram treinadas 741 SVM´s (combinação de 39, 2 a 2). Na fase de teste, o vetor a ser classificado é submetido a todas as 741 SVM´s e o fone com maior número de "vitórias" é declarado vencedor. Percebe-se que o número máximo de vitórias que um fone pode obter é 740, pois não existe uma SVM comparando o fone com ele mesmo.

Outra adaptação realizada foi a relacionada ao tamanho dos vetores. Todo classificador exige vetores de entrada com número fixo de elementos. Mas os fones possuem uma duração variável. A maneira utilizada, por ser bastante simples, foi a de mapear qualquer fone em um vetor de comprimento L, como descrito em Ganapathiraju (2002). A idéia é utilizar um *front-end* convencional no primeiro estágio, onde, para cada fone, são gerados T vetores de K parâmetros cada, a uma taxa fixa r. Após isso, divide-se linearmente os T quadros do fone em três segmentos de acordo com a razão 3-4-3. Uma vez determinados os três segmentos, retira-se a média dos parâmetros associados a cada um deles e obtém-se um vetor com $L = 3K$ elementos. A adoção de três segmentos inspira-se em RAV baseados em HMM´s, onde os fones são geralmente modelados como compostos de três partes aproximadamente estacionárias. Diversos trabalhos, como por exemplo, Sakoe; Chiba (1978), indicam que a técnica correta deveria ser o DTW, mas que implicaria em uma complexidade maior na implementação.

Para melhor convergência das SVM´s, os arquivos de treino foram normalizados de forma que todos os parâmetros ficassem restritos à faixa [0,1]. Os mesmos fatores de normalização obtidos com os dados de treino foram utilizados para normalizar os dados de teste.

4.1.3 Front-end´s

Os *front-end´*s utilizados foram descritos no Capítulo 2, sendo que a concatenação simples de todos os parâmetros resultantes gera um conjunto altamente redundante, que possibilita a utilização da técnica de Seleção de Parâmetros.

O primeiro *front-end* foi o Formantes, consistindo da freqüência fundamental F0, probabilidade de vozeamento e as quatro primeiras formantes. Estes parâmetros foram extraídos através dos algoritmos desenvolvidos por David Talkin (1987, 1995), encontrado no aplicativo Snack (Snack, 2006).

O segundo *front-end* foi o PLP tradicional (Hermansky, 1990), com 39 parâmetros por quadro, sendo que o primeiro corresponde à energia do quadro, do segundo ao décimo-terceiro correspondem aos 12 coeficientes estáticos, e os restantes, à primeira e segunda derivadas destes 13 anteriores.

Os próximos dois conjuntos de parâmetros, com 40 parâmetros cada, correspondem aos estágios *synchrony* e *envelope* do modelo Seneff para o sistema auditivo (Seneff, 1984).Neste modelo, todo o processamento é feito no domínio do tempo. Não é utilizada a FFT devido às não-linearidades assumidas pelo modelo. Assim, o tempo de processamento deste *front-end* chega a ser 100 vezes o tempo real.

Os dois próximos conjuntos são o MFCC (Davis; Merlmestein, 1980) e RASTA (Hermansky; Morgan, 1994) tradicionais, com 39 parâmetros cada.

O último *front-end*, com 50 parâmetros por quadro é chamado de Banco de filtros e consiste da potência de saída de 24 filtros espaçados na escala Mel (Stevens; Volkman, 1940), a energia normalizada do quadro e a primeira derivada destes 25 parâmetros.

Concatenando-se os parâmetros dos sete *front-end´s*, obtém-se um vetor com 253 elementos para representar cada quadro. Empregando o método citado na seção anterior, obtém-se 759 parâmetros. Além disso, foi incluído um último parâmetro, a duração de cada fone em número de quadros, totalizando assim, 760 parâmetros.

4.2 Resultados

Nesta seção serão apresentados os resultados obtidos com SVM´s para classificação fonética, associadas ao uso de AdaBoost para Seleção de Parâmetros.

