PAULO RODRIGO CAVALIN

UM MÉTODO PARA SEGMENTAÇÃO E RECONHECIMENTO DE PALAVRAS MANUSCRITAS USANDO MODELOS ESCONDIDOS DE MARKOV

Curitiba - PR

2005

PAULO RODRIGO CAVALIN

UM MÉTODO PARA SEGMENTAÇÃO E RECONHECIMENTO DE PALAVRAS MANUSCRITAS USANDO MODELOS ESCONDIDOS DE MARKOV

Dissertação apresentada ao curso de Mestrado em Informática Aplicada, da Pontifícia Universidade Católica do Paraná, como requisito parcial para a obtenção do título de Mestre em Ciências.

Área de concentração: Ciência da Imagem.

Orientador:

Prof. Dr. **Flávio Bortolozzi**

Co-orientadores:

Prof. Dr. **Alceu de Souza Britto Jr.**

Prof. Dr. **Robert Sabourin**

Pontifícia Universidade Católica do Paraná

Curitiba - PR

2005

AGRADECIMENTOS

Gostaria de agradecer primeiramente aos responsáveis pelo desenvolvimento técnico deste trabalho. Em especial meus orientadores, Dr. Flávio Bortolozzi, Dr. Alceu de Souza Britto Jr., e Dr. Robert Sabourin, que me forneceram todo o suporte necessário, participando e supervisionando durante todo o tempo em que necessitei. Gostaria de agradecer também ao auxílio fornecido por outros professores do PPGIA, em especial ao Dr. Jacques Facon, ao Dr. Luiz Eduardo Soares de Oliveira, e ao Dr. Alessandro Koerich. E ao professor Dr. Herman Martin Gomes, gostaria de salientar sua valiosa contribuição para o enriquecimento deste documento.

Gostaria também de agradecer aos amigos que conquistei durante o mestrado, entre eles Fausto Vanin, David Menoti, Islenho de Almeida, Fernanda Ramos, Éderson Sgarbi, Díbio Borges, e Cristiane e Willian Ferreira. Todos foram grandes amigos, em todos os momentos, e fica uma grande saudade dos tempos em que todos estavam em Curitiba.

À minha família, agradeço por todo o apoio durante esta fase da minha vida. Agradeço ao meu pai, Luiz Alberto Cavalin, e à minha mãe, Maria Cristina Jarema, que sempre me incentivaram nas minhas decisões. Agradeço também aos meus irmãos, Júnior, Ana Paula e Ana Luísa. Em especial, agradeço à minha esposa, Michelle, que também sempre me apoiou e teve compreensão nos momentos em que estive mais ocupado, além de me trazer toda a felicidade durante o período do mestrado, e à minha filha, que neste momento está a caminho deste mundo.

Fico muito agradecido à CAPES por ter me dado o suporte financeiro, sem o qual não seria possível toda a minha dedicação para a realização deste trabalho de mestrado.

RESUMO

Este trabalho apresenta um método de reconhecimento de palavras manuscritas, sem restrições de grafia, o qual é constituído de dois estágios. O primeiro estágio encontra as N melhores hipóteses de segmentação/reconhecimento utilizando segmentação implícita, programação dinâmica e auxiliada por léxico. Para cada hipótese este estágio fornece uma probabilidade e os pontos de segmentação que delimitam os caracteres dentro da palavra. No segundo estágio é feita a verificação das hipóteses geradas anteriormente. A segmentação implícita do primeiro estágio é realizada com base em informações do traçado da palavra, sendo utilizados dois conjuntos distintos de características. Ainda no primeiro estágio, as palavras são representadas pela concatenação dos Modelos Escondidos de Markov (MEMs) correspondentes aos caracteres. O treinamento é feito utilizando uma versão alterada do algoritmo de *Baum-Welch*. No segundo estágio, informações do traçado e do fundo das imagens, calculadas a partir de linhas e colunas, são combinados em MEMs representado cada caractere. As probabilidades calculadas no segundo estágio são combinadas com aquelas geradas no primeiro gerando-se assim uma lista final re-ordenada contendo as N melhores hipóteses de reconhecimento para uma dada palavra.

Para a avaliação do método, os experimentos foram realizados em imagens de palavras da base IAM. De um total de 18.624 imagens, 2.805 exemplos foram utilizados para os testes. Foram avaliados quatro tamanhos diferentes de léxico: 10, 100, 1000 e 3.717 palavras. Utilizando o conjunto de características do traçado no primeiro estágio, o sistema atingiu 78% de reconhecimento, com o maior léxico. Com a implementação utilizando as características da geometria, a taxa de reconhecimento foi de 71,51%. Com a combinação dos dois conjuntos de características, a taxa de reconhecimento foi de 77,68%. Nestes experimentos, a taxa de reconhecimento sem o estágio de verificação é em média 3,91% menor.

ABSTRACT

This work presents a two-stage method for unconstrained handwritten word recognition. The first stage finds the N best segmentation/recognition hypotheses based on implicit segmentation techniques, dinamic programming, and guided by a lexicon. For each hypothesis, this stage provides a probability and segmentation points which delimit the characters within the word image. The second stage verifies those hypothesis. The implicit segmentation at the first stage is performed based on foreground information, and two feature sets. In the first stage, the words are represented by Hidden Markov Models (HMMs) character concatenations. The training uses a modified version of *Baum-Welch* algorithm. In the second stage, background and foreground information of the words, computed from rows and columns, are combined to form HMMs representing each character. The probabilities estimated at second stage are combined with those ones computed by the first stage, and a final re-ranked list containing the N best recognition hypotheses for a given word.

The proposed method evaluation was done taking into account IAM database word images. The whole subset has 18,624 images, 2,805 are used in the test set. Four diferent lexicon were evaluated: 10, 100, 100, and 3,717 words. Using the background-foreground transitions feature set in the first stage, the system reaches 78% of recognition rate, for the larger lexicon. Using the geometric features in that stage, the recognition rate reached is 71.51%, and combining both feature sets the recognition rate is 77.68%. On those experiments, the average recognition rate lost without the verification stage is about 3.91%.

SUMÁRIO

LISTA DE TABELAS

LISTA DE FIGURAS

ABREVIAÇÕES

AC	Agrupamentos de Classificadores
BC	Borda Count
CC	Componentes Conectados
CEDAR	Center of Excellence in Document Analysis Recognition
Cid	Cidades
Col	Coluna
CTF	Características do Traçado e do Fundo
CG	Características Geométricas
CS	Classificação-Segmentação
CT	Características do Traçado
DB	Davies-Bouldin
ECT	Extração de Características do Traçado
ECTF	Extração de Características do Traçado e do Fundo
Est	Estágio
FP	Formas Parciais
IAM	Institute of Informatics and Applied Mathematics
IRONOFF	IRESTE On/Off
IP	Imagens Parciais
IV	Índices de Validação
Lin	Linha
LVQ	Learning Vector Quantization
MC	Múltiplos Classificadores
MEM	Modelos Escondidos de Markov
MLP	Multilayer Perceptron
MLPCM	Multilayer Perceptron Class-Modular

NIST	National Institute of Standarts and Technology
Pal	Palavras
QV	Quantização Vetorial
R	Reconhecimento
RN	Redes Neurais
XB	Xie-Beni

1 INTRODUÇÃO

O tratamento automático de documentos manuscritos tem recebido atenção especial em diversos centros de pesquisa no mundo todo. O motivo é a grande variedade de aplicações, tais como: processamento de cheques bancários, leitura automática do endereço em envelopes postais, processamento de formulários utilizados em recenseamentos, etc. Além disso, atualmente é clara a necessidade de uma maior interação entre a máquina e o homem, pois não apenas o homem deve adaptar-se às linguagens entendidas pelo computador, mas o computador deve estar apto a reconhecer códigos produzidos pelo homem.

1.1 DEFINIÇÃO DO PROBLEMA

Neste contexto, uma tarefa desafiadora é o reconhecimento de palavras manuscritas, onde são fatores de complexidade:

- **Variações de estilos**: palavras manuscritas podem possuir estilos variados, conforme o escritor. A Figura 1.1 ilustra exemplos de diferentes estilos de escrita que ocorrem com a língua inglesa e outras línguas ocidentais. Em alguns estilos a segmentação dá-se de forma natural, enquanto em outros, algoritmos avançados são necessários para esta tarefa. Para modelar o reconhecimento, a utilização de modelos diferentes para representar caracteres maiúsculos e minúsculos é uma alternativa para a redução de variações na forma dos caracteres, assim como o treinamento pode absorver outras diferenças.

- **Variação de tamanho**: pode ocorrer tanto entre palavras diferentes, como entre caracteres da mesma palavra. Na tentativa de amenizar este problema, diferentes abordagens de normalização têm sido propostas. Além disso, alguns autores têm buscado a definição de características invariantes a este fator, ou ainda a absorção

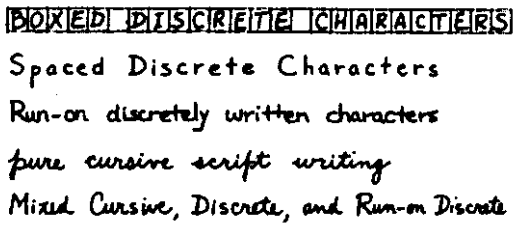
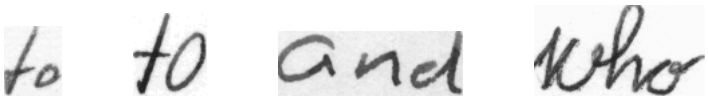


Figura 1.1: Ilustração de diferentes tipos de escrita ([TAPPERT et al, 1990]).

desta variação pelos modelos utilizados no reconhecimento [BRITTO, 2000]. A Figura 1.2 mostra exemplos da variação de tamanho entre palavras e entre caracteres de uma mesma palavra.



(a) (b) (c) (d)

Figura 1.2: Exemplos de variações de tamanho. Em (a) e (b) entre palavras, e em (c) e (d) na mesma palavra.

- **Inclinação vertical dos caracteres**: ou *slant*, pode ser considerada como uma característica do escritor em alguns métodos para identificação de autoria, mas em sistemas de reconhecimento pode representar um problema para o método de extração de características ou para métodos de reconhecimento baseados na segmentação explícita das palavras em caracteres. Métodos para tratar este problema estão descritos em [BOZINOVIC & SRIHARI, 1989], [KIMURA et al, 1993] e [GUILLEVIC, 1995]. A Figura 1.3 mostra uma palavra com caracteres inclinados.

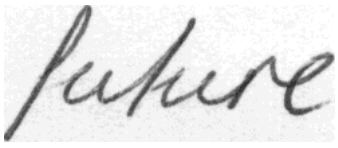


Figura 1.3: Exemplo de inclinação vertical dos caracteres.

- **Inclinação da linha de base**: ou *skew*, pode significar um problema para o método de extração de características, principalmente em sistemas onde uma *slide-window* é usada para o cálculo das características. Alguns métodos para a correção deste problema foram propostos [MORITA, 1998] [EL-YACOUBI, 1996]. Uma palavra com este tipo de inclinação pode ser vista na Figura 1.4.

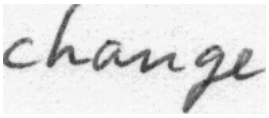
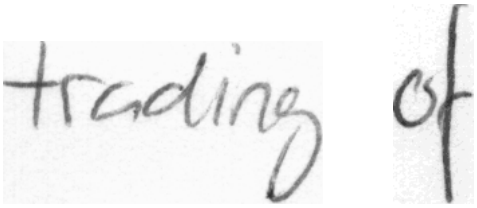


Figura 1.4: Exemplo de palavra com inclinação na linha de base.

- **Caracteres fragmentados**: a ocorrência de caracteres fragmentados pode ser resultado de uma falha de captura, ou mesmo do estilo da escrita, quando o caracter não possui uma continuidade ou perfeição na sua forma. Exemplos podem ser vistos na Figura 1.5.



(a) (b)

Figura 1.5: Exemplo de caracteres fragmentados. (a) mostra um “a” fragmentado, e (b) mostra um “o” fragmentado.

- **Variações dependentes do contexto**: alterações podem ocorrer na forma do caracter devido à sua posição na palavra. É um dos grandes problemas em sistemas de reconhecimento de palavras manuscritas, já que há alterações nas características do caracter. A Figura 1.6 mostra algumas alterações deste tipo.

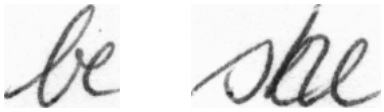


Figura 1.6: Exemplo de alterações de contexto. O caracter “e” possui alterações conforme o seu caracter antecessor.

- **Tamanho do léxico**: o tamanho do léxico tem um impacto direto no desempenho do sistema, conforme demonstrado em [KOERICH et al, 2003a]. Métodos para a redução do léxico, como a incorporação de restrições de contexto ao sistema, auxiliam na redução do espaço de busca e na possibilidade de erros.

1.2 MOTIVAÇÃO

O reconhecimento de palavras manuscritas tem sido objeto de estudo em diversos centros de pesquisa nas últimas décadas. O principal motivo é o grande número de aplicações e os benefícios que um sistema deste porte pode trazer para processos, como: triagem postal, processamento de cheques bancários, leitura de formulários em geral, indexação de documentos, etc... Além disso, o reconhecimento de manuscritos pode viabilizar interfaces mais amigáveis com o computador seja de maneira on-line (exemplo, tablet PC), ou off-line (exemplo, document imaging).

Além da motivação em termos de aplicação, os métodos existentes na literatura demonstram que o reconhecimento de palavras manuscritas ainda necessita de otimizações. Estas otimizações justificam-se tanto para aumentar o desempenho dos métodos em termos de confiabilidade, como em tempo de execução.

Cientificamente, a motivação está na avaliação de um método de segmentação implícita combinado com estratégia de verificação a posteriori com o objetivo de tratar os diferentes fatores de complexidade inerentes às palavras manuscritas.

1.3 OBJETIVOS

Este trabalho tem como objetivo o desenvolvimento de um método para o reconhecimento de palavras manuscritas, sem restrições de grafia, considerando-se os fatores de complexidade supracitados. Para tal, torna-se necessário cumprir as seguintes etapas:

1. Levantamento bibliográfico sobre os métodos de reconhecimento de caracteres, cadeias numéricas e palavras manuscritas. Com esse aprofundamento é possível conhecer os métodos já existentes, assim como, avaliar o desempenho do método proposto.

2. Estudo e adaptação do método proposto em [BRITTO, 2001], o qual originalmente foi desenvolvido para o reconhecimento de cadeias numéricas. Nesta etapa incluem-se os algoritmos necessários à modelagem dos caracteres a serem utilizados pelo método, assim como a estratégia para o reconhecimento de palavras.

3. Definição do pré-processamento necessário. Como o método original foi proposto para o reconhecimento de cadeias numéricas, o reconhecimento de palavras pode necessitar de outros tipos de pré-processamentos não utilizados no trabalho original.

4. Otimização dos parâmetros que melhor se adaptam a este problema, visando melhorar as taxas de reconhecimento.

5. Implementação de um conjunto de características alternativo para o estágio de segmentação/reconhecimento.

6. Avaliação de diferentes estratégias de combinação de conjuntos de características visando melhorar a segmentação/reconhecimento.

7. Avaliação da contribuição de um estágio de verificação dos resultados obtidos com a segmentação implícita.

1.4 MÉTODO PROPOSTO

Os fatores de complexidade citados anteriormente podem tornar difícil a idéia básica de segmentar uma palavra em caracteres, a fim de reconhecê-los de maneira isolada. Desta forma, dentre as diferentes abordagens propostas na literatura com o objetivo de resolver os problemas envolvidos no reconhecimento de palavras manuscritas, a execução de segmentação e reconhecimento em um único processo tem mostrado resultados interessantes [BUNKE & MARTI, 2000]. Nesta direção, o método proposto é constituído de dois estágios. A Figura 1.7 permite uma visão geral do método, o qual terá seus módulos detalhados no Capítulo 4.

No primeiro estágio, após o pré-processamento da palavra com o objetivo de corrigir problemas como inclinações e diferenças de tamanhos, ocorre a extração de características baseadas em informação do traçado da palavra, extraída de cada coluna da imagem. Estas características são utilizadas para avaliar MEMs representando palavras que são formadas pela concatenação de MEMs que representam cada classe de caractere maiúsculo e minúsculo. Como resultado deste estágio obtêm-se os pontos de segmentação e um *ranking* contendo as N melhores hipóteses de segmentação/reconhecimento para a palavra processada. Isto se dá através do módulo de segmentação e reconhecimento.

As N hipóteses geradas no estágio anterior são verificadas no segundo estágio. Para tal, os pontos de segmentação gerados serão utilizados para delimitar as regiões de cada caractere presente na palavra. O objetivo é extrair novas características com base em informações do fundo da imagem (concavidades). Novos modelos de Markov baseados nas linhas dos caracteres são combinados a novos modelos de coluna para a verificação das N hipóteses sugeridas na etapa anterior.

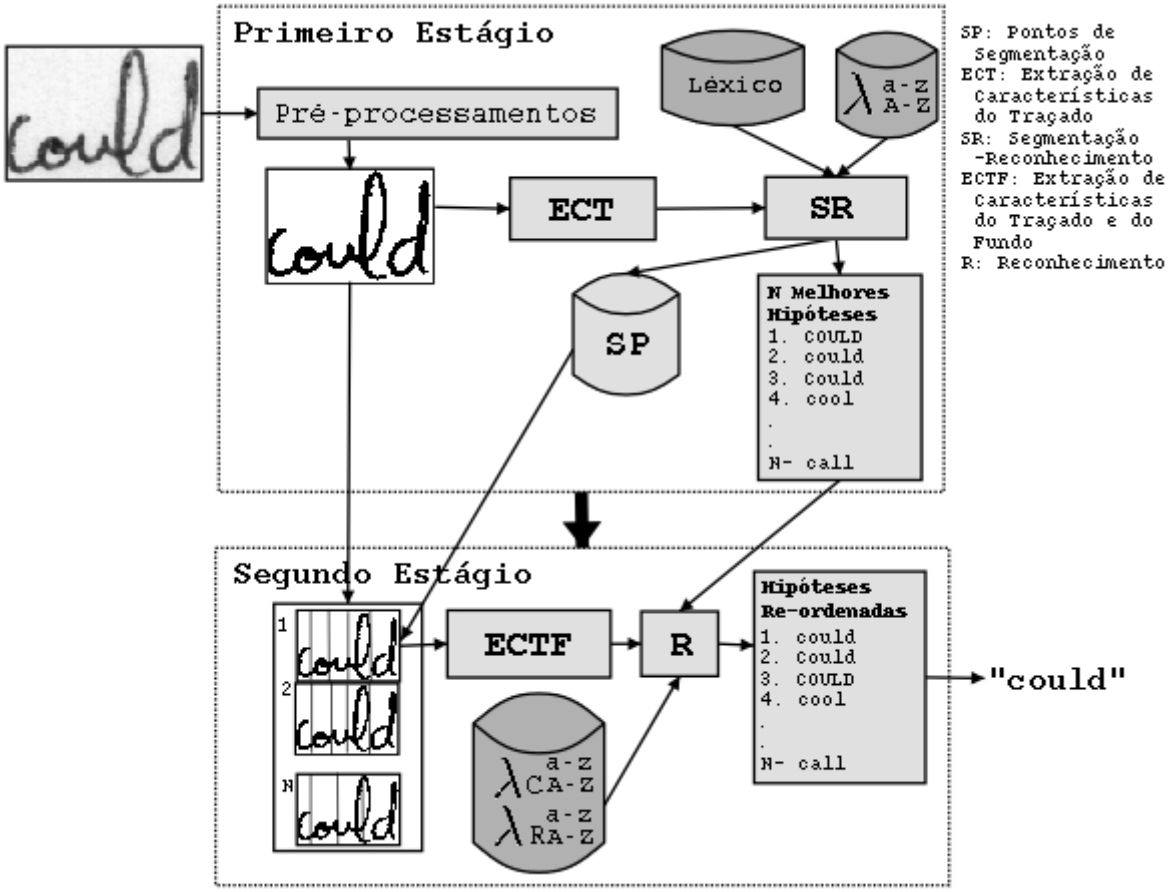


Figura 1.7: Visão geral do método proposto

1.5 CONTRIBUIÇÕES

Dentre as contribuições deste trabalho está um sistema para reconhecimento de palavras manuscritas podendo ser aplicado em diferentes escopos. Dentre eles, o processamento de cheques bancários e a leitura automática de endereços postais.