4.2.1 Classificação Fonética

A Tabela 14 mostra a taxa de erro para diferentes situações. A penúltima coluna indica o número de parâmetros distintos quando são aglutinados todos os parâmetros utilizados pelas SVM´s. Por exemplo, na penúltima linha, são adotados 25 parâmetros por SVM. Neste caso,

apenas 498 dos 760 parâmetros são utilizados. Percebe-se que o erro já é menor que os *front-end´s* mais utilizados (MFCC e PLP).

Percebe-se que os resultados são competitivos com os apresentados na literatura, ilustrados na Tabela 13. Em especial, nota-se que quanto mais alta a dimensão do espaço de entrada da SVM, menor é o erro, apesar da alta redundância. O valor de 21% é próximo do melhor resultado da Tabela 13 (nos casos independentes do contexto), reportado em Halberstadt (1998). Mais importante que a taxa de erro, é ganhar informações sobre os *front-end´s*, possibilitando melhorar os projetos futuros.

4.2.2 Seleção de Parâmetros

A Tabela 15 mostra os dez parâmetros mais "importantes" de acordo com *AdaBoost* para a distinção entre os pares de consoantes plosivas (/p/, /b/), (/d/, /t/) e (/g/,/k/). Como cada parâmetro, com exceção de duração, é utilizado em três segmentos, os sufixos "s1", "s2" e "s3" são usados para indicar do primeiro ao terceiro segmento, respectivamente. As letras "d" e "a" indicam a primeira e segunda derivadas, respectivamente.

Tabela 14 – Taxa de erro para classificação fonética usando SVM

Front-end	Parâmetros por fone	Parâmetros por SVM	Parâmetros distintos	Erro (%)
PLP + duração	118	118	118	28,2
MFCC + duração	118	118	118	29,1
todos	760	760	760	21,0
todos	760	100	758	22,7
todos	760	40	755	25,3
todos	760	25	498	26,6
todos	760	5	464	34,6

Como podia ser esperado, a Tabela 15 indica que para os três pares de consoantes, o parâmetro considerado mais importante foi a probabilidade de vozeamento (seguida pela duração). Vale ressaltar que o *AdaBoost* busca iterativamente encontrar parâmetros que são

complementares ao já escolhidos, o que leva a melhores resultados do que a escolha independente dos parâmetros, como mostrado em Klautau (2003). Por exemplo, para o par (/p/, /b/), após a escolha de probVozeamento-s1, os demais parâmetros não possuem uma relação direta com vozeamento. Por outro lado, para o par (/k/, /g/), *AdaBoost* volta a utilizar parâmetros como probVozeamento-s3 e F0-s1.

Tabela 15 – Os primeiros 10 parâmetros selecionados por *AdaBoost* para classificar pares de consoantes plosivas cuja distinção é a existência ou não de vozeamento

rank	/p/ versus /b/	/d/ versus /t/	/k/ versus /g/
1	probVozeamento-s1	probVozeamento-s2	probVozeamento-s1
2	duração	duração	duração
3	seneff-env-13-s2	mfcc-1-s1	F1-s3
4	seneff-syn-8-s1	plp-2-s1	seneff-env-34-s1
5	seneff-syn-7-s1	F0-s2	seneff-env-35-s1
6	fbank-4-s2	seneff-env-14-s1	plp-d-3-s2
7	seneff-env-40-s1	seneff-env-19-s1	rasta-d-9-s2
8	rasta-d-8-s2	rasta-d-2-s2	probVozeamento-s3
9	F1-s3	rasta-a-2-s3	F0-s1
10	mfcc-d-4-s1	seneff-env-4-s1	plp-d-8-s3

A Tabela 16 é similar à anterior, mas adota outros pares de fones.

Pode-se observar que o *AdaBoost* privilegia os parâmetros de *front-end* de Seneff, para a distinção das nasais (/n/, /ɲ/). Obviamente, isso não atesta que outros parâmetros não possam atingir bom desempenho na classificação de (n, /ɲ/), mas aponta para a necessidade de uma maior investigação. Nota-se a escolha de fbank-22-s2 na distinção entre (/s/, /f/), onde esse parâmetro é a potência na saída do 22° (antepenúltimo) filtro. Esse tipo de informação, obtida em freqüência relativamente alta, não é privilegiada em front-end´s convencionais. Outra observação é a importância dada às formantes no caso do par (/a/, /oʊ/), enquanto que nos outros dois, as formantes não aparecem.

Tabela 16 – Os primeiros 10 parâmetros selecionados por *AdaBoost* para classificar alguns pares de fones

rank	/s/ versus /f/	/a/ versus /oʊ/	/n/ versus /ɲ/
1	fbank-22-s2	F1-s2	seneff-env-24-s1
2	mfcc-2-s2	fbank-d-12-s2	seneff-env-23-s1
3	seneff-syn-38-s2	plp-3-s3	seneff-env-25-s1
4	fbank-3-s3	mfcc-3-s3	seneff-syn-32-s1
5	seneff-syn-39-s2	F3-s3	seneff-syn-31-s1
6	plp-5-s2	F1-s3	fbank-d-3-s1
7	plp-2-s2	F0-s3	seneff-env-24-s3
8	mfcc-2-s1	rasta-3-s3	fbank-d-6-s1
9	fbank-10-s3	F1-s1	seneff-env-23-s3
10	fbank-d-energy-s1	seneff-syn-34-s2	plp-d-1-s1

Nas Figuras 22 e 23, são mostrados os histogramas dos parâmetros selecionados ao se usar L_b = 5 e 100 por SVM, respectivamente. Para melhorar a visualização, foram agrupadas as ocorrências dos parâmetros nos três segmentos. Por exemplo, todas as ocorrências de plp-5-s1, plp-5-s2 e plp-5-s3 foram somadas e estão representadas pelo quinto elemento do *front-end* PLP. O parâmetro duração não aparece nos histogramas, mas foi selecionado 221 e 479 vezes, quando L_b = 5 e 100 por SVM, respectivamente.

Considerando as ocorrências nos três segmentos, probVozeamento foi escolhida 472 e 1102 vezes, para L_b = 5 e 100, respectivamente. Obviamente, com o aumento de L_b, os histogramas (se normalizados) convergem para uma distribuição uniforme. Comparando-se as Figuras 22 e 23, observa-se que os parâmetros de cada *front-end* mantém aproximadamente sua importância relativa aos outros do mesmo *front-end* ao variarmos L_b de 5 para 100. A exceção notória são os três últimos parâmetros do RASTA, que despontam quando L_b = 5 mas não mantém o status na Figura 23. Essas e outras informações "mineradas" nos experimentos, merecem mais pesquisa em busca de explicações adequadas.