Cientificamente, o trabalho apresenta a avaliação de um sistema de reconhecimento de palavras manuscritas em dois estágios. Neste trabalho é possível avaliar a segmentação implícita de palavras manuscritas, assim como o impacto de uma fase de verificação para compensar as possíveis perdas oriundas da combinação de segmentação

e reconhecimento em um mesmo processo.

O método avaliado passou por diversos ajustes até o seu resultado final. Dentre estes ajustes, podemos citar:

- Implementação do léxico em árvore. Com isto, o tempo de processamento médio foi reduzido em cerca de 80%;
- Avaliação de métodos de pré-processamentos, para correção de imperfeições na grafia das palavras da base de dados. Foram avaliados métodos de correção na inclinação vertical dos caracteres, para a correção da inclinação da linha de base, e para a normalização da altura dos caracteres;
- Otimização do tamanho do *codebook* para o problema proposto. Foram avaliadas duas estratégias: supervisionada e não-supervisionada.
- Avaliação da inclusão de dados de outras bases de dados para aprimorar o treinamento dos MEM. Caracteres isolados da base IRONOFF foram adicionados às bases de treinamento, tanto no primeiro como no segundo estágio;
- Avaliação de diferentes alternativas para estimar o número de estados dos MEMs. O mesmo método foi aplicado para a computação do número de estados, porém diferentes dados foram avaliados para isto;
- Avaliação de dois diferentes conjuntos de características para o primeiro estágio. Um conjunto de características alternativo foi avaliado para a tarefa de segmentação/reconhecimento;
- Avaliação da combinação dos dois conjuntos de características propostos para o primeiro estágio. Os dois conjuntos de características foram combinados utilizando métodos de combinação de classificadores existentes na literatura, e novos métodos foram propostos e avaliados;

- Criação de uma base de caracteres isolados a partir da segmentação automática da base de palavras. O estágio de segmentação/reconhecimento foi utilizado para criar de maneira semi-automática uma base de dados de caracteres isolados a partir das palavras da base IAM;

- Proposta de um limiar para selecionar as palavras a serem verificadas. Um limiar para selecionar as palavras a serem verificados foi proposto e avaliado.

1.6 ESTRUTURA DO TRABALHO

Além destra introdução, constam neste trabalho mais cinco capítulos. O Capítulo 2 traz uma revisão bibliográfica sobre alguns trabalhos da área de reconhecimento de manuscritos, com destaque para publicações de reconhecimento de palavras manuscritas, e trabalhos relacionados diretamente ao método proposto, como referências para o protocolo experimental. A fundamentação teórica, revisando os principais componentes e classificações de MEMs, assim como algoritmos para treiná-los e avaliá-los, estão descritos no Capítulo 3. Este capítulo também apresenta outras teorias utilizadas neste trabalho.

A descrição detalhada do método proposto é apresentada no Capítulo 4. Neste, passo-a-passo, os componentes do sistema são explorados, mostrando todos os detalhes sobre seu funcionamento, dados de entrada e dados de saída. Os experimentos realizados e as conclusões obtidas a partir destes estão descritos no Capítulo 5 e no Capítulo 6, respectivamente.

2 ESTADO DA ARTE

O objetivo deste capítulo é abordar os principais trabalhos relacionados ao método proposto. Com isso, é possível visualizar os métodos apresentados para a solução dos principais sub-problemas relacionados a esta área da pesquisa, assim como torna-se possível a comparação dos resultados apresentados neste trabalho.

Serão abordados na Seção 2.1 os principais conceitos e a situação atual do reconhecimento de caracteres isolados. A Seção 2.2 traz um resumo sobre o estado atual do reconhecimento de cadeias numéricas, enquanto na Seção 2.3 são abordados os métodos para reconhecimento de palavras. Para finalizar este capítulo, a Seção 2.4 traz algumas conclusões sobre os métodos aqui descritos.

2.1 RECONHECIMENTO DE CARACTERES ISOLADOS

O reconhecimento de caracteres isolados é uma etapa muito importante tanto para alguns sistemas de reconhecimento de cadeias numéricas como para alguns sistemas de reconhecimento de palavras. Isto se deve ao fato de muitos métodos modelarem as cadeias ou palavras como agrupamento de caracteres, os quais são reconhecidos após algum tipo de segmentação.

Dígitos e letras isolados podem ser considerados como sendo um mesmo problema, porém, considera-se que dígitos possuem fatores de complexidades menores que letras. Dentre estes, podemos citar o menor número de classes de dígitos, o que faz com que a confusão no reconhecimento seja menor. Já letras alfabéticas podem possuir diferentes padrões para representar um único caracter, representações cursivas das letras, entre outros fatores de complexidades.

Com um menor número de fatores de complexidade, hoje em dia os sistemas de reconhecimento de dígitos atingem altas taxas de reconhecimento, se comparados a sistemas de reconhecimento de letras. [BRITTO et al., 2004] consegue 8% a mais de taxa de

reconhecimento de dígitos isolados, se comparado ao reconhecimento de letras, utilizando as mesmas características e o mesmo método de classificação.

Para um sistema de reconhecimento de caracteres, dois aspectos principais devem ser considerados: a extração de características e o método de classificação. Um conjunto de características discriminantes é considerado o fator mais importante para atingir altas taxas de reconhecimento. Um importante trabalho sobre características para reconhecimento de caracteres foi apresentado em [TRIER et al, 1995]. Neste, o autor apresenta um estudo sobre métodos de extração de características para o reconhecimento de caracteres segmentados, descrevendo importantes aspectos que devem ser considerados ao selecionar o método de extração de características.

Os métodos para extração de características apresentados na literatura abordam basicamente dois caminhos: estatístico e estrutural. As primitivas estatísticas são derivadas de distribuições de pontos, como zoneamento, momentos, histogramas de projeções ou histogramas de direções [KIMURA & SHIDHAR, 1992] [CHEUNG & YEUNG, 1998]. Características estruturais são baseadas em propriedades geométricas e topológicas do caracter, como concavidades e suas direções, pontos finais ou intercessões de segmentos, e laços [CAI & LIU, 1998] [HIRANO et al, 1997]. Em alguns trabalhos, a integração de características estatísticas e estruturais são aplicadas, considerando que estes tipos de primitivas são complementares [CAI & LIU, 1998] [HEUTE et al, 1998].

Para a classificação, diferentes tipos de classificadores têm sido utilizados, como por exemplo, classificadores estatísticos [CHEUNG & YEUNG, 1998] e redes neurais [KOERICH et al, 2003c]. Recentemente, o uso de diferentes estratégias de combinação [SUEN et al, 1999], e o uso de *Support Vector Machines* [OLIVEIRA & SABOURIN, 2004] tem produzido melhorias às taxas de reconhecimento.

2.2 RECONHECIMENTO DE CADEIAS NUMÉRICAS

O número de aplicações potenciais, como reconhecimento do CEP e de cheques bancários, tem dado ao reconhecimento de cadeias numéricas atenção especial em diversos centros de pesquisas. Contudo, ainda é um desafio o reconhecimento de cadeias numéricas de tamanho desconhecido, assim como fatores de complexidade como dígitos conectados.

Podemos destacar dois tipos de sistemas de reconhecimento de cadeias de dígitos. O primeiro tipo é quando a cadeia numérica é reconhecida através de um método global, e o outro ocorre quando é reconhecida de maneira analítica (ou local). O primeiro é pouco utilizado devido às suas limitações ao pequeno número de aplicações. Já o último necessita de algum tipo de segmentação, extraindo caracteres ou algum tipo de sub-unidade, tornando-se mais complexo. O reconhecimento baseado em segmentação é mais robusto, porém, mais caro computacionalmente. Neste tipo de sistema ainda podem ser encontradas duas sub-divisões: segmentação explícita, onde regras de corte são utilizadas; e segmentação implícita, onde a segmentação é feita em coluna, geralmente por um classificador.

Em [OLIVEIRA et al, 2002] é apresentado um sistema para reconhecimento de cadeias numéricas utilizando uma abordagem de reconhecimento baseada em segmentação, e uma estratégia de reconhecimento e verificação (Figura 2.1). Utilizando um modelo probabilístico, o método combina as saídas dos diferentes níveis, como a segmentação, o reconhecimento, e o pós-processamento. O trabalho também propõe um novo esquema de verificação para tratar os problemas de excesso e falta de componentes causados pela segmentação, e um novo conjunto de características é proposto para lidar os problemas de excesso de componentes. Um pós-processador baseado em autômatos determinísticos é utilizado e o módulo de decisão global toma a decisão de aceitar ou rejeitar o reconhecimento. Experimentos em duas base de dados diferentes foram realizados.

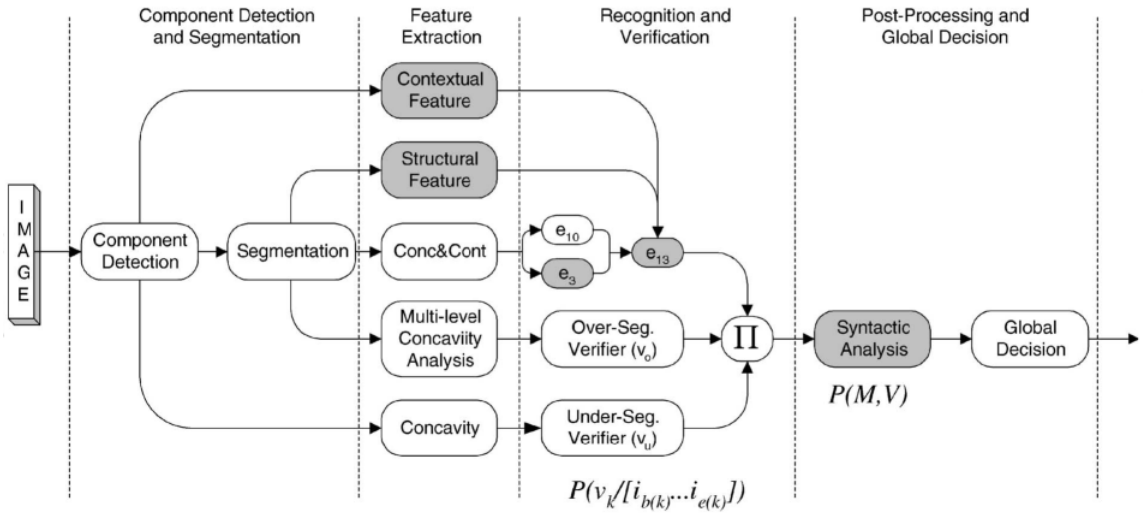


Figura 2.1: Visão geral do método proposto em [OLIVEIRA et al, 2002].

Um sistema utilizando segmentação implícita, com uma abordagem baseada em dois estágios para reconhecimento de cadeias numéricas manuscritas, é apresentado em [BRITTO, 2001]. No primeiro estágio, dez MEMs representando as classes de dígitos manuscritos são concatenados utilizando-se um processo dinâmico baseado no algoritmo de construção de níveis (*Level Building*). As características extraídas nesta primeira fase levam em conta apenas informações do traço dos dígitos, extraídas de cada coluna da imagem da cadeia numérica. Segundo o autor, neste estágio há uma perda em termos de reconhecimento, uma vez que as características e modelos utilizados buscam contemplar a segmentação e reconhecimento em um único processo. Os pontos de segmentação e o resultado de reconhecimento da cadeia fornecidos pelo primeiro estágio são utilizados em um procedimento de verificação (segundo estágio). Neste, os pontos de segmentação são utilizados para definição dos limites de cada dígito na cadeia, o que torna possível a extração de novas características baseadas no fundo da imagem. Estas novas características são combinadas com as anteriores em outros vinte MEMs representando colunas e linhas. Desta forma, neste segundo estágio o autor combina informações complemen-

tares como características de *foreground* e *background* extraídas das colunas e linhas da imagem dos dígitos reconhecidos no primeiro estágio. Com base na verificação os dez melhores resultados do primeiro estágio são reordenados. A Figura 2.2 traz uma visão geral sobre o funcionamento deste método.

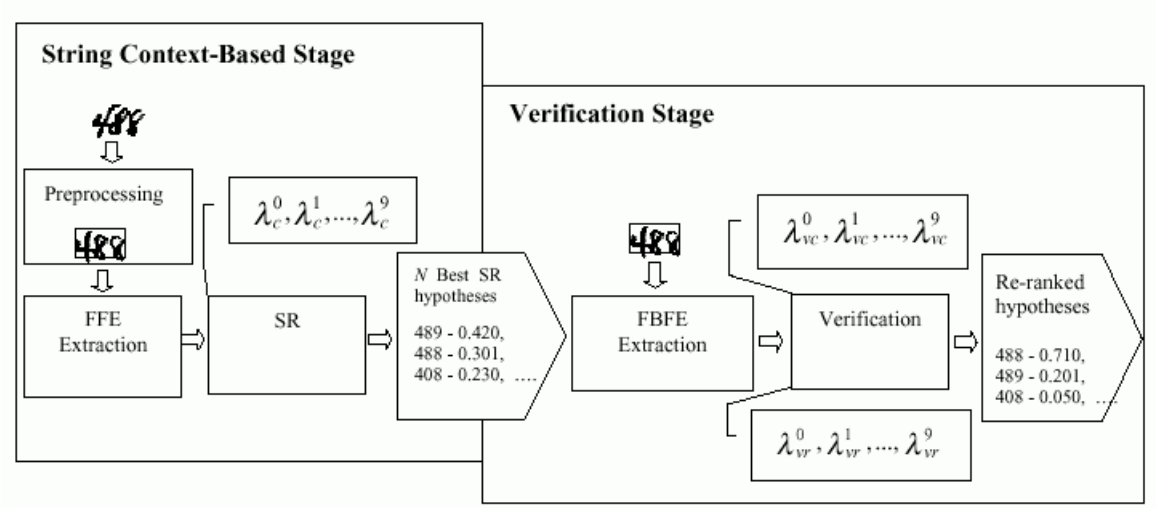


Figura 2.2: Visão geral do método proposto em [BRITTO, 2001].

Outro método interessante foi o proposto em [BUNKE & ZIMMERMANN, 1998], onde apresenta-se um método em dois estágios, desta vez para reconhecimento de cadeias numéricas manuscritas. O sistema combina um método de reconhecimento baseado em segmentação com outro livre de segmentação. Desta forma, o objetivo é que um método possa compensar a deficiência do outro. Um módulo global de decisão combina os resultados de ambos os métodos. Para o método baseado em segmentação explícita um algoritmo busca os componentes conectados (CC) na imagem a fim de segmentar a cadeia numérica em imagens parciais (IP). Os CC muito pequenos são eliminados através de filtragem, e uma fase de agrupamento junta os CC quebrados. As IP são analisadas pelo módulo de detecção de dígitos e reconhecidas se contiverem dígitos isolados. Se a IP for um grupo de dígitos, esta é enviada para o método

livre de segmentação. Este tem como finalidade o reconhecimento de IP contendo grupos de dígitos ou dígitos fragmentados não detectados na fase anterior. É dividido em dois níveis, um é responsável por dividir a IP em formas parciais (FP), e o outro analisa as melhores combinações das FP extraídas no nível anterior. Introduz-se aqui o conceito de *dummy symbol* para manipular ruídos que não são eliminados pelos principais algoritmos de filtragem. Alguns experimentos foram realizados utilizando imagens da base de dados CEDAR e NIST SD3. Na base CEDAR, utilizando 18.468 imagens para treinamento e 495 imagens para teste, o sistema atingiu 83,6% de reconhecimento com nível zero de rejeição. Já na outra base, treinando o sistema com 50.000 imagens e testando com 4.925 imagens, o sistema atingiu a taxa de 92,7% de reconhecimento (nível zero de rejeição).

2.3 RECONHECIMENTOS DE PALAVRAS MANUSCRITAS

Sistemas de reconhecimento de palavras manuscritas possuem algumas diferenças em relação aos sistemas de reconhecimento de cadeias numéricas. Uma destas é a necessidade de um léxico, que é o vocabulário (classes de palavras existentes) da aplicação, e este é determinado pela aplicação em que será dirigido.

Relacionada ao léxico, duas classificações podem ser consideradas para sistemas que reconhecem palavras manuscritas [STEINHERZ et al, 1998]. A primeira delas é a global, onde a palavra é considerada como uma unidade. E a segunda, a local, onde a palavra é considerada como possuindo sub-unidades. A relação com o léxico dá-se pelo fato de que geralmente apenas em aplicações onde o tamanho do léxico é pequeno a abordagem global é utilizada. Em um sistema onde o léxico é maior, a utilização da abordagem local é muito utilizada pois permite o compartilhamento de dados para o treinamento.

Assim como em sistemas de reconhecimento de cadeias numéricas baseados em segmentação, os sistemas de reconhecimento de palavras manuscritas com abordagem local necessitam de algum mecanismo de segmentação. E assim como os primeiros, esta

pode ser explícita ou implícita (com a ajuda de um classificador).

Um classificador muito utilizado neste tipo de sistema são os Modelos Escondidos de Markov. Este tem sido aplicado devido ao seu bom desempenho para a segmentação implícita, além de possuir boas taxas de reconhecimento. Em [CONNELL, 1996], o autor faz comparações de desempenho de três técnicas diferentes de reconhecimento (*Learning Vector Quantization* (LVQ), MEM e *Multilayer Perceptor* (MLP)). Os experimentos realizados em caracteres previamente segmentados mostraram que MEM conseguiu melhores resultados que LVQ, e ficou próximo, embora abaixo, dos resultados apresentados por MLP. Porém, são mostradas avaliações sobre a capacidade de segmentação de MEM, comparando resultados deste com resultados de um método de projeção e segmentações manuais, e os MEM conseguiram um ótimo desempenho para esta tarefa.

Uma abordagem baseada em MEMs para o reconhecimento de palavras manuscritas foi proposta em [BUNKE et al., 1995]. Nesta, cada canto, extraído do esqueleto da palavra, é mapeado em um vetor de 10 características. O algoritmo de extração de características busca sempre fazer a extração na mesma ordem para cada exemplo. Os modelos das palavras são construídos a partir da concatenação dos modelos das letras, que são compartilhados para todas as letras. Os experimentos, que foram realizados com 12000 palavras, utilizando um léxico de 150 palavras escolhido aleatoriamente, alcançaram taxas de 98% de reconhecimento numa base de dados obtida com formulários que diminuíram a ocorrência de certas irregularidades nas palavras.

El-Yacoubi [EL-YACOUBI et al., 1999] propos um sistema contendo um léxico, para reconhecimento *off-line* de palavras manuscritas, utilizando segmentação explícita das palavras em caracteres ou pseudo-caracteres. Neste sistema são extraídos dois vetores de características (um contendo primitivas globais e outro informações obtidas de uma da análise do histograma do contorno bi-direcional de transições nas direções verticais e horizontais). Para a modelagem das palavras são concatenados MEMs com oito

estados representado caracteres. Uma representação da palavra em grafos permite que se considerem letras maiúsculas e/ou minúsculas em sua formação, sendo que na fase de reconhecimento este grafo será percorrido. Os experimentos foram realizados com uma base de dados de nomes de cidades francesas, e com um léxico de 1000 possíveis palavras. Os melhores resultados desta abordagem atingiram 88,9% de taxa de reconhecimento.

Em [BUNKE & MARTI, 2000] apresenta-se um sistema livre de segmentação para reconhecimento de palavras manuscritas, onde linhas completas de texto (sentenças) são a unidade básica para o reconhecedor. No pré-processamento os textos manuscritos são segmentados em linhas, e nestas são feitas normalizações, correções e outras operações. Em seguida são extraídas nove características geométricas em cada coluna da imagem. As três primeiras características trazem descrições globais da distribuição dos pontos da palavra, enquanto que as outras seis trazem detalhes sobre a escrita. Na fase de reconhecimento os MEMs contínuos, representando os caracteres, são concatenados para formar as palavras, e estas são concatenadas para formar as sentenças (Figura 2.3). Os modelos são treinados utilizando o algoritmo de *Baum-Welch*, e o reconhecimento utiliza o algoritmo de *Viterbi*. Neste, as palavras são balanceadas conforme a ocorrência na base. Vários experimentos foram realizados, sendo que a maior taxa de reconhecimento foi 79,50%, com um vocabulário de 412 palavras e 4.523 imagens. Com um grande vocabulário, possuindo 7.719 palavras, e 44.019 imagens, atingiu-se taxa de 60,05%. A taxa média de reconhecimento ficou em 60,48%. Em todos os experimentos, 80% das imagens foram utilizadas para treinamento, e 20% para teste.

Uma continuação do trabalho acima é descrito em [BUNKE & MARTI, 2001]. Neste as linhas de texto são segmentadas explicitamente no pré-processamento. O algoritmo de segmentação busca componentes conectados, e a distância entre dois destes é computada. Se a distância for maior que um dado limiar, estes são segmentados. As fases posteriores ao pré-processamento são semelhantes aos de [BUNKE & MARTI, 2000], diferenciando-se apenas na fase de reconhecimento onde

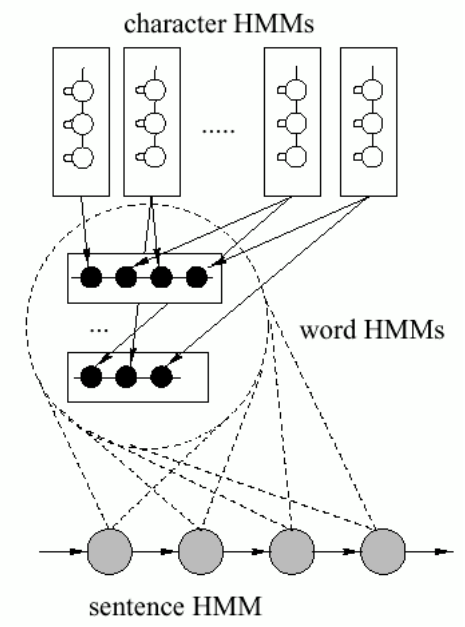


Figura 2.3: Esquema de concatenação dos MEM em [BUNKE & MARTI, 2000].

necessita-se apenas concatenar os MEMs para formar as palavras, não mais precisando formar as sentenças. Os experimentos foram realizados com 3.899 imagens em um vocabulário de 412 palavras. Considerando o valor de limiar que obteve melhores resultados na segmentação, este sistema obteve 77,2% de palavras corretamente reconhecidas, sendo que foram consideradas apenas palavras perfeitamente segmentadas (95,56% das palavras foram segmentadas corretamente), e 73,45% considerando os erros de segmentação.

Em [BUNKE & ZIMMERMANN, 2002a] descreve-se um sistema correspondente a [BUNKE & MARTI, 2000], adaptado para reconhecimento de palavras isoladas. O objetivo é analisar diferentes maneiras de otimizar o número de estados dos MEMs: modelagem de tamanho fixo, onde todos os modelos possuem o mesmo tamanho (número de estados); tamanho de *Bakis*, onde o número de estados por modelo é uma fração da média do número de observações de exemplos correspondentes no conjunto de treinamento; e a modelagem *quantile*, onde o tamanho de cada MEM é dado pela *quantile* especificada do

histograma correspondente de cada caracter. Experimentos foram realizados com 10.929 imagens, sendo que destas 1.000 são escolhidas aleatoriamente para o conjunto de teste, utilizando um vocabulário de 2.296 palavras. Considerando os parâmetros ótimos de cada método de modelagem, a modelagem *quantile* atingiu a maior taxa de reconhecimento (69,6%). O tamanho de *Bakis* atingiu 69,2% com seu melhor parâmetro, e com tamanho fixo, atingiu-se 61% com dezesseis estados.

Um sistema utilizando segmentação explícita para o reconhecimento de palavras manuscritas cursivas foi proposto em [ARICA & YARMAN-VURAL, 2002]. Neste o autor propõe um esquema analítico, utilizando uma seqüência de algoritmos de segmentação e reconhecimento. Ver Figura 2.4 para mais detalhes. Em primeiro lugar, alguns parâmetros globais, como o ângulo da inclinação vertical, linhas de bases, e a largura e a altura das concavidades são estimadas. Posteriormente, um método de segmentação encontra os caminhos de segmentação combinando informações binárias e em níveis de cinza. Por último, MEMs são utilizados pelo reconhecimento para classificar os caracteres candidatos. A estimação do espaço de características e as classificações dos MEMs são combinados num problema de otimização de grafos para o reconhecimento em nível de palavras. O desempenho do sistema foi testado em 2.000 palavras da base de dados Lancester-Oslo/Berges, o qual contém manuscritos de um único autor. O método obteve 90,8% de reconhecimento utilizando um léxico de mil palavras.

As Redes Neurais (RN) também são classificadores muito utilizados em sistemas com abordagem local [OLIVEIRA JR. et al, 2002], ou em sistemas em dois estágios [KIM et al, 2000]. Em [KAPP et al, 2004], o autor apresenta a comparação entre uma rede MLP convencional e uma rede MLP classe-modular (MLPCM) num sistema limitado ao reconhecimento de doze classes (os meses do ano). São utilizadas características perceptivas e características baseadas em concavidades/convexidades, e utilizando um esquema de zoneamento para torná-las mais robustas. Os resultados apresentados pela MLPCM foram melhores que a MLP convencional, já que a taxas de reconhecimento para a primeira foi

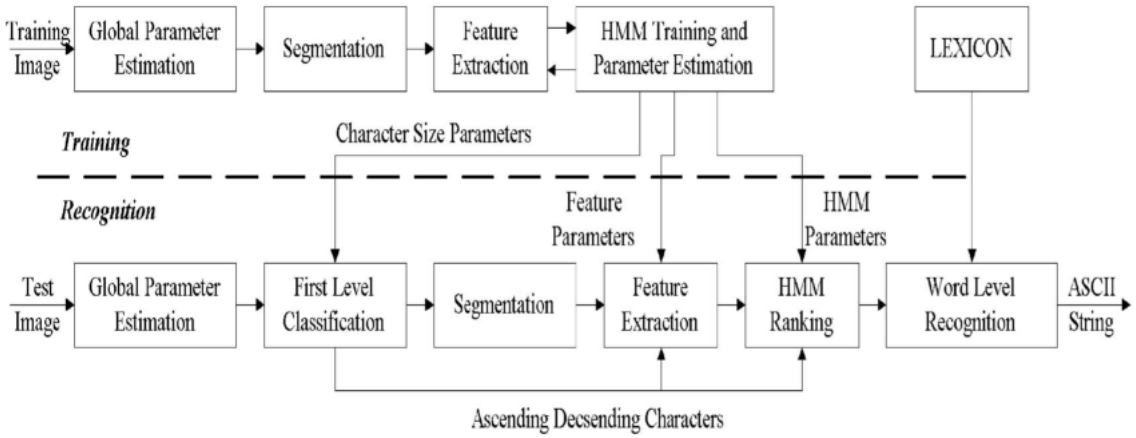


Figura 2.4: Visão geral do método proposto em [ARICA & YARMAN-VURAL, 2002].

de 81,75%, enquanto que a primeira atinge 77,01%. Este trabalho também apresenta um mecanismo de rejeição com múltiplos limiares, e os experimentos realizados mostraram que este tipo de mecanismo pode ser aplicado em ambas arquiteturas de MLP.

Outro método utilizando RN foi mostrado [KOERICH et al, 2002]. Neste o autor propõe um sistema híbrido, integrando MEM e RN (Figura 2.5). Os dados de entrada são primeiro processados por um reconhecedor baseado em MEM, auxiliado por um léxico, onde são geradas as listas com as N melhores hipóteses assim como os pontos de segmentação para cada uma destas. Uma RN é utilizada para avaliar cada caracter segmentado, e as saídas dos MEM e das RN são combinadas para otimizar o desempenho. Utilizando um léxico de 80.000 palavras, os experimentos mostraram um aumento de 10% na taxa de reconhecimento utilizando a verificação com RN.

Entre os trabalhos mais recentes, podemos citar os trabalhos de Günter e Bunke, em [GÜNTER, 2004a] e [GÜNTER, 2004b]. No último, os autores combinam três classificadores com diferentes arquiteturas para o reconhecimento de palavras manuscritas. Além disso, um novo método de agrupamento trabalhando com alguns classificadores é aplicado e os resultados do agrupamento são comparados com os resultados da

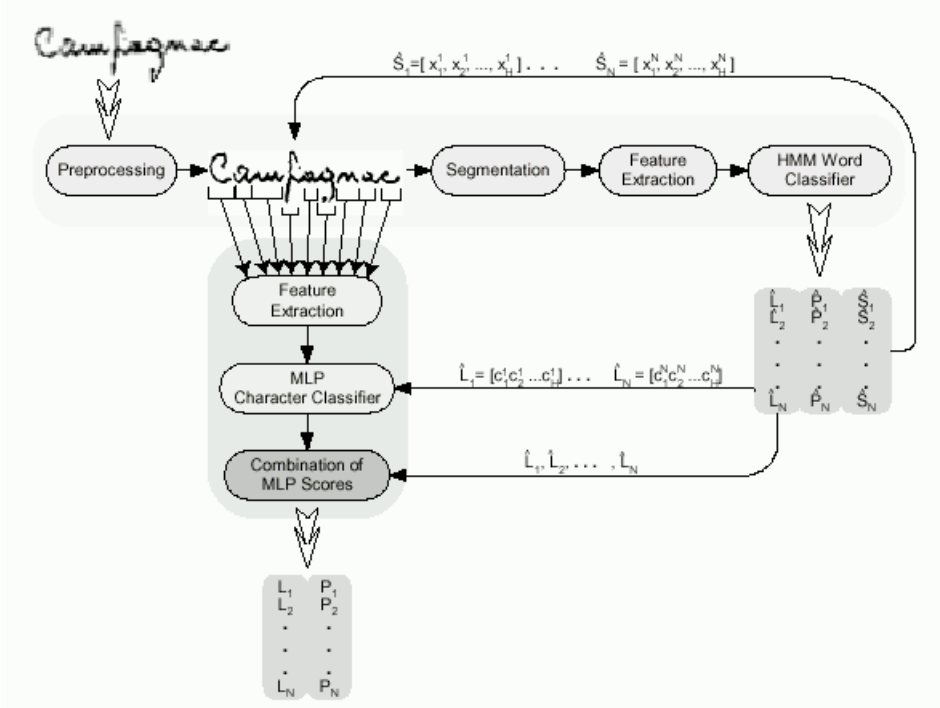


Figura 2.5: Visão geral do método proposto em [KOERICH et al, 2002].

combinação dos três classificadores. Os experimentos mostram que um aumento de 2,98% é obtido com o melhor esquema de combinação. Os autores também demonstram que método clássicos de criação de conjuntos de classificadores, como *Bagging* e *AdaBoost*, podem também aumentar o desempenho. Entretanto, o maior aumento de desempenho foi obtido através de um novo método proposto pelos autores. O desempenho deste foi 1,5% melhor a melhor combinação de classificadores, 2,94% maior que os métodos clássicos de agrupamento, e 4,48% maior que o melhor classificador isolado.

Para finalizar esta seção, um resumo do desempenho de algumas das aplicações recentes encontradas na literatura é apresentado na Tabela 2.1. Este resumo é útil para ilustrar o estado atual dos sistemas de reconhecimento de palavras. Porém, é difícil fazer comparações entre os resultados destes métodos, já que os experimentos utilizam bases de dados diferentes, diferentes classes de palavras, e diferentes números

de exemplos para o teste. Os resultados são dependentes tanto do tamanho do léxico, como das ambigüidades entre as classes de palavras e do sistema ser dependente ou não do escritor.

Tabela 2.1: Resumo do desempenho de alguns métodos para reconhecimento de palavras manuscritas. *: O sistema utiliza algum tipo de verificação.

Método	**Classif.**	**Léx. (#)**	**Im. Teste**	**Des. (%)**	**Base**
[BUNKE et al., 1995]	MEM	150	3.000	98,40	Pal.
[BUNKE & MARTI, 2000]	MEM	7.719	8.803	60,05	Pal.
[BUNKE & MARTI, 2001]	MEM	412	780	77,02	Pal.
[BUNKE & ZIMMERMANN, 2002a]	MEM	2.296	1.000	69,60	Pal.
[EL-YACOUBI et al., 1999]	MEM	1.000	4.313	88,90	Cid.
[ARICA & YARMAN-VURAL, 2002]*	MEM	1.000	2.000	90,80	Pal.
[KAPP et al, 2004]	MLPCM	12	1.200	81,75	Pal.
[KOERICH et al, 2002]*	MEM/RN	80.000	4.674	78,05	Cid.
[GÜNTER, 2004b]	MEM	3.997	3.264	80,48	Pal.
	MC	3.997	3.264	83,46	Pal.
	AC	3.997	3.264	84,96	Pal.

2.4 DISCUSSÃO

Este capítulo abordou diferentes tópicos do reconhecimento de manuscritos. Partiu-se da abordagem do reconhecimento de caracteres isolados, já que esta é a base para muitos sistemas de reconhecimento de cadeias de dígitos e palavras. Depois foram estudados alguns métodos de reconhecimento de cadeias numéricas e de palavras manuscritas existentes na literatura.

Tanto o estudo de sistemas de reconhecimento de cadeias numéricas como o estudo de reconhecimento de caracteres isolados podem ser importantes para a aplicações de reconhecimento de palavras. Primeiramente, o estudo de unidades isoladas podem ser um ponto de partida para a proposta de um conjunto de características ou até mesmo para o projeto de um verificador. É também muito importante o estudo de sistemas de reconhecimento de cadeias numéricas já que alguns métodos podem ser aplicados no contexto de

reconhecimento de palavras. Mesmo que o reconhecimento de palavras apresente alguns fatores de complexidade não-existentes no reconhecimento de cadeias numéricas.

Com relação aos sistemas de reconhecimento de palavras manuscritas, observa-se que MEM tem sido uma técnica muito utilizada em sistemas utilizando segmentação implícita. A capacidade de tratar com seqüências temporais de eventos, assim como a possibilidade de concatenar vários MEMs formando um único MEM, são fatores favoráveis à modelagem da escrita e à segmentação implícita. Já RN tem sido mais aplicados em sistemas com algum tipo de segmentação, ou sistemas com abordagem global, já que não possuem a mesma capacidade que MEM para a segmentação.

Diversos fatores podem contribuir para o aumento do desempenho, como foi visto neste capítulo. Sistemas que possuem um mecanismo de verificação mostraram que este esquema pode aumentar a taxa de reconhecimento final. A combinação de classificadores também tem se mostrado como uma técnica eficiente para atingir altas taxas de reconhecimento, já que combina informação de diferentes fontes.

Um ponto importante a ser salientado é que, além de um método ser muito dependente da escolha do conjunto de características e do método de classificação, ainda outros pontos devem ser considerados. Devem ser considerados e analisados métodos de pré-processamento, que podem diminuir a distância entre os exemplos da mesma classe. Devem ser considerados também a avaliação de todos os parâmetros do classificador. Uma prova disso foi demonstrado em [BUNKE & ZIMMERMANN, 2002a] e em [GÜNTER, 2004a]. Nestes trabalhos os autores demonstram que as otimizações em MEM podem render cerca de 10% a mais na taxa de reconhecimento.

Para concluir, podemos observar que apesar dos métodos mais recentes conseguirem boas taxas de reconhecimento, estes sistemas de reconhecimento de palavras ainda são alvos de pesquisas futuras. Os fatores de complexidade ainda pesam muito neste tipo de sistema, e uma prova disso é a utilização da informação de léxico pela grande maioria deles. Com a incorporação da informação de léxico há uma redução muito grande no

espaço de busca do problema, e melhores resultados são alcançados com a delimitação do léxico relacionando-o com a sua aplicação.

3 FUNDAMENTAÇÃO TEÓRICA

Esta capítulo apresenta toda a fundamentação teórica necessária para a compreensão da implementação do método proposto. Além da implementação, são abordadas teorias fundamentais para o entendimento de alguns experimentos realizados.

A organização do capítulo aborda, na Seção 3.1, os conceitos básicos, a topologia, o treinamento e a avaliação dos Modelos Escondidos de Markov (MEMs). Além disso, na Seção 3.2, é apresentado o método de Wang [WANG, 1994] para a definição do tamanho dos MEMs, considerando-se o tamanho das seqüências de observações da base de treinamento. Fora do escopo dos MEMs, a Seção 3.3 apresenta o conceito de Índices de Validação (IV), assim como alguns métodos propostos para IV.

3.1 MODELOS ESCONDIDOS DE MARKOV

Segundo Rabiner ([RABINER, 1989]), um Modelo de Markov é um processo estocástico onde a saída do modelo é um conjunto de estados em cada instante de tempo, sendo que cada estado corresponde a um evento físico (observável). Modelos Escondidos de Markov são uma extensão de Modelos de Markov, onde a observação é uma função probabilística do estado, ou seja, são processos duplamente estocásticos onde um processo não é observável (escondido), mas podem ser observados somente através de outro conjunto de processos estocásticos que produzem a seqüência de observações. Os estados dos MEMs não estão necessariamente relacionados a eventos físicos.

Dada a definição de MEM, as seções aqui contidas mostram a formalização de um MEM, assim como possíveis classificações. Em seqüência, alternativas para o treinamento e para a avaliação de um MEM também são expostas.

3.1.1 Elementos de um MEM

Um MEM pode ser considerado como possuindo os seguintes elementos:

1. Um conjunto de estados denotado por $S = \{S_1, S_2, \ldots, S_N\}$, onde N é o número de estados no modelo, e o estado no tempo t é denotado por q_t

2. Um conjunto de símbolos observáveis denotado por $V = \{v_1, v_2, \ldots, v_M\}$, onde M é o número de símbolos observáveis distintos por estado.

3. A distribuição de probabilidade de transição de estado $A = \{a_{ij}\}$, onde

$$a_{ij} = P[q_{t+1} = S_j | q_t = S_i], \qquad 1 \leq i, j \leq N \tag{3.1}$$

4. A distribuição de probabilidade de observação de símbolo em cada estado j, B $= \{b_j(k)\}$, onde

$$b_j(k) = P[v_k \text{ em } t | q_t = S_j], \qquad 1 \leq j \leq N, \quad 1 \leq k \leq M \tag{3.2}$$

5. A distribuição inicial de estado $\pi = \{\pi_i\}$,

$$\pi_i = P[q_1 = S_i], \qquad 1 \leq i \leq N \tag{3.3}$$

Por conveniência, pode-se usar a notação compacta para denotar um MEM. Esta notação pode ser vista na Equação 3.4. Um MEM pode ser denotado por λ. Dada uma seqüência de observações $O = O_1\ O_2\ \ldots\ O_T$, onde O_t é um dos símbolos de V, e T é o tamanho da seqüência de observações, λ é utilizado para processar uma seqüência de observações O.

$$\lambda = (A, B, \pi) \tag{3.4}$$

3.1.2 Os Três Problemas Básicos de MEMs

Dada a definição de MEM, para que este seja útil em aplicações reais, três problemas básicos devem ser resolvidos. Estes problemas são os seguintes:

Problema 1 : Dada a seqüência de observações $O = O_1\ O_2\ \ldots\ O_T$, e um modelo $\lambda = (A, B, \pi)$, este problema diz respeito sobre como computar eficientemente $P(O|\lambda)$, ou seja, a probabilidade da seqüência de observações, dado o modelo.

Problema 2 : Este problema, dada a seqüência de observações $O = O_1\ O_2\ \ldots\ O_T$, e um modelo $\lambda = (A, B, \pi)$, trata a respeito da escolha da seqüência de estados $Q = q_1\ q_2\ \ldots\ q_T$, a qual é ótima em algum sentido (ou seja, a qual faz mais significado para as observações).

Problema 3 : Este é o problema do treinamento do modelo. Isto é, o ajuste dos parâmetros do modelo $\lambda = (A, B, \pi)$ para maximizar $P(O|\lambda)$.

3.1.3 Tipos de MEMs

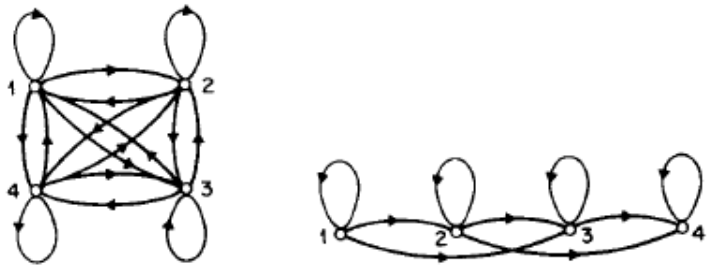
Segundo a descrição em [RABINER, 1989], dois tipos de MEMs são destacados: ergótico e *left-right*. O modelo ergótico é o caso em que todos os estados podem estar conectados, e partindo de um estado, pode haver transição para qualquer outro estado.