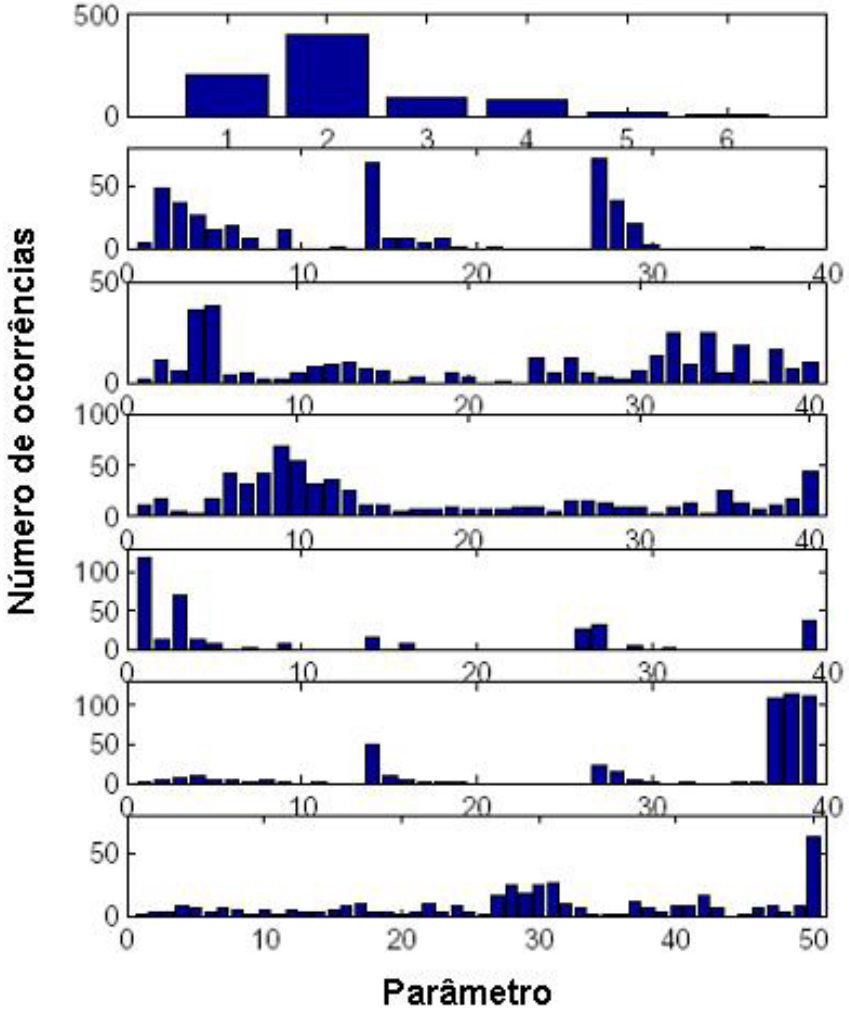

Figura 22 – Histogramas para 5 Parâmetros Selecionados por SVM, para os Front-End´s Formantes, PLP, Synchrony, Envelope, MFCC, RASTA e Banco de Filtros, respectivamente

Outro tipo de inferência interessante consiste em se identificar os parâmetros que facilitam a distinção entre classes fonéticas distintas. Por exemplo, a Tabela 17 mostra os parâmetros mais "importantes" e o número de vezes que cada um foi escolhido ao se considerar L_b =10 e todos os pares envolvendo vogais (e semi-vogais) versus consoantes. Pode-se atentar para a importância das formantes e do uso da segunda derivada.