Já no MEM *left-right*, os estados estão conectados de maneira que o modelo siga sempre uma ordem da esquerda para o direita, o que é muito importante na área de reconhecimento de fala e de manuscritos, já que modela sinais de maneira temporal. As transições ocorrem apenas para estados em que o índice é maior ou igual ao estado corrente. Isto pode ser melhor expressado na Equação 3.5.

$$a_{ij} = 0, \qquad j < i \tag{3.5}$$

Como a seqüência de estados deve começar no estado 1 e deve terminar no estado N, a Equação 3.6 mostra as propriedades da variável π_i no MEM *left-right*.

$$\pi_i = \left\{ \begin{array}{lll} 1 & \text{se} & i = 1 \\ 0 & \text{se} & i \neq 1 \end{array} \right\} \tag{3.6}$$



(a) (b)

Figura 3.1: Um MEM ergótico na Figura (a) e um MEM *left-right* na Figura (b).

3.1.4 Densidades de Observações Discretas, Contínuas e Semi-contínuas

Além da topologia do modelo, um MEM pode ser diferenciado pela natureza em que os símbolos são observados. A observação dos símbolos pode ser feita de três maneiras:

- **Discreta**: a distribuição é não-paramétrica e quantificada, onde cada observação possui um valor discreto dentro de um conjunto finito. Como geralmente os dados reais são contínuos, cria-se um *codebook* para mapear os sinais contínuos em sinais discretos. Como vantagem, este tipo não necessita de um conhecimento prévio da distribuição dos dados; e como desvantagem, a quantização dos dados no *codebook* cria uma certa distorção, e para adicionar novas classes ao sistema, é preciso reconstruir o *codebook*;

- **Contínua**: utiliza uma função de mistura de densidade, balanceada com a soma de um número de distribuições paramétricas. Evita as distorções e as re-criações do *codebook* existentes na maneira discreta, mas a função de densidade probabilística deve ser definida previamente, e mais dados são necessários para um eficiente treinamento do MEM;

- **Semi-contínua**: utilizada para reduzir as distorções na criação do *codebook* existentes nos MEMs discretos, assim como reduzir a quantidade de dados necessários para treinar os MEMs contínuos. Utiliza um agrupamento de dados não-supervisionados, já que parte da hipótese que diferentes fontes podem possuir similaridades. O seu *codebook* é criado para modelar sinais contínuos em MEMs discretos, possui um conjunto de funções de densidades probabilísticas, onde as distribuições estão em intervalos. O problema deste tipo de MEM é que procura agrupar observações semelhantes, e não observações que possuem informações discriminantes.

3.1.5 Treinamento do MEM

Treinamento é a fase onde os parâmetros do modelo são ajustados para para maximizar $P(O|\lambda)$. Para isso, várias seqüências de observações de uma mesma classe, as quais geralmente possuem semelhanças, são utilizadas pelo algoritmo de *Baum-Welch* para ajustar os valores das variáveis A, B e π.

O algoritmo é uma maneira analítica de resolver este problema, uma vez que não existe uma maneira ótima de resolvê-lo. Antes de entrar em maiores detalhes sobre este algoritmo, as variáveis α e β precisam ser definidas, assim como estimar a probabilidade do modelo através destas.

3.1.5.1 Cálculo da Probabilidade do Modelo Aqui é mostrado como calcula-se a probabilidade de uma seqüência de observações $O = O_1\ O_2\ \ldots\ O_T$, dado o modelo λ

($P[O|\lambda]$). Um procedimento para este cálculo é o *Forward-Backward*.

Este possui duas variáveis, *forward* ($\alpha_t(i)$) e *backward* ($\beta_t(i)$), definidas respectivamente na Equação 3.7 e na Equação 3.8. $\alpha_t(i)$ é a probabilidade da seqüência de observações parcial $O_1\ O_2\ \ldots\ O_t$, (até tempo t) dado o estado S_i no tempo t, dado o modelo λ. Já $\beta_t(i)$ é a probabilidade da seqüência de observações parcial de $t+1$ até o fim, dado o estado S_i no tempo t e o modelo λ.

$$\alpha_t(i) = P(O_1\ O_2\ \ldots\ O_t, q_t = S_i|\lambda) \tag{3.7}$$

$$\beta_t(i) = P(O_{t+1}\ O_{t+2}\ \ldots\ O_T, q_t = S_i|\lambda) \tag{3.8}$$

$\alpha_t(i)$ pode ser calculada intuitivamente da seguinte maneira:

1. Inicialização:

$$\alpha_1(i) = \pi_i b_i(O_1), \qquad 1 \leq i \leq N \tag{3.9}$$

2. Indução:

$$\alpha_{t+1}(j) = \left[\sum_{j=1}^{N} \alpha_t(i)a_{ij}\right] b_j(O_{t+1}), \qquad t = 1,\ \ldots,\ T-1, \quad 1 \leq j \leq N \tag{3.10}$$

3. Terminação

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{3.11}$$

De maneira similar, intuitivamente $\beta_t(i)$ pode ser calculada da seguinte maneira:

1. Inicialização:

$$\beta_T(i) = 1, \qquad 1 \leq i \leq N \tag{3.12}$$

2. Indução:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j), \qquad t = T-1,\ T-2\ ,\ \ldots,\ 1, \qquad 1 \leq i \leq N \tag{3.13}$$

3. Terminação:

$$P(O|\lambda) = \sum_{i=1}^{N} \beta_1(i) \pi_i b_i(O_1), \tag{3.14}$$

Apesar das variáveis traçarem caminhos inversos para o cálculo da probabilidade, ambos os procedimentos produzem o mesmo resultado final. Ou seja, $P(O|\lambda)_{forward} = P(O|\lambda)_{backward}$.

3.1.5.2 Algoritmo de Baum-Welch Um primeiro requisito neste algoritmo é a variável $\xi_t(i,j)$, definida como a probabilidade de estar no estado S_i no tempo t, e no estado S_j no tempo $t+1$, dado o modelo e a seqüência de observações. Uma melhor representação é mostrada na Equação 3.15.

$$\xi_t(i,j) = P(q_t = S_i, q_{t+1} = S_j | O, \lambda) \tag{3.15}$$

Com o conhecimento das variáveis de *forward* e *backward* (definidas na seção 3.1.5.1), $\xi_t(i,j)$ pode ser definida e calculada como mostrado na Equação 3.16, onde o numerador é $P(q_t = S_i, q_{t+1} = S_j, O|\lambda)$, e sua divisão por $P(O|\lambda)$ dá a medida de probabilidade desejada.

$$\begin{aligned} \xi_t(i,j) &= \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{P(O|\lambda)} \\ &= \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)} \end{aligned} \tag{3.16}$$

A variável $\gamma_t(i)$ é definida como a probabilidade de estar no estado S_i no tempo t, dada a seqüência de observações e o modelo. $\gamma_t(i)$ pode ser definida de duas maneiras: uma co-relacionada às variáveis *forward-backward*, como na Equação $3.17a$; ou relacionada com $\xi_t(i,j)$, como pode ser visto na Equação $3.17b$.

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(O|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N}\alpha_t(i)\beta_t(i)} \tag{3.17a}$$

$$\gamma_t(i) = \sum_{j=1}^{N}\xi_t(i,j) \tag{3.17b}$$

Dadas estas definições, o número esperado de transições de S_i é denotado por

$$\sum_{t=1}^{T-1}\gamma_t(i), \tag{3.18}$$

e o número esperado de transições de S_i para S_j é denotado por

$$\sum_{t=1}^{T-1}\xi_t(i,j). \tag{3.19}$$

A partir das fórmulas definidas acima, é possível ter um método para re-estimar os parâmetros π, A e B do MEM. Estas fórmulas são as seguintes:

1. Freqüência esperada no estado S_i no tempo $t = 1$

$$\bar{\pi}_i = \gamma_1(i) \tag{3.20}$$

2. Número esperado de transições do estado S_i para o estado S_j, dividido pelo número

esperado de transições do estado S_i (coeficiente de transição).

$$\bar{a_{ij}} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{3.21}$$

3. Número esperado de vezes no estado j, enquanto observa-se o símbolo v_k, dividido pelo número esperado de vezes no estado j (probabilidade de observação do símbolo).

$$\bar{b_j}(k) = \frac{\sum_{t=1 \; s.t. O_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \tag{3.22}$$

3.1.6 Avaliação do MEM

A avaliação de um MEM consiste em retornar a probabilidade de uma seqüência de observações computada neste. Um procedimento para resolver este problema é o algoritmo de *Viterbi* [RABINER, 1989], que além de buscar a melhor seqüência de estados $Q = \{q_1\, q_2\, \ldots\, q_T\}$, para uma dada seqüência de observações $O = \{O_1\, O_2\, \ldots\, O_T\}$, faz o cálculo da probabilidade do modelo.

Para encontrar a melhor seqüência de estados para uma seqüência de observações O, ou seja, a seqüência "ótima" de estados, o algoritmo utiliza a maior probabilidade através de um caminho, no tempo t, levando em conta as t primeiras observações, e termina no estado S_j.

O algoritmo de *Viterbi* pode ser computado da seguinte maneira:

1. Inicialização:

$$\delta_1(i) \;=\; \pi_i b_i(O_1), \qquad 1 \leq i \leq N \tag{3.23a}$$

$$\psi_1(i) \;=\; 0 \tag{3.23b}$$

2. Recursão:

$$\delta_t(j) \;=\; \max_{1\le i\le N}[\delta_{t-1}(i)a_{ij}]b_j(O_t), \qquad 2\le t\le T, \quad 1\le j\le N \tag{3.24a}$$

$$\psi_t(j) \;=\; \arg\max_{1\le i\le N}[\delta_{t-1}(i)a_{ij}], \qquad 2\le t\le T, \quad 1\le j\le N \tag{3.24b}$$

3. Terminação:

$$P^* \;=\; \max_{1\le i\le N}[\delta_T(i)] \tag{3.25a}$$

$$q_T^* \;=\; \arg\max_{1\le i\le N}[\delta_T(i)] \tag{3.25b}$$

4. Retrocesso do caminho (seqüência de estados):

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \qquad t = T-1,\ T-2,\ \ldots\ ,\ 1 \tag{3.26}$$

As variáveis P^* e q_t^*, ao final do procedimento, possuem respectivamente o valor da probabilidade e a melhor seqüência de estados do MEMs processado pelo algoritmo.

3.2 MÉTODO DE WANG PARA DEFINIÇÃO DO TAMANHO DO MEM

Conforme descrito em [WANG, 1994] é possível determinar o número de estados do MEM a partir das seqüências de observações do conjunto de treinamento. Após calcular o tamanho médio μ e a variância σ^2 das seqüências de observações, N pode ser

calculado conforme a Equação 3.27.

$$\frac{\mu(\mu-1)+\sigma^2}{\mu-1+\sigma^2} < N < \mu+1-\sqrt{2\sigma^2+1} \tag{3.27}$$

Sumariamente, o método, de maneira estatística, estima um intervalo de valores para o número de estados do MEM. Como o método estima um intervalo, então os valores contidos neste intervalo devem ser avaliados.

3.3 ÍNDICES DE VALIDAÇÃO

Índices de Validação [HALKIDI et al, 2001] são métodos utilizados para avaliar quantitativamente os resultados produzidos por algum algorítmo de agrupamento de dados. Existem três categorias diferentes de IV:

- Externos: testam se os pontos do conjunto de dados estão aleatoriamente estruturados ou não.
- Internos: avaliam o resultado de um agrupamento com uma partição conhecida.
- Relativos: avalia o melhor entre uma série de resultados de agrupamentos.

O objetivo da utilização de IV neste trabalho é a avaliação não-supervisionada de diferentes tamanhos para o *codebook*. Com isso, Índices de Validação Relativos (IVR) mostram-se como uma alternativa para a avaliação dos agrupamentos produzidos com o algoritmo K-Médias, utilizando diferentes tamanhos para o número de centros. O número de centros, no caso deste trabalho, é igual ao tamanho do codebook. Nesse contexto, podemos selecionar dois IVR para a avaliação dos resultados produzidos pelo K-Médias: o índice Daives-Bouldin e o índice Xie-Beni.

3.3.1 Índice Davies-Bouldin (DB)

O índice DB é computado através de uma função entre a compactação interna dos agrupamentos e a separação entre eles. A compactação interna do i_{esimo} agrupamento C_i é dada por $S_i = \frac{1}{|C_i|} \sum_{x \in C_i} \{\|x - z_i\|\}$ e a distância entre os agrupamentos C_i e C_j, denotada por d_{ij}, é definido como $d_{ij} = \|z_i - z_j\|$, sendo que z_i é o centro do i_{esimo} agrupamento.

Dadas as definições acima, considerando np como sendo o número total de pontos no espaço, o índice DB é definido por:

$$DB = \frac{1}{np} \sum_{i=1}^{np} R_i \tag{3.28}$$

onde $R_i = \max_{j,j \neq i} \left\{ \frac{S_i + S_j}{d_{ij}} \right\}$.

O objetivo é minimizar o índice DB, ou seja, o resultado de um agrupamento com o menor valor é considerado o melhor resultado.

3.3.2 Índice Xie-Beni (XB)

O índice XB foi originalmente criado para a avaliação de agrupamentos difusos. Porém, pode ser aplicado também para agrupamentos exatos, uma vez que podem ser atribuídos valores binários à matriz de ocorrência. Considerando os pontos de dados $X = \{x_i,\ 1 \leq i \leq np\}$, onde np é o número total de pontos, e os centróides v_j dos grupos $c_j, 1 \leq j \leq nc$, onde nc é o número total de grupos (ou classes), então a matriz de ocorrências $U =\mid u_{ij} \mid$ em agrupamentos binários é dada por:

$$u_{ij} = \left\{ \begin{array}{ll} 0 & \text{se} \quad x_i \notin c_j \\ 1 & \text{se} \quad x_i \in c_j \end{array} \right\} \tag{3.29}$$

Dada a definição acima, então o índice XB pode ser definido matematicamente

como:

$$XB = \frac{\sum_{j=1}^{nc} \sum_{i=1}^{np} u_{ij}^2 \|x_i - v_j\|^2}{np \min_{i,j} \|v_i - v_j\|^2} \tag{3.30}$$

Assim como DB, o objetivo é minimizar o índice produzido pelo XB.

3.4 COMBINAÇÃO DE CLASSIFICADORES

Um sistema com múltiplos classificadores é uma solução poderosa para problemas de reconhecimento de padrões envolvendo um grande número de classes e padrões ruidosos. O seu desempenho tende a ser melhor que um sistema com um único classificador porque permite o uso simultâneo de descritores de características e técnicas de classificação de naturezas distintas.

As soluções para o problema de combinação de classificadores ficam dependentes do tipo da saída fornecida pelos classificadores. Estas podem ser divididas em três categorias, de acordo com o nível de informação provido pelo classificador [XU et al, 1992]. Na primeira, o classificador retorna apenas a informação da classe mais provável, ou com menor distância do padrão classificado. Na segunda categoria, o classificador retorna uma lista ascendente ou descendente das classes escolhidas como correspondente àquele padrão. Na última categoria, além da informação da lista, o classificador retorna um valor de distância ou probabilidade relacionado a cada uma das classes. Da primeira para a terceira categoria há um nível crescente de informação para ser extraída da classificação. As nomenclaturas para estes tipos de classificadores são, respectivamente, nível abstrato, nível de *ranking*, e nível de medida.

Quando a combinação de classificadores contém diferentes tipo de saídas o problema da combinação fica restrito ao classificador com menos informação. Por exemplo um classificador com saída em nível de *ranking* e um classificador com saída em nível abstrato, o problema fica restrito à combinação de informações de nível abstrato.

Seguindo o tipo de informação provida pelos classificadores, existem três tipos

de combinações de classificadores [XU et al, 1992]. O tipo 1 é baseado na combinação de informação em nível abstrato. Este tipo de combinação pode combinar diferentes naturezas de classificadores, portanto, é o método mais abrangente de combinação. O método de votação é um modo de combinação neste nível, onde a classe com maior número de votos entre os classificadores é escolhida. Para solução de empates, a atribuição de pesos aos classificadores é uma alternativa propostas.

O tipo 2 é baseado na combinação de informações em nível de *ranking*. Este nível possui mais informações que o nível abstrato, e um método para a solução deste problema é chamado *Borda Count* [HO et al, 1994]. Este método é uma variante do método de votação, onde são atribuídos votos de acordo com a posição da classe no *ranking*. Assim como o método de voto, a atribuição de pesos aos classificadores também é uma solução proposta para a decisão no caso de empate.

Por último, o tipo 3 é baseado na combinação em nível de medida. Este tipo é uma variante do nível de *ranking*, porém com a informação da distância ou da probabilidade de cada classe. Com isso, esta informação pode ser utilizada tanto como desempate nos métodos de combinação citados anteriormente, como para propor novos métodos de combinação. Como os classificadores geralmente são diferentes, é necessário que as informações sejam normalizadas antes da combinação.

3.5 RESUMO

Este capítulo apresentou as principais teorias relacionadas à implementação do método proposto. Algumas definições e classificações de MEMs foram apresentadas, assim como os principais algorítmos para resolver problemas como o treinamento, a avaliação dos modelos, e a seqüência de estados absorvidos pelo modelo.

Foram apresentados também dois métodos relacionados à otimização dos MEM. Primeiro foi descrito um método para calcular o número de estados, de maneira

que estes estão relacionados aos tamanhos das seqüências de observações na base de treinamento. Em seguida, dois IV foram descritos. Estes são utilizados neste trabalho para estimar o tamanho do *codebook* de maneira não-supervisionada. O desempenho nesta tarefa é avaliado experimentalmente.

Para finalizar este capítulo, foi dada uma introdução sobre combinação de classificadores. Nesta foram descritos os três tipos de problemas relacionados, assim como foram citadas algumas soluções para estes.

4 MÉTODO PROPOSTO

Neste capítulo é apresentado o método proposto para o reconhecimento de palavras manuscritas. Este método tem como objetivo o reconhecimento de palavras manuscritas sem restrições quanto à grafia, utilizando um léxico para validar a palavra a ser reconhecida.

A Seção 4.1 traz uma explicação sobre o método como um todo, dando uma breve introdução e esclarecimentos sobre a função de cada módulo. Na Seção 4.2 e na Seção 4.3 são apresentados detalhes sobre cada um dos estágios do sistema.

4.1 ARQUITETURA

O sistema consiste de dois estágios, conforme é possível observar na Figura 4.1. No primeiro estágio são feitos os pré-processamentos na imagem para que possam ser extraídas as características de uma imagem binária. Após esta primeira extração de características, a imagem é classificada através da concatenação dos MEMs de cada caracter, conforme o léxico do sistema, utilizando o algoritmo de *Viterbi* (descrito na Seção 3.1.6). Os N melhores resultados ou palavras candidatas serão selecionadas para compor o léxico do segundo estágio.

O segundo estágio extrai características dos segmentos da imagem utilizada no estágio anterior. Desta vez é possível extrair novas informações, uma vez que agora dispomos dos limites de cada caracter encontrados no primeiro estágio. Depois da extração destas características é feita a reclassificação, utilizando modelos diferentes dos utilizados no primeiro estágio, os quais consideram informação extraída não só de colunas, mas também das linhas dos caracteres segmentados. Desta forma, a lista contendo as N palavras candidatas fornecidas pelo primeiro estágio será reclassificada segundo as novas probabilidades obtidas no segundo estágio.

Resumindo, o primeiro estágio tem como objetivo a segmentação implícita

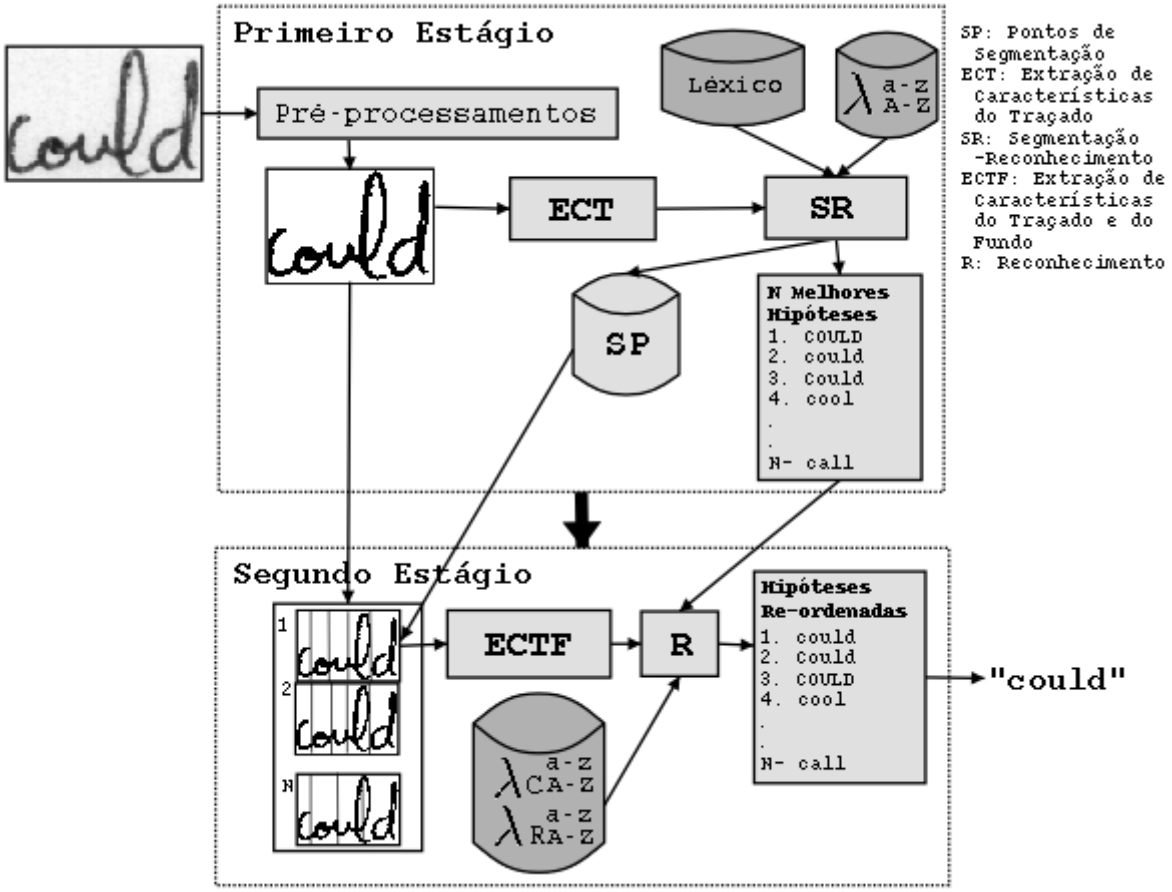


Figura 4.1: Visão geral do método proposto

da palavra e a geração das hipóteses de segmentação/reconhecimento da palavra. Já o segundo estágio é utilizado como forma de compensar as possíveis perdas de desempenho do primeiro estágio, uma vez que este tenta contemplar segmentação e reconhecimento utilizando apenas informações do traço da palavra extraídas das colunas da mesma.

4.2 PRIMEIRO ESTÁGIO

Esta seção traz explicações sobre o funcionamento de cada componente do primeiro estágio do sistema. Os componentes deste estágio são: Pré-processamento,

Extração de Características do Traçado (ECT) e Classificação-Segmentação (CS). O objetivo principal desta fase é escolher as N melhores hipóteses com base na segmentação implícita.

4.2.1 Pré-Processamento

Nesta fase são feitas adaptações e correções na imagem para que esta possa ter suas características extraídas. Esta fase procura reduzir ruídos ou deformações nas palavras para que não sejam produzidas grandes divergências entre as amostras.

A primeira operação feita na imagem é sua binarização (limiarização). A necessidade ou não de pré-processamento se dá pela natureza do método de extração de características utilizado e a base de dados em que está sendo aplicado. Se o conjunto de características utilizado for baseado em características binárias, e a base de dados se encontrar em níveis de cinza, então este pré-processamento torna-se necessário. Para esta tarefa é utilizado o método de Otsu ([OTSU, 1979]), sendo que este seleciona automaticamente o melhor limiar para que uma imagem em nível de cinza seja convertida para uma imagem em preto e branco, sem a necessidade da avaliação manual de diferentes limiares para esta tarefa. Um exemplo de limiarização pode ser visto na Figura 4.2.

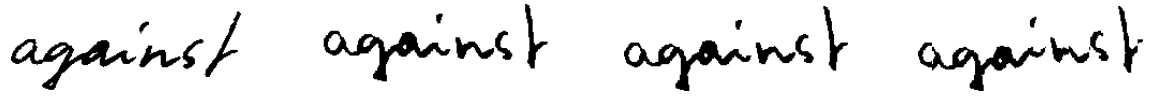
(a) (b)

Figura 4.2: Exemplo da aplicação do algoritmo de Otsu para limiarização de imagens em nível de cinza, sendo que a Figura 4.2(b) é o resultado da aplicação deste método na Figura 4.2(a).

Os outros pré-processamentos incorporados ao método tem a finalidade de corrigir imperfeições que podem causar confusões entre as classes. Entre estas imperfeições, podemos citar a inclinação vertical dos caracteres, a inclinação na linha de base da palavra,

e as diferenças entre os tamanhos dos caracteres. Todas estas imperfeições podem causar problemas para o extrator de características. Primeiro, se o método de extração de primitivas extrai características das colunas, a inclinação vertical dos caracteres pode fazer com que dados de caracteres distintos sejam processados no mesmo conjunto de características. Já a inclinação da linha de base e as diferenças entre os tamanho do caracteres podem trazer confusões, por exemplo, se é considerada a distância entre o topo da *bounding box*. Métodos para normalizarem estas imperfeições podem fazer com que o ruído causado por estes problemas seja amenizado.

Para a correção de inclinação vertical, o método escolhido foi proposto em [BRITTO, 2001]. A escolha deste deu-se pelo fato que o autor mostrou que o método contribuiu para o aumento na taxa de reconhecimento do sistema. Já os métodos para correção da inclinação na linha de base e para a normalização do tamanho dos caracteres foi proposto em [EL-YACOUBI, 1996].



(a) (b) (c) (d)

Figura 4.3: Exemplo dos pré-processamentos aplicados no métodos. Primeiro é corrigida a inclinação vertical da imagem original (a) em (b). Em seguida da imagem (b) é corrigida a inclinação da linha de base, resultando na imagem (c), a qual é normalizada com relação ao corpo da palavra, resultando em (d).

A importância de cada um destes fatores será avaliado individualmente, através de experimentos, e apenas os pré-processamentos significativos ao método serão mantidos. Ou seja, o método proposto apenas irá incorporar os pré-processamentos que se adequarem melhor ao conjunto de características e à base de dados, e experimentos com a presença e a ausência dos métodos de pré-processamento propostos dão a possibilidade para esta avaliação.

4.2.2 Extração de Características do Traçado (ECT)

Esta seção destina-se a explicar o módulo de ECT contido no primeiro estágio do sistema. Este estágio funciona como o *Foreground Feature Extraction*, do *String Contextual-Based Stage*, do sistema proposto em [BRITTO, 2001]. Primeiramente é extraída a imagem $BB_{L\times C}$, que é a *bounding box* da imagem. BB deve existir já que apenas esta parte da imagem conterá informações relevantes para o sistema. C corresponde ao número de colunas da imagem, e L corresponde ao número de linhas.

O processo de extração de características é feito através de uma *slide-window* $SL_{L\times 1}$, onde L é igual ao número de linhas em BB. Esta *slide-window* percorre BB, da esquerda para a direita, extraindo o vetor das características descritas na Seção 4.2.2.1. Estes vetores são mapeados para um *Codebook* CB com CN símbolos. Inicialmente CB terá $CN = 256$ símbolos, já que com este número o trabalho apresentado em [BRITTO, 2001] alcançou os melhores resultados. Experimentalmente são testados outros valores para CN, analisando os melhores resultados obtidos, a fim de ajustar este parâmetro com o melhor valor para este sistema.

Após o mapeamento de todos os vetores extraídos, estes vetores mapeados são salvos como a seqüência de observações S_{SN} da imagem. SN corresponde ao tamanho da seqüência de observações, e deve ser igual ao valor de C em BB.

A extração de características neste módulo considera dois conjuntos de características distintos. O primeiro, é apresentado na Seção 4.2.2.1, possuindo 34 características. Já o segundo possui apenas nove primitivas, e é apresentado na Seção 4.2.2.2. Estes conjuntos são analisados individualmente, permitindo a comparação dos resultados de ambos tanto para a segmentação quanto para o reconhecimento. A combinação destes conjuntos também é avaliada em diferentes alternativas.

4.2.2.1 Características do Traçado (CT) As características aqui descritas são extraídas do traçado da palavra contida na imagem, ou seja, dos pontos pretos de uma imagem binária. Estas características estão descritas com maiores detalhes em [BRITTO, 2001].

Este conjunto de características compõe um total de 34 características, sendo que 32 são características locais e duas são características globais. Todas estão normalizadas entre zero e um. As características locais são baseadas em transições do fundo para o traçado, assim como transições do traçado para o fundo. Para cada transição são obtidas a direção média e a variância correspondente através de médias de estimadores estatísticos. A Figura 4.4 mostra um exemplo das transições e de suas observações direcionais. Além das direções médias e das variâncias, são calculadas as posições de cada transição em relação ao topo da *bounding box*; e se as transições pertencem ao contorno interno ou externo, o que indica a presença de *loops* na imagem. Como explicado em [BRITTO, 2001], a análise de diversas imagens de dígitos mostrou um máximo de 8 transições por coluna. Como cada transição possui 4 características relacionadas, o vetor de características locais possui 32 características.

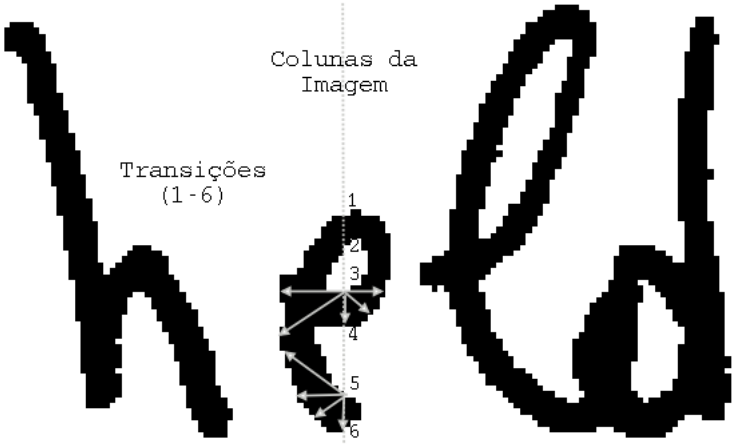


Figura 4.4: Transições na coluna de uma imagem da palavra "held" e as observações direcionais das transições 3 e 5. Figura adaptada de [BRITTO, 2001].

Em adição às características locais comentadas no parágrafo anterior, duas características globais são acrescentadas no vetor de características, totalizando 34 características. As características globais são baseadas na projeção vertical de pontos pretos para cada coluna, e a derivação de projeções verticais entre colunas adjacentes. Estas 34 características constituem as informações extraídas de cada coluna da imagem.

4.2.2.2 Características Geométricas (CG) Neste conjunto de características, proposto em [MARTI & BUNKE, 2001], nove características geométricas são computadas. Destas, três trazem informações sobre a quantidade de pontos pretos e sua posição na janela $SL_{L\times 1}$, enquanto que as outras seis guardam informações complementares sobre a escrita.

As três primeiras primitivas, caracterizando a janela de um ponto de vista global, são o peso da janela (número de pontos pretos), seu centro de gravidade e o momento de segunda ordem. As outras características, que trazem informações mais específicas sobre a escrita, armazenam as seguintes informações:

1. Posição do contorno superior da janela;
2. Posição do contorno inferior da janela;
3. Orientação do contorno superior da janela através do gradiente do contorno na posição da janela;
4. Orientação do contorno inferior da janela através do gradiente do contorno na posição da janela;
5. O número de transições branco-preto (e *vice-versa*) entre o contorno superior e o inferior;
6. Número de pontos pretos entre o contorno superior e inferior.

Todas as nove características aqui descritas, assim como no outro conjunto de características, são normalizadas entre zero e um. Sobre a orientação do contorno superior e inferior, é necessário considerar duas janelas sucessivas para calcular o gradiente.

4.2.3 Classificação-Segmentação (CS)

Este módulo faz uma primeira classificação da palavra escolhendo as N melhores hipóteses, ao mesmo tempo que encontra os pontos de segmentação no traçado desta. A probabilidade de cada palavra é calculada através de uma busca no léxico, utilizando o algoritmo de *Viterbi* para avaliar cada uma das hipóteses.

Como os MEMs ($\lambda_F^a, \ldots, \lambda_F^z, \lambda_F^A, \ldots, \lambda_F^Z$, sendo F o número de características) do sistema modelam cada caracter, os modelos das palavras são formados a partir da concatenação destes conforme a informação do léxico. A Figura 4.5 mostra a ilustração desta concatenação.

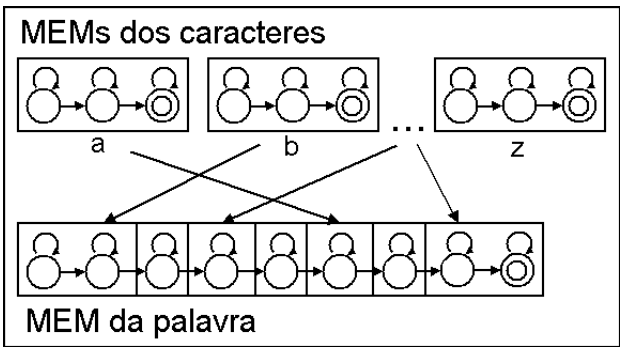


Figura 4.5: Modelos das palavras formados da concatenação dos MEMs dos caracteres.

Para otimizar o tempo de processamento utilizado na computação das hipóteses, um léxico em árvore é utilizado (Figura 4.6). Esta estrutura armazena os dados computados, e permite seu compartilhamento com palavras possuindo os mesmo prefixos. Isto faz com que sejam evitados vários processamentos que são redundantes num léxico linear.

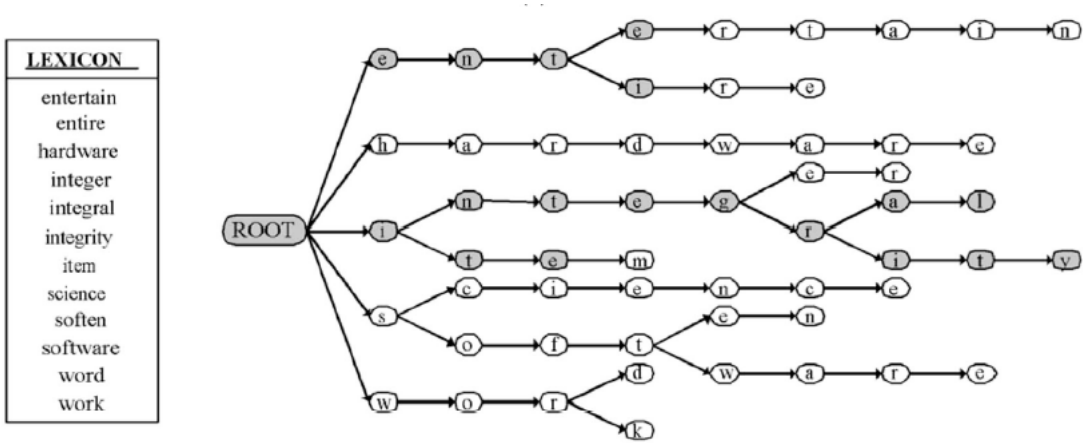


Figura 4.6: Modelo de léxico utilizando uma árvore trie. Figura adaptada de [ARICA & YARMAN-VURAL, 2002].

Os MEMs ($\lambda_F^a, \ldots, \lambda_F^z, \lambda_F^A, \ldots, \lambda_F^Z$) são treinados com uma versão modificada do algoritmo de *Baum-Welch* (Seção 3.1.5). Como os MEMs para cada palavra são formados pela concatenação de sub-unidades (caracteres), foi necessária a adaptação deste algoritmo para re-estimar os parâmetros de cada modelo de caracter utilizando imagens contendo palavras inteiras.

Esta fase gera um *ranking* com as N melhores possibilidades entre as palavras contidas no léxico. Estas N hipóteses, além de conter os pontos de segmentação, são o léxico utilizado no segundo estágio.

4.2.3.1 Modificação no Algoritmo de Baum-Welch A modificação feita no algoritmo de *Baum-Welch* visa o treinamento de concatenações de MEMs, de maneira que a re-estimação dos parâmetros destes realize-se de maneira individual. Esta modificação permite que os MEMs recebam informações isoladas para cada caracter, ao mesmo tempo em que são incorporadas as informações de contexto.

Considerando um sistema para reconhecimento de palavras manuscritas, onde os modelos das palavras são formados pela concatenação dos modelos dos caracteres, o

algoritmo, incluindo as modificações acrescentadas, pode ser descrito nos seguintes passos:

1. **Concatenação dos modelos**: dada uma palavra PL contendo L caracteres, e sabe-se o valor correspondente da palavra, o modelo λ utilizado pelo algoritmo é o seguinte:

$$\lambda = \lambda_1 \oplus \lambda_2 \oplus \ldots \oplus \lambda_L \tag{4.1}$$

   onde $\lambda_1, \lambda_2, \ldots, \lambda_L$ corresponde a cada caracter da palavra.

2. **Cálculo de ξ e γ**: o cálculo destas variáveis é feita da mesma maneira como foi apresentado na Seção 3.1.5.

3. **Re-estimação dos parâmetros**: a re-estimação dos parâmetros dá-se de forma individual para cada modelo, levando-se em consideração um deslocamento conforme o número de estados de cada modelo, armazenado na variável $offset(i)$. Sendo n_j, onde $0 \leq j \leq L$, o número de estados de cada sub-modelo da palavra, $offset(i)$ é dado por:

$$offset(i) = \left\{ \begin{array}{ll} 0 & \text{se} \quad i = 1 \\ \sum_{j=1}^{i-1} n_j & \text{se} \quad i > 1 \end{array} \right\} \tag{4.2}$$

   As alteração na re-estimação das variáveis π, A e B, para cada modelo, foram as seguintes:

$$\bar{\pi}_i(l) = \gamma_1(i + offset(l)), \qquad 1 \leq l \leq L \tag{4.3a}$$

$$\bar{a}_{ij}(l) = \frac{\sum_{t=1}^{T-1} \xi_t(i + offset(l), j + offset(l))}{\sum_{t=1}^{T-1} \gamma_t(i + offset(l))}, \qquad 1 \leq l \leq L \tag{4.3b}$$

$$\bar{b}_j(l,k) = \frac{\sum_{t=1}^{T}{}_{s.t.O_t=v_k} \gamma_t(j+offset(l))}{\sum_{t=1}^{T} \gamma_t(j+offset(l))}, \qquad 1 \leq l \leq L \tag{4.3c}$$

Estas alterações no algoritmo garantem que, para cada exemplo da base de treinamento, mesmo havendo a necessidade da concatenação dos modelos para que estes sejam treinados, os parâmetros de cada um destes são individualmente re-estimados.

4.3 SEGUNDO ESTÁGIO

Neste estágio é verificada a classificação feita no estágio anterior, já que novas informações estão disponíveis devido à segmentação implícita do outro estágio. Além disso, aqui o espaço de busca é reduzido, já que o léxico é formado pelo *ranking* construído por CS.

Diferentemente do estágio anterior, com a informação dos pontos de segmentação, é possível extrair características separadamente de cada caracter. Deste modo, a extração de características pode ser feita utilizando um esquema de zoneamento, extraindo informações de linhas e colunas.

As seções a seguir trazem explicações detalhadas sobre cada módulo do segundo estágio, explicando o funcionamento e as informações extraídas pelo sistema.

4.3.1 Extração de Características do Traçado e do Fundo (ECTF)

Neste estágio, em adição ao conjunto de características proposto na Seção 4.2.2.1, são extraídas informações do fundo de cada caracter. Estas informações procuram adicionar informações com maior poder discriminativo. Os pontos de segmentação encontrados no primeiro estágio permitem a delimitação de cada caracter na palavra, e com isso, determinar o fundo correspondente a cada um destes.

Os pontos de segmentação também permitem que sejam extraídas informações

das linhas do caracter, e não apenas das colunas. Isto faz com que haja um zoneamento implícito. Espera-se com esta combinação de informações de linhas e colunas que ocorra um aumento do desempenho do sistema. A ocorrência deste é avaliado experimentalmente no Capítulo 5.

4.3.1.1 Características de Traçado e Fundo (CTF) As características de fundo da imagem são baseadas nas informações de concavidades, que trazem informações sobre a geometria dos caracteres. Para cada coluna (ou linha) é extraído um vetor de concavidades com treze posições, onde em cada posição do vetor está a contagem dos pontos brancos conforme a configuração mostrada na Figura 4.7. Esta configuração é baseada em código de cadeia de quatro direções.