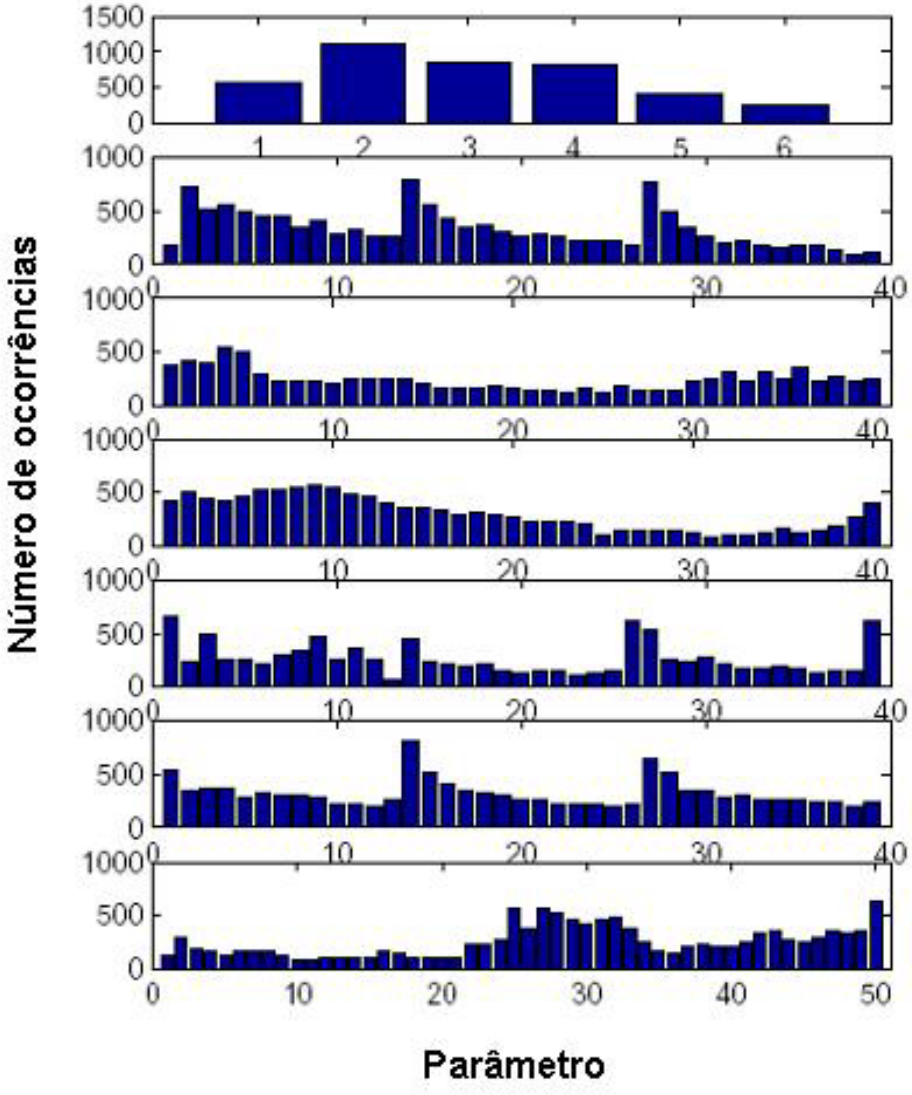

Figura 23 – Histogramas para 100 Parâmetros Selecionados por SVM, para os Front-End´s Formantes, PLP, Synchrony, Envelope, MFCC, RASTA e Banco de Filtros, respectivamente

Tabela 17 – Parâmetros mais "importantes" e número de vezes que foi escolhido para todos os pares de vogais versus consoantes

rank	Parâmetro	Ocorrências
1	probVozeamento – s2	178
2	duração	168
3	F1-s2	117
4	F0-s2	113
5	F2-s2	104
6	rasta-a-energy-s1	104
7	rasta-a-12-s1	102
8	F4-s2	99
9	plp-1-s2	97
10	rasta-a-11-s1	93

5 Conclusões

Este trabalho aplica, em Classificação Fonética, conceitos já conhecidos de Seleção de Parâmetros com Boosting, já utilizados em outras áreas (como Processamento de Imagens), possibilitando a extração de novas inferências sobre quais os parâmetros mais adequados para a classificação de determinadas classes fonéticas, ou pares de fones.

Para se obter um front-end competitivo, foram utilizadores Classificadores SVM heterogêneos, aperfeiçoados através da técnica de Boosting, sendo que foram utilizados 760 parâmetros conhecidos, o que fez com que o sistema ficasse com um alto custo computacional.

Os resultados mostram que esses classificadores são competitivos, mas principalmente quando o espaço de entrada possui dimensão elevada, mesmo que os parâmetros sejam altamente redundantes. Foram encontradas taxas de erro na base TIMIT, na ordem de 21%, com todos os 760 parâmetros disponíveis e na ordem de 26% com apenas 40 parâmetros por SVM, o que ajuda a reduzir a complexidade computacional, se comparado ao PLP, que consegue apenas 28% com 118 parâmetros por SVM.

Este método possibilita uma nova visão de projeto dos *front-end´s*: por que não utilizar um, com um parâmetro oriundo do MFCC, outro do Seneff, outro do RASTA, e assim sucessivamente. Para se chegar nestas conclusões foi utilizado um classificador do tipo *Decision Stumps*, que ajuda a definir quais os melhores parâmetros para um determinado tipo de classificação. Além disso, cada par de fones pode utilizar um conjunto característico de parâmetros, pois para cada par existe um grupo de parâmetros mais adequado, mais relevante. Por exemplo, para separar o p do b, o ideal é utilizar a Probabilidade de Vozeamento – s1, enquanto que para separar as vogais aa e ow o ideal é utilizar a Primeira Formante – s2.

Outra conclusão interessante foi a de que para separar vogais de consoantes, o melhor parâmetro é a Probabilidade de Vozeamento – s2.

Isto traz uma motivação para inserir, em trabalhos futuros, outros coeficientes, como por exemplo, os DCTC´s de Zahorian; Nossair (1999) ou os perfis espectrais de energia de Araújo (2000). Outros trabalhos futuros englobam a utilização de outros métodos de Seleção de Parâmetros, descritos em Hall (1999), e o desenvolvimento de um front-end heterogêneo para reconhecimento de dígitos.

REFERÊNCIAS BIBLIOGRÁFICAS

ANDREÃO, Rodrigo. **Implementação em tempo real de um sistema de reconhecimento de dígitos conectados.** 2001. 79f. Dissertação de Mestrado. UNICAP, 2001.

ARAUJO, Antonio. **Jogos computacionais fonoarticulatórios para crianças com deficiência auditiva.** 2000. 157f. Tese de Doutorado. UNICAMP, 2000.