Cada direção é explorada até encontrar um ponto preto ou alcançar o limite da *bounding box* da imagem. As posições dez a treze (A a D) são consideradas para detectar com maior precisão a ocorrência de *loops*. Um exemplo da extração de informação de fundo pode ser visto na Figura 4.8 e sua respectiva Tabela 4.1. Após a contagem dos pontos, cada posição do vetor é normalizada entre zero e um, dividindo cada posição pelo total de pontos computados em cada coluna ou linha.

Este conjunto de característica procura adicionar mais informações ao conjunto proposto na Seção 4.2.2.1, para aprimorar o reconhecimento. Como CT constitui um vetor de 34 primitivas, e agora são extraídas treze características do fundo, CTF forma um vetor com 47 posições.

Tabela 4.1: Tabela de extração de concavidades das colunas da imagem 4.8(b).

coluna	1	2	3	4	5	6	7	8	9	A	B	C	D
1	3	20	0	0	0	0	0	0	0	0	4	0	0
⋮	-	-	-	-	-	-	-	-	-	-	-	-	-
53	0	0	5	19	0	0	0	0	0	0	3	0	0

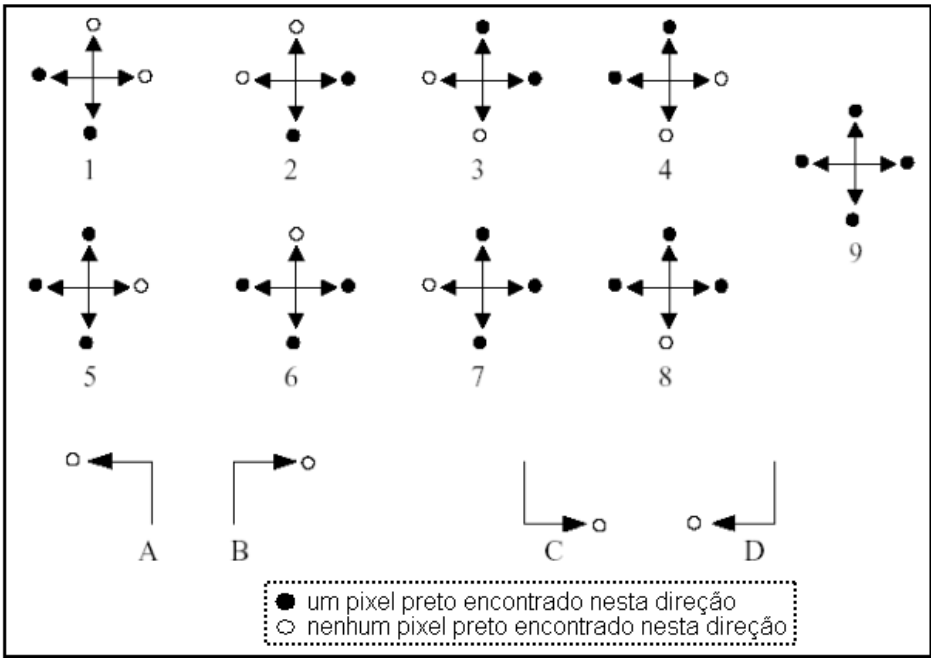


Figura 4.7: Configurações de concavidades utilizadas para relacionar pontos brancos. Figura adaptada de [BRITTO, 2001].

4.3.2 Reconhecimento (R)

Com a extração de características descrita na Seção 4.3.1, este estágio fará a verificação das palavras classificadas em CR. Nesta etapa os caracteres são treinados e classificados de maneira individual, e suas probabilidades são utilizadas para verificar a hipótese.

Este estágio possui 104 novos MEMs. O treinamento dos modelos ($\lambda_c^a, \ldots, \lambda_c^z, \lambda_c^A, \ldots, \lambda_c^Z$ e $\lambda_l^a, \ldots, \lambda_l^z, \lambda_l^A, \ldots, \lambda_l^Z$) é realizado utilizando o algoritmo de *Baum-Welch* (Seção 3.1.5), e cada MEM é treinado individualmente utilizando uma base de dados com caracteres isolados. Cada caractere possui dois MEMs associados, sendo um para observações de coluna, e o outro para observações de linha.

O reconhecimento é feito através da soma das probabilidades de linha e coluna de cada segmento das hipóteses geradas no primeiro estágio. Em maiores detalhes, para

(a) Imagem original

```
22222222222222222BBBB55BBBB333333333333333333333333
22222222222222BBBBBBBBBBBBBB333333333333333333333333
222222222222222BBBBBBBBBBBBBBBB333333333333333333333
22222222222222BBB999999999BBBBBB333333333333333333333
222222222222BBBBB9999999999BBBBBB333333333333333333333
2222222222BBBBB99999999999BBBBBB955555555555555555BBBB
22222222BBBBBB99999999999BBBBBB955555555555555555BBBBB
2222222BBBBBB99999999999BBBBBB995555555555555555BBBBBB
2222222BBBB999999999999BBBBB9995555555555555555BBBBBB4
222222BBBBB999999999999BBBBB99955555555555555BBBBBBBB4
22222BBBBB9999999999999BBBB999955555555555BBBBBBB7BB4
2222BBBBB99999999999999BBBBB9999955555555555BBBBBB7777B4
2222BBBBB999999999999BBBBBB9999955555555BBBBBB4444444
2222BBBB99999999999BBBBBBB999995555555BBBBBBB44444444
222BBBB999999999999BBBBBBB99999955555BBBBBBB444444444
222BBBB99999999999BBBBBBB9999995555BBBBB44444444444
22BBBB9999999999BBBBBBBBB999999555BBBBBBB444444444444
22BBBB99999999999BBBBBBBBB999999BBBBBBBB4444444444444
2BBBBB999999999BBBBBB7BBBBB99999BBBBBBBB44444444444444
2BBBB9999999BBBBBB777BBBBB999BBBBBBBB4444444444444444
BBBBB999999BBBBBBB777BBBBBBBBBBBBBBB444444444444444444
BBBBB9999BBBBBBBB7777BBBBBBBBBBBBBB4444444444444444444
BBBBBBBBBBBBBBB777777BBBBBBBBBBBBB44444444444444444444
BBBBBBBBBBBBB777777777BBBBBBBBBB444444444444444444444
1BBBBBBBBBBB7777777777BBBBBBBBB4444444444444444444444
11BBBBBBBB77777777777777BBBB444444444444444444444444
1111BBBB444444444444444444444444444444444444444444444
```

(b) Concavidades da imagem (a)

Figura 4.8: Exemplo de extração de concavidades

cada imagem a ser reconhecida, esta possui N hipóteses geradas pelo primeiro estágio. Cada hipótese Hyp_i ($0 \leq i \leq N$) possui Ts_i segmentos, e cada segmento é denotado por $seg_{i,j}$, onde $0 \leq j \leq Ts_i$. Sendo $PC(seg_{i,j})$ e $PL(seg_{i,j})$ as probabilidades de coluna e linha, respectivamente, para cada segmento, então a probabilidade da palavra é i é dada por:

$$Pver_i = \sum_{j=1}^{Ts_i} (\log PC(seg_{i,j}) + \log PL(seg_{i,j})) \tag{4.4}$$

Além da probabilidade mostrada na Equação 4.4, a verificação poderá utilizar a probabilidade de cada hipótese no primeiro estágio como uma informação relevante nesta verificação. Considerando $Phyp_i$ a probabilidade da palavra computada pelo primeiro

estágio, então a probabilidade final da palavra i através da combinação é dada por:

$$Ppalavra_i = \log Phyp_i + \log Pver_i \tag{4.5}$$

4.3.2.1 Limiar de Decisão O limiar de decisão é uma alternativa para a combinação das probabilidades do primeiro estágio e da verificação. Neste caso, a aplicação de um limiar pode ser eficiente para limitar a verificação apenas das palavras em que o primeiro estágio não demonstram um certo nível de certeza. Para verificar este nível de certeza, a primeira e a segunda hipótese do primeiro estágio são avaliadas. A verificação só ocorre com as palavras onde a diferença das probabilidades da primeira e da segunda hipótese estão abaixo do limiar estimado.

Matematicamente, seguindo a notação da Equação 4.6, e considerando lm como o valor do limiar de decisão, então a probabilidade $Ppalavra_i$ é dada da seguinte maneira:

$$Ppalavra_i = \left\{ \begin{array}{rl} \log Phyp_i & \text{se} \quad (Phyp_1 - Phyp_2) \geq lm \\ \log Phyp_i + \log Pver_i & \text{se} \quad (Phyp_1 - Phyp_2) < lm \end{array} \right\} \tag{4.6}$$

4.4 COMBINAÇÃO DE CONJUNTOS DE CARACTERÍSTICAS

Na Seção 4.2.2 foram propostos dois conjuntos de características para serem utilizados no primeiro estágio do sistema. Com isso, foi possível definir dois sistemas distintos para a solução do problema proposto. Como ambos conjuntos extraem características distintas, então é possível traçar estratégias para combiná-los, com o objetivo de melhorar o desempenho do sistema.

Uma maneira de combinar estes conjuntos é utilizando métodos clássicos de combinações de classificadores, combinando a saída final do sistema utilizando o conjunto de características CG com o sistema utilizando o conjunto de características CT. Esta

combinação dá-se após a verificação das hipóteses computadas em cada implementação do primeiro estágio. Considerando que a saída do método, após a verificação, é uma lista contendo as N hipóteses do primeiro estágio reordenadas, assim como suas respectivas probabilidades, a Seção 4.4.1 traz os detalhes de como os métodos de combinação de classificadores são aplicados neste trabalho.

Outra maneira é a combinação dos dois conjuntos de primitivas no primeiro estágio, com o objetivo de otimizar a extração dos pontos de segmentação e a computação das hipóteses. Com isso a verificação dá-se num único processo, e duas alternativas são propostas para esta tarefa. A primeira utiliza dois conjuntos de características, e dois MEMs para cada classe. A segunda utiliza dois conjuntos de características mas apenas um MEM para cada classe. Estas abordagens são detalhadas na Seção 4.4.2 e Seção 4.4.3, respectivamente.

4.4.1 Combinação de Sistemas Com Métodos da Literatura

A saída produzida por cada sistema, após a verificação, é uma lista contendo N classes, cada qual com sua probabilidade correspondente. Esta lista está ordenada de maneira decrescente, onde a primeira posição possui a maior probabilidade e a N_{esima} posição possui a menor probabilidade.

Três métodos são avaliados para a combinação das listas produzidas: Método de Votação, Método *Borda Count* (BC) e Combinação de Probabilidades. Originalmente apenas o último utiliza as probabilidades diretamente, porém, neste trabalho estas são utilizadas para a tomada de decisão em caso de empate nos dois primeiros métodos.

Para garantir que as probabilidades dos classificadores estão com valores compatíveis, é necessário normalizá-las previamente à combinação. Supondo que $P(i, j)$ seja a probabilidade da palavra i no sistema j, então $P_{norm}(i, j)$ é a probabilidade normalizada

conforme a seguinte equação:

$$P_{norm}(i,j) = \frac{P(i,j)}{\sum_{k=1}^{N} P(k,j)} \tag{4.7}$$

4.4.1.1 Votação O método de votação considera apenas a melhor hipótese de cada sistema. Dentre estas, é escolhida a hipótese com maior número de votos entre todos os sistemas utilizados. Como neste caso apenas dois sistemas são utilizados, e considerando que $palavra_1$ e $palavra_2$ são as palavras com maior probabilidade nos sistemas um e dois, respectivamente, então a votação segue o seguinte critério:

$$resultado\_final = \left\{ \begin{array}{rcl} palavra_1 & \text{se} & palavra_1 = palavra_2 \\ desempate & \text{se} & palavra_1 \neq palavra_2 \end{array} \right\} \tag{4.8}$$

Dois critérios são avaliados para a solução em caso de empate: a probabilidade máxima entre as hipóteses, e a probabilidade mínima. No caso da utilização da probabilidade máxima, o desempate é computado da seguinte maneira:

$$desempate = \left\{ \begin{array}{rcl} palavra_1 & \text{se} & P_{norm}(palavra_1) > P_{norm}(palavra_2) \\ palavra_2 & \text{se} & P_{norm}(palavra_1) < P_{norm}(palavra_2) \end{array} \right\} \tag{4.9}$$

Para utilizar o desempate considerando a probabilidade mínima, basta fazer a inversão das comparações na Equação 4.9. Com isso, a palavra escolhida é a que possui menor valor de probabilidade entre as duas.

4.4.1.2 Borda Count O método BC é uma variação do método de votação para considerar listas de hipóteses na combinação. Cada hipótese, na sua respectiva lista, recebe um número de votos, sendo que este depende da classificação da palavra na lista. O votos são atribuídos de maneira decrescente, sendo que a melhor hipótese recebe um maior número de votos. Considerando que existam TL listas na combinação, então é criada uma lista

$TL + 1$ contendo a união das hipóteses nas listas e o somatório do número de votos para cada uma destas.

Neste problema, duas listas são combinadas e faz-se necessário a criação de uma terceira lista. Para atribuir votos nas duas primeiras listas, considerando N hipóteses, é seguida a seguinte equação:

$$votos_{ij} = N - 1, \quad onde\ 1 \leq j \leq 2 \tag{4.10}$$

Considerando que $palavras1$ e $palavras2$ são as listas de palavras a serem combinadas, então $palavras3$ é criada segundo a equação:

$$palavras3 = palavras1 \cup palavras2 \tag{4.11}$$

O número de votos na terceira lista, sendo N_{max} o número de palavras nesta, é computado da seguinte maneira:

$$votos_{i3} = votos_{j1} + votos_{k2}, \tag{4.12}$$

$$onde\ j = \arg(palavras1_l = palavras3_i), 1 \leq l \leq N,$$

$$k = \arg(palavras2_m = palavras3_i), 1 \leq m \leq N,$$

$$1 \leq i \leq N_{max} \tag{4.13}$$

Após a computação dos votos, a lista $palavras3$ é ordenada maximizando o número de votos. Em caso de empate, as probabilidades são utilizadas. Para a combinação destas, três alternativas são utilizadas. A primeira e a segunda, assim como no método de votação, são as probabilidades normalizadas máximas e mínimas das palavra, e a terceira é a média das probabilidades normalizadas.

4.4.1.3 Combinação de Probabilidades A combinação de probabilidades funciona de maneira similar ao método BC, porém as palavras são ordenadas considerando apenas as probabilidades. As mesmas combinações de probabilidades utilizadas em BC para o desempate são utilizadas nesta abordagem. Ou seja, as probabilidades normalizadas máximas e mínimas das palavras, e a média das probabilidades normalizadas.

Outras combinações, como o produto e a soma das probabilidades normalizadas, não são aplicáveis já que as listas de saída dos sistemas não contêm necessariamente as mesmas palavras. Isto é, uma palavra incluída na terceira lista pode estar em apenas uma das duas primeiras listas, e o produto faria com que as probabilidades destas palavras ficassem maiores que as probabilidades das palavras que sofreram a multiplicação. Portanto, a média, o máximo e o mínimo, são alternativas adequadas para o caso em que a ocorrência de palavras não é homogênea.

4.4.2 Combinação de Conjuntos de Características com MEMs Distintos

Nesta abordagem, MEMs diferentes são treinados para cada um dos conjuntos de características, possuindo os parâmetros ótimos encontrados nos respectivos experimentos. A combinação dá-se através das saídas dos MEMs de cada conjunto de características, gerando um conjunto de pontos de segmentação para cada conjunto de primitivas, mas um único *ranking*. A Figura 4.9 traz uma visão geral sobre o funcionamento desta combinação.

O *ranking* com as melhores hipóteses é compostos das palavras do léxico com maior probabilidade, segundo a Equação 4.14. Nesta, $P_{CG}(palavra_i)$ é a probabilidade da $palavra_i$ com o conjunto CG, $P_{CT}(palavra_i)$ é a probabilidade da $palavra_i$ com o conjunto CT, e TL é o tamanho do léxico.

$$P(palavra_i) = \log P_{CG}(palavra_i) + \log P_{CT}(palavra_i), 0 \leq i \leq TL \tag{4.14}$$

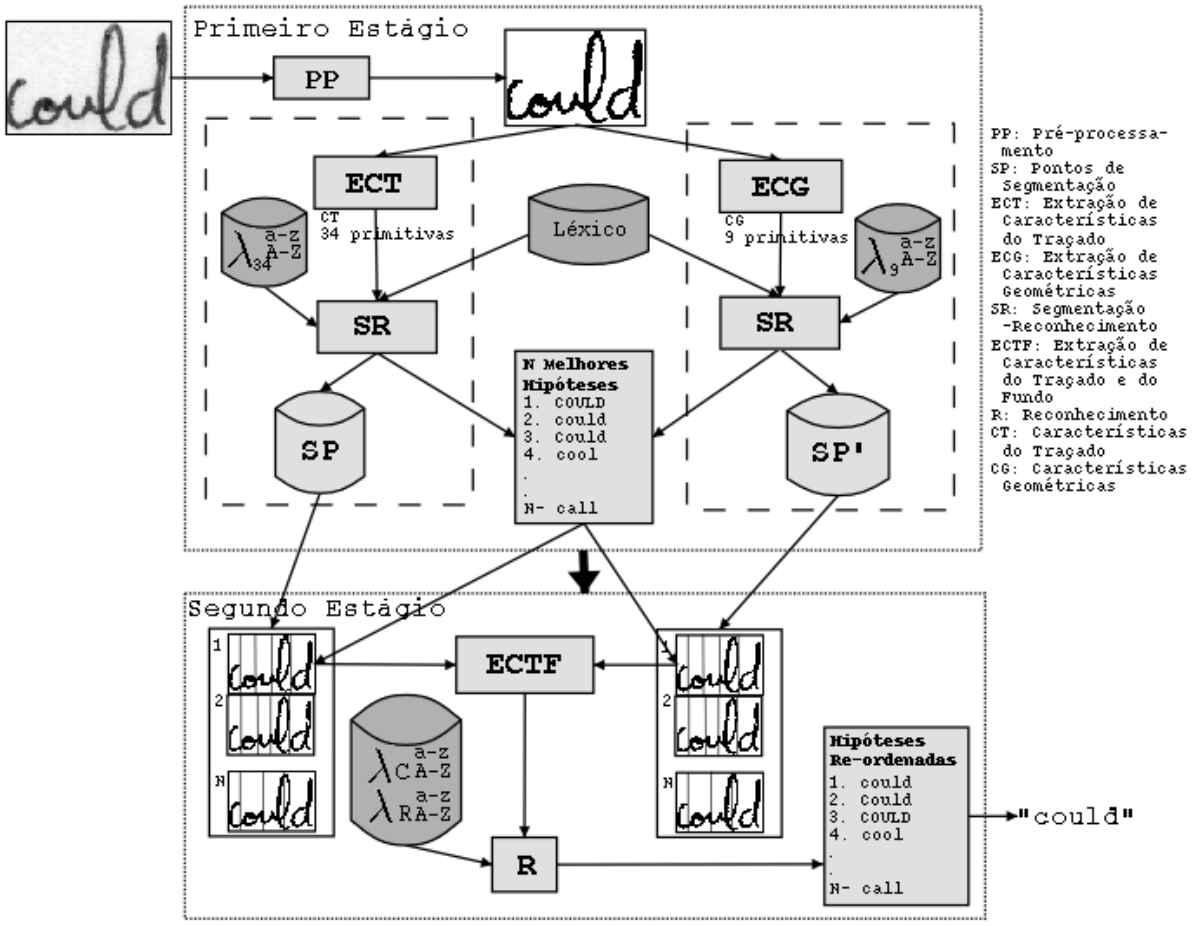


Figura 4.9: Visão geral do método proposto para a combinação de dois conjuntos de características utilizando MEMs distintos.

O segundo estágio constitui-se de uma única etapa, porém agora considera dois conjuntos de pontos de segmentação. Com isso, as informações de cada conjunto de características tornam-se complementares tanto para a estimação das melhores hipóteses no primeiro estágio, como para a verificação destas. A probabilidade de cada hipótese na verificação é reavaliada através da soma das probabilidades dos segmentos gerados com o conjunto de características CG, e as probabilidades dos segmentos gerados com o conjunto de características CT.

4.4.3 Combinação de Conjuntos de Características com MEMs Compartilhados

Uma outra alternativa para combinar os dois conjuntos de características no primeiro estágio é treinando-os num único MEM para cada classe. Com isso, são consideradas duas seqüências de observações tanto durante o treinamento, quanto para o reconhecimento e a extração dos pontos de segmentação. A Figura 4.10 traz uma visão geral sobre este método.