BOSER, Bernhard et al. A training algorithm for optimum margin classifiers. In: Fifth Annual Workshop on Computational Learning Theory, 1992, Pittsburg. **Proceedings of the COLT 1992.** Pittsburg, EUA, 1992, p144-152.

BOURLARD, Hervé et al. Multi-stream speech recognition. **CC-AI, The Journal for the Integrated Study of Artificial Intelligence, Cognitive Science and Applied Epistemology.** v.15, n.3, p.215-234, 1998.

CHENGALVARAYAN, Rathi; **DENG,** Li. Use of generalized dynamic feature parameters for speech recognition. **IEEE Transations on Speech and Audio Processing.** v.5, p.232-242, 1997.

_____. Speech trajectory discrimination using the minimum classification error learning. **IEEE Transations on Speech and Audio Processing.** v.6, p.505-515, 1998.

CORTES, Corinna; **VAPNIK,** Vladimir. Support-vector networks. **Machine Learning.** v.20 n.3, p.279-297, 1995.

COSI, Piero. On The Use of Auditory Models in Speech Technology. In **Lecture Notes in Artificial Intelligence: Intelligent Perception Systems**, v.745, 1993.

DAVIS, Steven; **MERLMESTEIN,** Paul. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. **IEEE Transations on ASSP.** p.357-366, Ago. 1980.

DUDA, Richard; **HART**, Peter. **Pattern Classification.** New York: John Wiley & Sons, 2000.

FREUND, Yoav; **SCHAPIRE,** Robert. A decision-theoretic generalization of on-line learning and an applications to boosting. In: European Conference on Computational Learning Theory, 13-15 mar. 1995, Barcelona. **Proceedings of Second European Conference on Computational Learning Theory .** Barcelona, Espanha, 1995, p23-37.

GANAPATHIRAJU, Aravind. **Support Vector Machines for Speech Recognition.** 2002. 200f. PhD thesis. Mississippi State University, 2002.

GUNN, Steve. 1998. **Support Vector Machines for Classification and Regression.** University of Southampton, Technical Report.

HALBERSTADT, Andrew. **Heterogeneous acoustic measurements and multiple classifiers for speech recognition.** 1998. 173f. PhD thesis. MIT, 1998.

HALL, Mark. **Correlation-based feature selection for machine learning**. 1999. 198f. PhD thesis. Universidade de Waikato, 1999.

HAYKIN, Simon. **Communication Systems**. New York: John Wiley & Sons, 2001.

HAZEN, Timothy; **HALBERSTADT**, Andrew. Using aggregation to improve the performance of mixture Gaussian acoustic models. 1998, Seattle. **Proceedings of the IEEE ICASSP.** Seattle, EUA, 1998, p653-656.

HERMANSKY, Hynek. Perceptual linear predictive (PLP) analysis of speech. **Journal of the Acoustical Society of America.** v.87, n.4, p.1738-1752, Abr. 1990.

HERMANSKY, Hynek; **MORGAN**, Nelson. Rasta processing of speech. **IEEE Transations on ASSP.** v.2, n.4, p.578-589, Out. 1994.

HUANG, Xuedong et al. **Spoken Language Processing.** New Jersey: Prentice-Hall, 2001.

IBA, Wayne; **LANGLEY**, Pat. Induction of one-level decision-trees, 1992, Aberdeen. **Proceedings of the Ninth International Conference on Machine Learning.** Aberdeen, Escócia, 1992, p233-240.

KLAUTAU, Aldebaro. Mining speech: automatic selection of heterogeneous features using boosting, 2003, Hong Kong. **Proceedings of the IEEE ICASSP.** Hong Kong, China, 2003, p312-315.

KLAUTAU, Aldebaro et al. Speech recognition based on discriminative classifiers, 2003, Rio de Janeiro. **Anais do SBT 2003.** Rio de Janeiro, Brasil, 2003.

_____. On nearest-neighbor ECOC with application to all-pairs multiclass SVM. **Journal of Machine Learning Research.** v.4, n.1, p1-15, Jan 2004.

LDC. **LDC Top Ten Corpora**. Disponível em: <http://www.ldc.upenn.edu/Catalog/topten.jsp>. Acesso em: 01/01/2006.

LEE, Kai-Fu; **HON**, Hsiao-Wuen. Speaker-independent phone recognition using hidden Markov models. **IEEE Transations on ASSP.** v.37, n.11, p.1641-1648, Nov. 1989.

MAKHOUL, John; **COSELL**, Lynn. LPCW: An LPC vocoder with linear predictive spectral warping. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, Abr 1976, Philadelphia. **Proceedings of the IEEE ICASSP.** Philadelphia, EUA, 1976, p466-469.

MEIR, Ron; **RATSCH,** Gunnar. An introduction to boosting and leveraging, In **Advanced Lectures in Machine Learning**, p.119-184, 2003.

MOORE, Brian; **GLASBERG**, Brian. Suggested formulae for calculating auditory-filter bandwidths and excitation patterns. **Journal of the Acoustical Society of America.** v.74, p.750-753, 1983.

PETERSON, Gordon; **BARNEY,** Harold. Control methods used in a study of the vowels. **Journal of the Acoustical Society of America.** v.24, n.2, p.175-184, Mar. 1952.

RABINER. Lawrence; **JUANG**, Biing-Hwang. **Fundamentals of Speech Recognition.** New Jersey: Prentice-Hall, 1993.

RABINER. Lawrence; **SCHAFER**, Ronald. **Digital Processing of Speech Signals.** New Jersey: Prentice-Hall, 1978.

SAKOE, Hiroaki; **CHIBA**, Shigeru. Dynamic programming algorithm optimization for spoken word recognition. **IEEE Transations on ASSP.** v.26, n.1, p.43-49, 1978.

SALOMON, Jesper et al. Framewise phone classification using support vector machines. 2002, Denver. **Proceedings of the International Conference on Spoken Language Processing.** Denver, EUA, 2002, p2645-2648.

SCHAPIRE, Robert. The boosting approach to machine learning: an overview, 2002, New York. **Proceedings of the MSRI Workshop on Nonlinear Estimation and Classification.** New York, EUA, 2002, n.171, p149-172.

SCHROEDER, Manfred. Recognition of complex acoustic signals. In **Life Sciences Research Report**, v.55, p.324, 1977.

SENEFF, Stephanie. Pitch and spectral estimation of speech based on auditory synchrony model. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, Mar 1984, San Diego. **Proceedings of the IEEE ICASSP.** San Diego, EUA, 1984, v.3, p3/1-4.

_____. A joint synchrony/mean-rate model of auditory speech processing. **Journal of Phonetics.** Special Issue, v.16 (1) , p.55-76, Jan 1988.

SNACK. The Snack Sound Toolkit. Disponível em: < http://www.speech.kth.se/snack>. Acesso em: 01/01/2006.

STEVENS, Stanley; **VOLKMAN,** John. The relation of pitch to frequency. **Journal of Psychology.** v.53, p.329, 1940.

TALKIN, David. 1987. **Speech formant trajectory estimation using dynamic programming with modulated transition costs.** AT & T Bell Laboratories, Technical Report 11222-870720-07TM.

_____. A robust algorithm for pitch tracking (RAPT). In **Speech Coding and Synthesis**: Paliwal, Kuldip; Kleijn, Willem, editores. Amsterdã, Holanda. Elsevier, 1995

TIEU, Kinh; **VIOLA**, Paul. Boosting Image Retrieval. 2000, Hilton Head Island. Proceedings of the IEEE CVRP. Hilton Head Island, South Carolina, USA, 2000.

ZAHORIAN, Stephen et al. Phone classification with segmental features and a binary-pair partitioned neural network classifier. 1997, Munique. **Proceedings of the IEEE ICASSP.** Munique, Alemanha, 1997, p1011-1014.

ZAHORIAN, Stephen; **NOSSAIR,** Zaki. A partitioned neural network approach for vowel classification using smoothed time/frequency features. **IEEE Transations on Speech and Audio Processing.** v.7,n.4, p.414-425, Jul 1999.

ZBYNIK, Tychtl; **PSUTKA,** Josef**.** Speech Production Based on the Mel-Frequency Cepstral Coefficients. 1999, Budapeste. **Proceedings of the EUROSPEECH'99.** Budapeste, Hungria, 1999, p2335-2338.