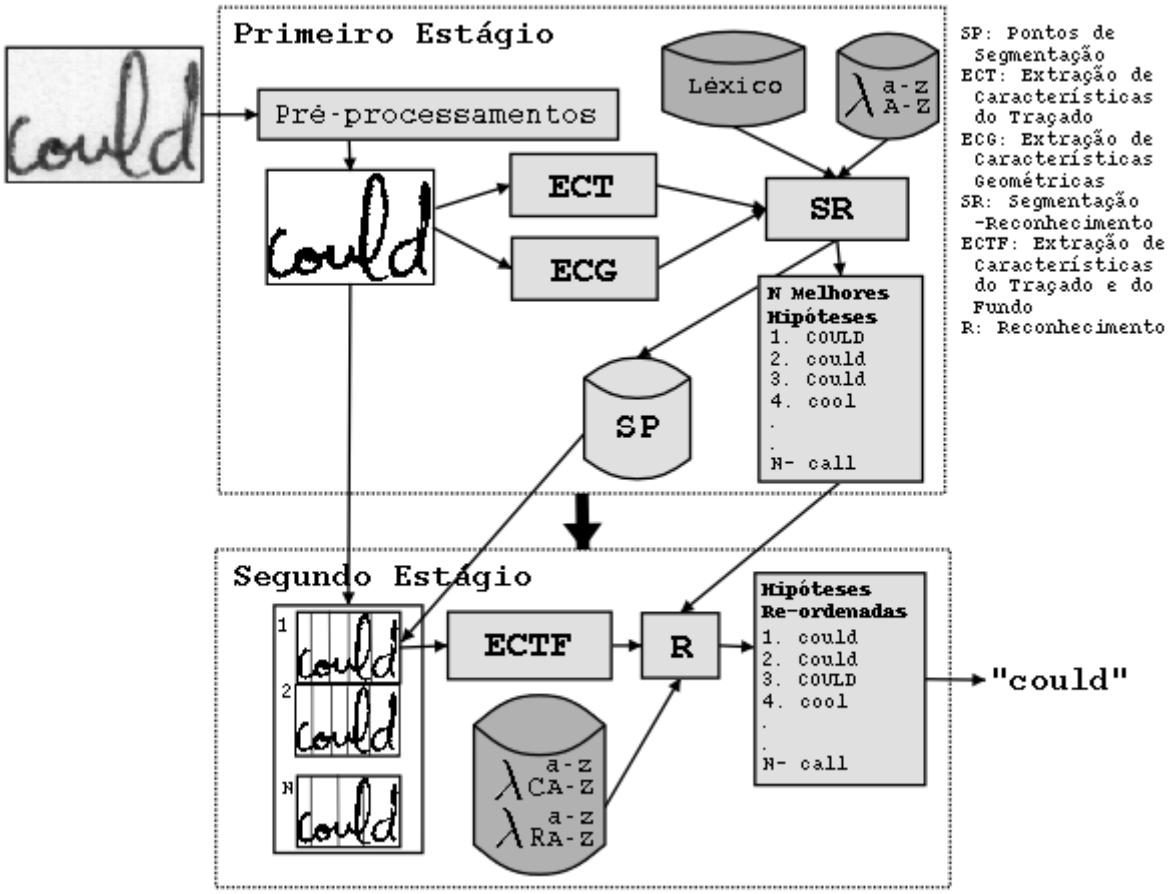


Figura 4.10: Visão geral do método proposto para a combinação de dois conjuntos de características utilizando MEMs treinados com os dois conjuntos.

Considerando que este sistema possui uma seqüência de observações para cada conjunto de características, representados por $O = O_1\ O_2\ \ldots\ O_T$ e $O' = O'_1\ O'_2\ \ldots\ O'_T$,

estas duas seqüências possuem o mesmo tamanho T. Como são consideradas duas seqüências de observações, então existem duas matrizes B para armazenar a distribuição de probabilidade de observação dos símbolos k e k' em cada estado j, sendo respectivamente estas matrizes representadas por B $= \{b_j(k)\}$ e B' $= \{b'_j(k')\}$, onde:

$$b_j(k) = P[v_k \text{ em } t | q_t = S_j], \qquad 1 \leq j \leq N, \quad 1 \leq k \leq M \tag{4.15}$$

$$b'_j(k') = P[v'_{k'} \text{ em } t | q_t = S_j], \qquad 1 \leq j \leq N, \quad 1 \leq k' \leq M' \tag{4.16}$$

Deve ser considerado que os conjuntos de características podem ser mapeados para *codebooks* de tamanhos diferentes, sendo que as variáveis M e M' representam o número de símbolos para cada conjunto de características. Os algorítmos para o treinamento e a avaliação dos modelos devem receber as alterações para avaliarem dois conjuntos de características. Basicamente, a probabilidade da observação dos símbolos no instante t, estando no estado i deve receber a seguinte atribuição:

$$b_i(O_t) = b_i(O_t) \times b'_i(O'_t), \; 0 \leq i \leq N, \; 0 \leq t \leq T \tag{4.17}$$

A re-estimação das matrizes B no algoritmo de Baum-Welch deve considerá-las separadamente. Re-escrevendo a Equação $4.3c$, a Equação 4.18 e a Equação 4.19 demonstram matematicamente esta re-estimação.

$$\bar{b}_j(l,k) = \frac{\sum_{t=1}^{T}{}^{s.t.O_t=v_k} \gamma_t(j + offset(l))}{\sum_{t=1}^{T} \gamma_t(j + offset(l))}, \qquad 1 \leq l \leq L \tag{4.18}$$

$$\bar{b'}_j(l,k') = \frac{\sum_{t=1}^{T}{}^{s.t.O'_t=v'_{k'}} \gamma_t(j+offset(l))}{\sum_{t=1}^{T} \gamma_t(j+offset(l))}, \qquad 1 \leq l \leq L \tag{4.19}$$

Com estas alterações propostas, apenas um conjunto de pontos de segmentação é considerado no segundo estágio. Isto deve-se ao fato desta abordagem absorver em MEMs comuns mais de um conjunto de características.

4.5 RESUMO

Neste capítulo foi apresentado o método para o reconhecimento de palavras manuscritas, sem restrições de grafia, e auxiliado por um léxico. O objetivo do método é reconhecê-las utilizando dois estágios. No primeiro, considera-se a palavra como um todo e são extraídas informações do seu traçado. Com isto, é feita uma segmentação implícita da palavra, gerando um *ranking* com as melhores hipóteses desta fase.

O segundo estágio é aplicado para complementar a classificação feita pelo primeiro, utilizando informações obtidas através da segmentação implícita anteriormente realizada. Com os pontos de segmentação, podem ser extraídas características do fundo de cada caracter, assim como é possível a combinação das informações das linhas e das colunas. Com mais informações disponíveis, é verificado o *ranking* criado pelo primeiro estágio, e se necessário, este é reordenado.

Dois conjuntos de características foram propostos para a realização da classificação/segmentação do primeiro estágio, um extraindo informações das transições do traçado para o fundo, e o outro informações geométricas da escrita. Duas alternativas para a combinação destas no primeiro estágio também foram propostas neste capítulo.

5 EXPERIMENTOS

Este capítulo apresenta todos os experimentos relacionados ao método proposto no Capítulo 4. Estes experimentos vão desde a validação do método para o reconhecimento de palavras a otimizações com o propósito de melhorar o desempenho do sistema.

Este capítulo traz também os experimentos necessários para validar o sistema na base de dados utilizada. Portanto, todos os experimentos para otimizar o sistema nesta base são apresentados neste capítulo.

5.1 DEFINIÇÃO DA BASE DE DADOS

Para seguir um protocolo, tornando possível comparar o desempenho com outros sistemas, foi escolhido o subconjunto da base IAM utilizado em [GÜNTER, 2004b]. Os resultados deste trabalho constam na Tabela 2.1.

Deste conjunto de dados, foram utilizados apenas os exemplos contendo palavras sem qualquer caractere diferente de letras alfabéticas (A-Z e a-z). Outra modificação foi a inclusão de todos os exemplos em caixa alta nos conjuntos de treinamento e validação, não os deixando no conjunto de teste. Isto deve-se ao fato do pequeno número de exemplos deste tipo, o que poderia causar problemas devido ao pequeno (e alguns casos nulo) número de exemplos para algumas classes.

No total, a base de dados utilizada nos experimentos contém 18.624 imagens. O conjunto de treinamento possui 12.651 exemplos, o de validação 3.168 e o de teste 2.805. O léxico considera todas as palavras ocorridas em toda esta base de dados, sendo que esta contém 3.717 palavras diferentes. A Tabela 5.1 traz dados detalhados sobre a quantidade de caracteres contida na base.

Tabela 5.1: Número de exemplos em cada classe no conjunto de dados utilizado (Base IAM).

Caracter	#	**Caracter**	#	**Caracter**	#	**Caracter**	#
a	6430	A	324	b	1316	B	191
c	2000	C	162	d	3308	D	110
e	11017	E	244	f	1897	F	90
g	1551	G	188	h	5163	H	201
i	5684	I	240	j	86	J	23
k	446	K	20	l	3266	L	137
m	2095	M	119	n	5928	N	226
o	6073	O	137	p	1616	P	144
q	57	Q	6	r	5138	R	147
s	5252	S	233	t	7814	T	401
u	2284	U	58	v	932	V	43
w	1698	W	199	x	174	X	1
y	1563	Y	35	z	36	Z	5

5.2 DEFINIÇÃO DO PRIMEIRO ESTÁGIO DO SISTEMA

Os experimentos mostrados nesta seção têm como objetivo ajustar um sistema, partindo do método para reconhecimento de cadeias de dígitos proposto em [BRITTO, 2001], até otimizá-lo para o reconhecimento de palavras manuscritas. Dentre estas otimizações, podemos citar a escolha dos pré-processamentos adequados, assim como o tamanho mais adequado para o *codebook*.

A estrutura do protocolo experimental primeiramente busca otimizar o primeiro estágio, e somente após todos os ajustes deste que o segundo estágio é avaliado. Nesta lógica, os experimentos com maiores taxas de reconhecimento são considerados os melhores, e portanto, os parâmetros são mantidos para os experimentos seguintes.

Partindo daquele trabalho, os números de estados dos MEMs foram estimados aplicando-se o método de Wang (Seção 3.2) nos caracteres da base de dados NIST, a mesma base utilizada em [BRITTO et al., 2004]. A Tabela 5.2 traz os valores exatos utilizados. Foram utilizados os valores mínimos obtidos. A escolha de estimar o número de

estados utilizando uma base de dados de caracteres deu-se pelo fato que vários trabalhos já demonstraram que o número de estados de tamanho fixo possui desempenho inferior aos métodos que estimam diferenças de tamanho entre as classes. Posteriormente, são avaliados outros conjuntos de dados para estimar os estados utilizando o mesmo método. Para o *codebook* foram utilizados 256 símbolos, já que este foi o melhor resultado encontrado por Britto, sendo que este valor é reavaliado experimentalmente.

Tabela 5.2: Número de estados para cada MEM utilizado no primeiro estágio. Número estimado com os caracteres isolados da base NIST.

Classe	**#**	**Classe**	**#**	**Classe**	**#**	**Classe**	**#**
a	10	n	12	A	12	N	13
b	12	o	12	B	10	O	11
c	12	p	12	C	8	P	9
d	11	q	10	D	9	Q	15
e	12	r	10	E	12	R	13
f	11	s	11	F	11	S	8
g	11	t	10	G	15	T	9
h	11	u	10	H	11	U	8
i	3	v	10	I	2	V	9
j	6	w	15	J	6	W	16
k	12	x	10	K	15	X	12
l	5	y	11	L	8	Y	10
m	14	z	10	M	14	Z	11

O primeiro experimento foi realizado apenas com o primeiro estágio do sistema e sem a aplicação de qualquer método de pré-processamento, tanto nas bases de treinamento e validação, como na base de teste. A Tabela 5.3 e a Tabela 5.4 trazem o número de caracteres utilizados para o treinamento e validação de cada classe, respectivamente. Todas as implementações do primeiro estágio utilizam este mesmo número de exemplos por classe.

Neste experimento o sistema alcançou 61,07% de reconhecimento, avaliando todo o léxico. O experimento mostrou também que a taxa de reconhecimento aumenta à

Tabela 5.3: Número de exemplos utilizados para o treinamento de cada classe (Base IAM).

Caracter	#	Caracter	#	Caracter	#	Caracter	#
a	4397	A	227	b	870	B	139
c	1342	C	129	d	2207	D	78
e	7510	E	184	f	1311	F	67
g	1052	G	126	h	3542	H	135
i	3853	I	182	j	66	J	11
k	288	K	13	l	2285	L	78
m	1476	M	71	n	3996	N	188
o	4020	O	103	p	1077	P	117
q	31	Q	5	r	3430	R	122
s	3513	S	148	t	5399	T	257
u	1543	U	46	v	641	V	36
w	1147	W	177	x	113	X	0
y	1040	Y	32	z	11	Z	2

Tabela 5.4: Número de exemplos utilizados para a validação de cada classe (Base IAM).

Caracter	#	Caracter	#	Caracter	#	Caracter	#
a	1158	A	64	b	251	B	38
c	373	C	21	d	626	D	29
e	1910	E	51	f	306	F	13
g	312	G	37	h	823	H	54
i	1007	I	38	j	8	J	3
k	78	K	5	l	486	L	44
m	348	M	30	n	1139	N	31
o	1097	O	30	p	320	P	13
q	5	Q	1	r	971	R	17
s	907	S	49	t	1264	T	69
u	388	U	11	v	179	V	5
w	269	W	14	x	20	X	0
y	280	Y	3	z	18	Z	2

medida em que diminui-se o tamanho do léxico. A Tabela 5.5 traz mais detalhes sobre estes experimentos. Estes experimentos também foram úteis para avaliar o desempenho

do léxico em árvore. Com o léxico linear o tempo médio de processamento [1] para cada palavra de entrada é cerca de 5 minutos, enquanto que com o léxico em árvore este tempo fica em cerca de 50 segundos. Portanto, o léxico em forma de árvore traz uma redução de cerca de 80% no tempo de processamento.

Tabela 5.5: Taxas de reconhecimento para o primeiro estágio do sistema com 256 símbolos e sem pré-processamentos. Top 1: melhor hipótese. Top 5: entre as cinco melhores hipósteses. Top 10: entre as dez melhores hipóteses.

Léx. (#)	**Top 1**	**Top 5**	**Top 10**
10	96,58	98,79	100,00
100	89,45	96,01	97,18
1000	74,37	88,23	91,55
3717	61,07	80,93	85,92

5.2.1 Definição dos Pré-processamentos

Após a avaliação do comportamento do sistema sem qualquer otimização, deve-se então aplicar novos métodos e observar as diferenças no desempenho do sistema. Nesta seção são avaliados os métodos de pré-processamento propostos em 4.2.1.

Os métodos propostos para a realização destas tarefas são avaliados gradativamente. Isto é, primeiramente é avaliada a diferença de desempenho dada a aplicação do método para a correção de inclinação vertical. Em seqüência são testados os métodos para correção da inclinação da linha de base e normalização do corpo da palavra, respectivamente. Os métodos que resultam em melhores resultados para o sistema são mantidos para os experimentos seguintes. Ou seja, se a correção de inclinação vertical apresenta melhores resultados, então esta é mantida, e os outros pré-processamentos são aplicados após a correção da inclinação vertical. É importante ressaltar que os métodos são aplicados em toda a base de dados, portanto é necessário treinar o sistema conforme a inclusão

[1] Experimentos realizados em um computador Pentium 4 com 1,5GHz de poder de processamento e 512MB de memória principal.

de um novo pré-processamento.

Como mencionado anteriormente, o primeiro teste realizado avaliou a aplicação de um corretor de inclinação vertical dos caracteres. Esta etapa mostrou um aumento significativo de 13% na taxa de reconhecimento, elevando de 61,07% para 74,08% a taxa de reconhecimento, quando avaliado todo o léxico. Maiores detalhes podem ser vistos na Tabela 5.6. Estes resultados mostraram uma grande contribuição deste pré-processamento, já que o aumento da taxa de reconhecimento foi muito significativo.

Tabela 5.6: Taxas de reconhecimento para o primeiro estágio do sistema com correção de inclinação vertical.

Léx. (#)	**Top 1**	**Top 5**	**Top 10**
10	97,07	99,00	100,00
100	93,33	96,64	97,14
1000	84,24	93,44	94,69
3717	74,08	88,84	92,01

Em seqüência ao experimento acima, foi testado o impacto da aplicação de um método para correção da inclinação da linha de base da palavra. Este é um pré-processamento muito utilizado em sistemas deste tipo, e o teste de sua influência neste sistema é um experimento importante. Neste experimento, com relação ao experimento anterior, a taxa de reconhecimento teve uma redução de 1,16%, decaindo de 74,08% para 72,92% quando utiliza-se todo o léxico. Com este resultado, a correção de linha de base não é utilizada nos experimentos seguintes. A Tabela 5.7 mostra estes resultados em detalhes.

Para finalizar esta série de experimentos, foi avaliado um método para normalização do corpo da palavra, já que este também é muito utilizado em sistemas de reconhecimento de palavras manuscritas. Assim como a correção da linha de base, este pré-processamento trouxe certa redução na taxa de reconhecimento. O sistema teve uma queda, utilizando todo o léxico, de 74,08% para 72,37%. Maiores detalhes sobre os

Tabela 5.7: Taxas de reconhecimento para o primeiro estágio do sistema com correção da linha de base e normalização do corpo da palavra, separadamente.

	Linha de base			**Norm. corpo**		
Léx. (#)	**Top 1**	**Top 5**	**Top 10**	**Top 1**	**Top 5**	**Top 10**
10	95,97	98,57	100,00	96,33	98,79	100,00
100	91,66	95,58	96,34	92,44	95,86	96,72
1000	82,88	91,19	93,40	82,99	92,44	93,94
3717	72,92	87,63	89,73	72,37	88,05	90,69

experimentos utilizando este método estão na Tabela 5.7.

Dados os resultados apresentados nesta seção, ficou definido que o único pré-processamento utilizado no sistema, com o conjunto de características original e com esta base de dados, é a correção de inclinação vertical. Com a aplicação desta o sistema teve um aumento significativo nas taxas de reconhecimento, enquanto que os outros pré-processamentos mostraram redução de desempenho quando aplicados no sistema já com correção de inclinação vertical.

5.2.2 Otimização do Tamanho do Codebook

O *codebook* é um conjunto de vetores contendo N centros que são utilizados para a quantização dos vetores de características em um símbolo do MEM. Existem vários algorítmos para esta tarefa, mas há uma grande dificuldade para estimar os parâmetros iniciais destes algorítmos.

Para estimar o *codebook*, este sistema utilizou o algorítmo *K-Means*. Para este é passado o parâmetro K que corresponde ao número *centroids*. O grande problema aqui é estimar qual o melhor valor para K, já que este possui influência direta nas taxas de reconhecimento do sistema.

Para encontrar o melhor número de *centroids*, duas maneiras são abordadas neste trabalho. Primeiro, na maneira supervisionada, são testados vários centros e aquele com a maior taxa de reconhecimento é considerado o melhor. Este método garante ao

sistema o melhor resultado, já que todos os valores são testados. Porém, como desvantagem, é um método exaustivo, já que para cada novo centro é preciso gerar novos MEMs, consumindo muito tempo para treinar e testar os novos modelos criados.

Outra maneira de selecionar os melhores centros é utilizando uma abordagem não-supervisionada. Para isso, foi proposto neste trabalho a utilização de Índices de Validação (IV) para avaliar diversos *codebooks* gerados. Estes métodos necessitam apenas ser aplicados na fase de Quantização Vetorial (QV), portanto não faz-se necessário o treinamento dos MEMs para cada centro gerado.

O objetivo com a utilização das duas abordagens é fazer uma avaliação dos métodos não-supervisionados. Tendo os resultados com a abordagem supervisionada, é possível comparar os resultados apresentados pelos IV. A Figura 5.1 traz um resumo sobre os resultados dos diferentes centros avaliados através do método supervisionado. A avaliação foi feita entre o intervalo 32 a 448, com um intervalo de 32 centros em cada iteração. Foi levado em consideração o sistema utilizando apenas a correção de inclinação vertical, dados os resultados apresentados na seção anterior. Os melhores resultados foram apresentados com 256, 288, 320 e 352. Suas taxas de reconhecimento considerando todo o léxico foram, respectivamente, 74,08%, 74,47%, 74,44% e 74,44%. Portanto, pode-se considerar o sistema com 288 símbolos como o melhor entre os testados, ficando cerca de 0,4% acima do sistema com 256 símbolos. Para os léxico de 10, 100 e 1.000 palavras, as taxas de reconhecimento com 288 símbolos foram de 96,01%, 92,87% e 84,01%, respectivamente.

O primeiro IV a ser testado foi o índice DB. Este foi testado no mesmo intervalo em foram feitos os testes supervisionados. O melhor resultado encontrado pelo algoritmo (valor mínimo) foi com 32 centros (Figura 5.2). A taxa de reconhecimento para este número de centros é 64,81%, cerca de 10% menor que a taxa de reconhecimento com 288 símbolos. Portanto, não foi possível encontrar uma relação com os resultados apresentados por este índice e pelo teste supervisionado.

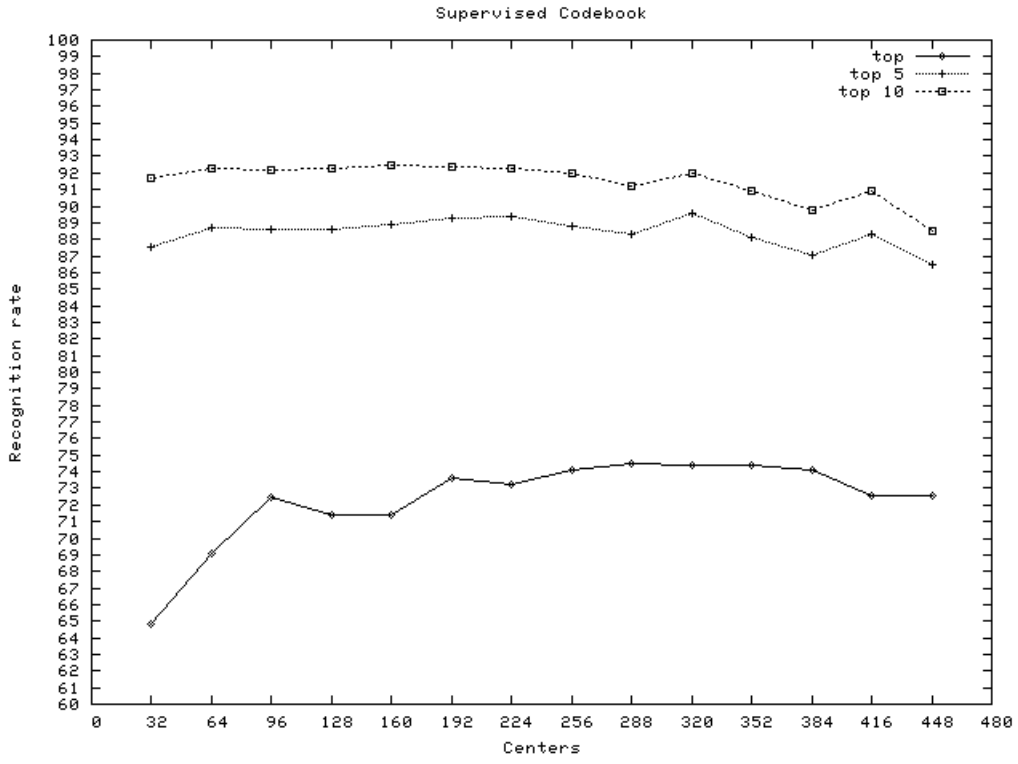


Figura 5.1: Avaliação do codebook através da maneira supervisionada.

O outro testado, o XB, também não produziu os resultados esperados. Como pode ser visto na Figura 5.3, a curva produzida por este método fica parecida com a produzida pelo DB. O melhor valor (valor mínimo) encontrado pelo XB foi 96, sendo que a taxa de reconhecimento com este número de símbolos é de 72,48%. É um valor melhor que o encontrado pelo DB, porém fica 3% abaixo do encontrado pelo supervisionado.

Portanto, apesar do método supervisionado ainda ser a melhor alternativas para maiores taxas de reconhecimento, o índice XB ainda pode ser aplicado para este propósito. Este produziu melhores resultados que o índice DB, e sua aplicação otimização do tamanho do *codebook* é uma alternativa a ser seguida caso deseje-se evitar o grande número de processamentos da abordagem supervisionada, porém com uma perda de 3% no desempenho.

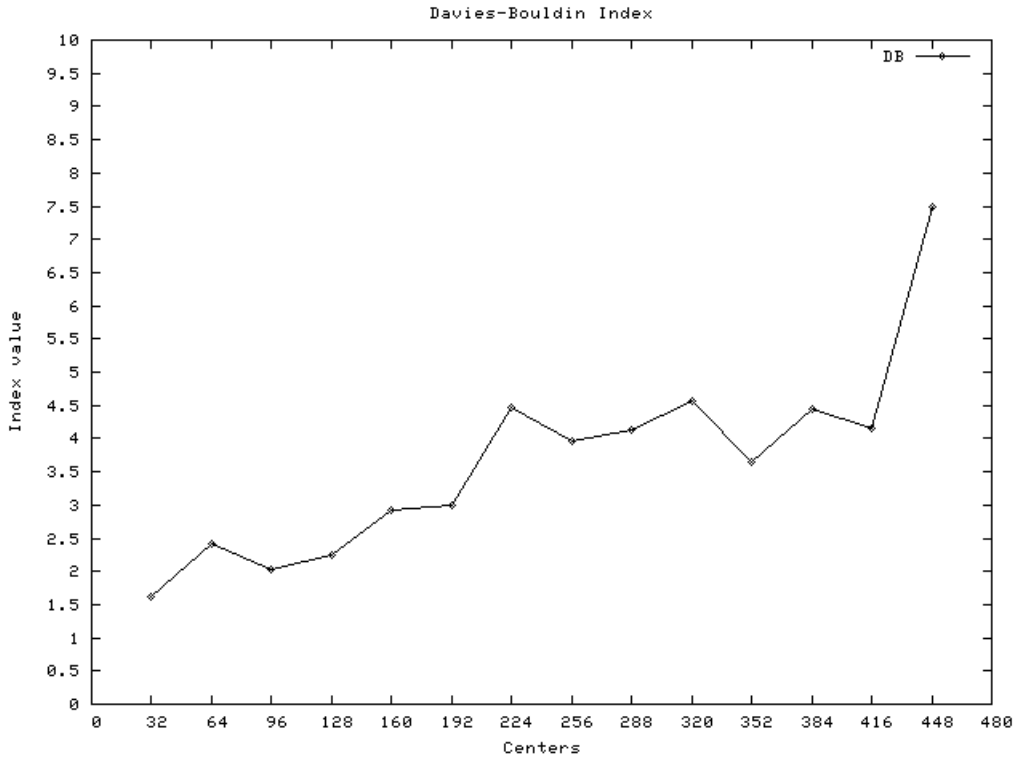


Figura 5.2: Avaliação do codebook utilizando o índice DB.

5.2.3 Otimização do Número de Estados dos MEMs

Outro fator importante, que pode contribuir para a melhoras no desempenho, é a otimização dos parâmetros dos MEM. Um destes parâmentros é o número de estados de cada modelo, que pode contribuir tanto para maiores taxas de reconhecimento, como para a extração de melhores pontos de segmentação.

Essa definição do número de estados não é uma tarefa trivial. Primeiro é necessário tomar a decisão entre uma topologia com tamanho fixo, ou com tamanho variado. Trabalhos recentes da literatura ([BUNKE & ZIMMERMANN, 2002a] e [BRITTO, 2001]) já demonstraram que o tamanho fixo geralmente não produz boas taxas de reconhecimento. Porém, como os MEM modelam caracteres, então é necessário possuir a informação sobre os caracteres isolados para estimar tamanhos variados.

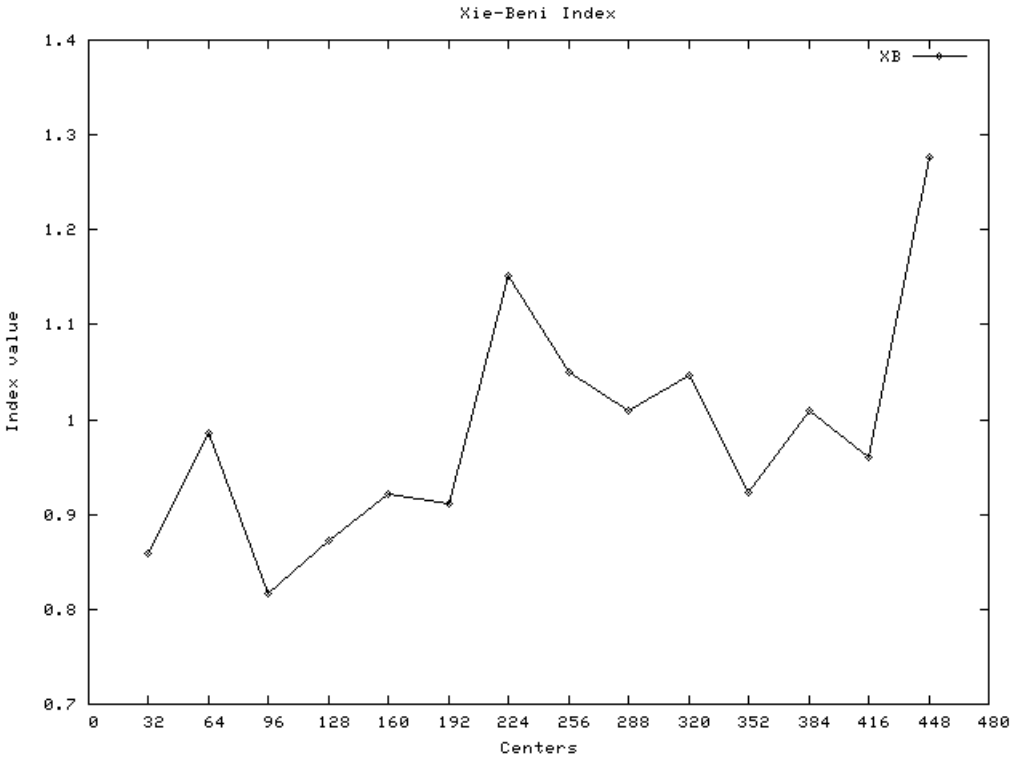


Figura 5.3: Avaliação do codebook utilizando o índice XB.

Devido ao desempenho inferior com número fixo de estados, este trabalho vai apenas abordar experimentos para a estimação de números de estados heterogêneos. O método escolhido para esta tarefa foi apresentado na Seção 3.2. Os valores mínimos encontrados por este método, utilizando os caracteres isolados da base NIST, produziram os números de estados utilizados nos experimentos anteriores.

São propostas aqui duas abordagens para otimizar os números de estados utilizados. A primeira alternativa é utilizar uma outra base de dados para estimar estes números. A base escolhida é a base IRONOFF, que é uma base Européia, assim como a base IAM, e pode causar menores diferenças entres o exemplos que, por exemplo, a base NIST. Os experimentos foram realizados utilizando os parâmentros já otimizados anteriormente, ou seja, utilizando apenas correção de inclinação vertical e com 288 símbolos

nos MEMs. Os resultados podem ser vistos na Tabela 5.10. Para todo o léxico, este experimento trouxe uma queda de 3,24% na taxa de reconhecimento, se comparado ao sistema com melhores taxas de reconhecimento. O número de estados utilizados nestes experimentos podem ser visualizados na Tabela 5.8.

Tabela 5.8: Número de estados estimados com a base IRONOFF.

Classe	**#**	**Classe**	**#**	**Classe**	**#**	**Classe**	**#**
a	12	n	7	A	10	N	13
b	9	o	11	B	12	O	13
c	13	p	7	C	11	P	7
d	10	q	10	D	11	Q	14
e	11	r	7	E	11	R	10
f	9	s	11	F	8	S	12
g	11	t	8	G	11	T	12
h	8	u	8	H	10	U	10
i	4	v	10	I	4	V	12
j	8	w	12	J	11	W	17
k	9	x	8	K	12	X	11
l	5	y	9	L	10	Y	10
m	11	z	11	M	13	Z	12

A segunda alternativa proposta para a otimização dos números de estados é estimá-los na própria base de palavras. Para isso, são necessários os pontos de segmentação para limitar os caracteres. Os pontos utilizados foram extraídos com a implementação otimizada do primeiro estágio. A Tabela 5.10 traz um resumo dos taxas de reconhecimento utilizando estes números de estados, e a Tabela 5.9 traz os estados utilizados em cada classe. Os resultados apresentados por esta abordagem também foram inferiores aos apresentados anteriormente, trazendo uma queda de 4,6% na taxa de reconhecimento ao utilizar todo o léxico.

Após uma análise dos resultados apresentados, empiricamente foi observada uma relação entre o número médio de estados e a taxa de reconhecimento. O sistema com maiores taxas de reconhecimento, possui a maior média no número de estados (10,58). O

Tabela 5.9: Número de estados estimados com a base IAM.

Classe	**#**	**Classe**	**#**	**Classe**	**#**	**Classe**	**#**
a	9	n	9	A	13	N	12
b	9	o	8	B	10	O	10
c	8	p	10	C	10	P	11
d	9	q	13	D	8	Q	14
e	8	r	6	E	12	R	13
f	7	s	7	F	10	S	8
g	10	t	6	G	10	T	8
h	8	u	8	H	7	U	12
i	3	v	8	I	2	V	7
j	6	w	10	J	8	W	12
k	10	x	7	K	14	X	10
l	4	y	8	L	10	Y	11
m	10	z	8	M	8	Z	10

Tabela 5.10: Sistema utilizando o número de estados baseado em diferentes bases de dados.

	NIST			**IRONOFF**			**IAM**		
Léx. (#)	**Top 1**	**Top 5**	**Top 10**	**Top 1**	**Top 5**	**Top 10**	**Top 1**	**Top 5**	**Top 10**
10	**96,01**	98,54	100,00	95,97	98,07	100,00	94,47	98,43	100,00
100	**92,87**	95,79	96,54	92,92	95,61	96,58	90,52	94,97	95,86
1000	**84,01**	92,62	93,98	82,39	91,94	93,90	80,07	90,16	92,30
3717	**74,47**	88,27	91,19	71,23	87,49	90,41	69,87	84,95	88,52

conjunto de estados com a menor média (9,17), foi o conjunto que apresentou as menores taxas de reconhecimento. Conseqüentemente, os estados estimados na base IRONOFF, que alcançaram resultados intermediários, possuem média de 10,12 estados por classe.

5.2.4 Otimização do Treinamento

Outro ponto muito importante, num sistema de reconhecimento de padrões, é a otimização do treinamento do classificador. Neste caso, há a necessidade de avaliar se os MEMs estão com seus parâmetros bem estimados. Um dos fatores que podem levar

a uma má estimação destes parâmetros é a falta de dados na base de treinamento. Esta deve conter exemplos suficientes para que o sistema possa absorver as diferenças intra e inter-classes.

Além do sistema não absorver as diferenças, a falta de exemplos também pode causar ruídos. MEMs discretos necessitam ter seus parâmetros inicializados. Esta inicialização dá-se de forma arbitrária, e o número insuficiente de dados pode fazer com que estes valores sejam pouco alterados durante o treinamento.

Outro problema que pode existir é o desequilíbrio de exemplos entre as classes. Este é um problema também existente no mundo real, já que realmente alguns caracteres possuem um número realmente inferior a outros. Porém, para o treinamento, é necessário que as classes possuam um número mínimo de exemplos, já que a quantidade de parâmetros dos MEMs é igual para todas as classes.

Propõe-se aqui a inserção de exemplos de caracteres isolados para solucionar o problema acima. A idéia é testar o impacto da inserção de dados em todas as classes, assim como o impacto da inserção de dados nas classes com menos exemplos. Como a base utilizada anteriormente não possui exemplos suficientes de caracteres isolados para isso, torna-se necessária então a utilização de uma base de dados contendo caracteres isolados. A base escolhida para esta tarefa é a base IRONOFF, por ser uma base Européia, e também por seu número de exemplos para cada classe não ser tão grande, já que a inserção de um grande número de exemplos pode descaracterizar os MEMs. A Tabela 5.11 traz em detalhes o número de exemplos para cada classe.

O primeiro experimento realizado considerou a inserção de dados em todas as classes, sendo que 70% dos caracteres foram inseridos na base de treinamento, e 30% na base de validação. Com este experimento a taxa de reconhecimento para todo o léxico apresentou uma queda de 3,98%. Já o segundo experimento considerou a inserção de exemplos de caracteres da base IRONOFF apenas nas classes com poucos exemplos (A-Z, j, q, x, z), seguindo a mesma proporção de inserção utilizado no experimento anterior. A

Tabela 5.11: Número de exemplos de caracteres isolados na base IRONOFF.

Caracter	#	**Caracter**	#	**Caracter**	#	**Caracter**	#
a	412	A	409	b	411	B	412
c	411	C	412	d	411	D	411
e	412	E	412	f	410	F	412
g	409	G	412	h	411	H	411
i	410	I	411	j	409	J	411
k	411	K	410	l	410	L	411
m	409	M	412	n	412	N	411
o	412	O	410	p	412	P	412
q	412	Q	408	r	412	R	410
s	412	S	409	t	410	T	412
u	410	U	411	v	411	V	410
w	412	W	409	x	411	X	411
y	412	Y	411	z	411	Z	409

quantidade de exemplos para cada classe pode ser visto na Tabela 5.1. Este experimento apresentou um aumento de 0,29%, que é um aumento pequeno, porém é considerado como um avanço do sistema. Os resultados apresentados por ambos experimentos podem ser vistos na Tabela 5.12.

Tabela 5.12: Sistema utilizando a base de dados IRONOFF para otimização do treinamento. Resultados para a inclusão de dados em todas as classes e em apenas as classes com poucos exemplos.

	Todas as classes			**Classes selecionadas**		
Léx. (#)	**Top 1**	**Top 5**	**Top 10**	**Top 1**	**Top 5**	**Top 10**
10	97,65	99,57	100,00	**96,68**	98,93	100,00
100	92,62	97,33	98,18	**93,01**	96,47	96,83
1000	81,46	92,55	94,97	**83,64**	92,48	93,33
3717	70,55	86,99	90,12	**74,76**	87,81	90,77

5.3 CRIAÇÃO DE UMA BASE DE DADOS DE CARACTERES ISOLADOS

Como o objetivo do segundo estágio do sistema é trabalhar com informações de caracteres isolados, ele necessita de uma base com este tipo de dado para que seu treinamento seja feito. Para possuir uma base de caracteres com os mesmos dados da base de palavras utilizadas no primeiro estágio, este foi utilizado para segmentá-las automaticamente, e então criar uma base de caracteres com as mesmas características que as palavras.

Foram segmentadas todas as palavras contidas nos conjuntos de treinamento e validação, e o pré-processamento realizado previamente corrigiu apenas a inclinação vertical dos caracteres. Após a segmentação automática, os caracteres foram verificados manualmente. Foram excluídos os exemplos com algum tipo de erro na segmentação para diminuir os ruídos no treinamento, assim como a confusão entre as classes. Após todo este processo de segmentação automática e verificação manual, o número de exemplos em cada classe pode ser visto na Tabela 5.13.

Tabela 5.13: Número de exemplos em cada classe no conjunto de dados criada através da segmentação das palavras.

Caracter	#	**Caracter**	#	**Caracter**	#	**Caracter**	#
a	3710	A	215	b	735	B	131
c	1188	C	120	d	1963	D	72
e	6391	E	174	f	990	F	58
g	864	G	115	h	3020	H	120
i	2853	I	159	j	61	J	9
k	257	K	10	l	2019	L	68
m	1298	M	67	n	3360	N	176
o	3202	O	97	p	960	P	112
q	27	Q	5	r	2622	R	119
s	2717	S	134	t	4600	T	231
u	1374	U	43	v	568	V	34
w	899	W	155	x	108	X	1
y	853	Y	31	z	9	Z	1

Esta base é utilizada para o treinamento e para a validação dos MEMs utilizados nos experimentos com o segundo estágio. Para isto, os exemplos foram separados arbitrariamente em 70% destes para treinamento, e 30% para validação.

5.4 DEFINIÇÃO DO SEGUNDO ESTÁGIO DO SISTEMA

Assim como foi feito com o primeiro estágio, esta seção destina-se a explicar as definições do estágio de verificação do método proposto (Seção 4.3). Os primeiro parâmetros utilizados são baseados nos melhores resultados obtidos em [BRITTO, 2001], e a partir destes parâmetros é que são feitas as otimização. O único pré-processamento utilizado corrige a inclinação vertical dos caracteres.

Primeiramente, a partir dos pontos de segmentação extraídos com o primeiro estágio, e utilizando modelos treinados com 256 símbolos, a verificação é avaliada conforme a Equação 4.4. O próximo passo é verificar o melhor *codebook* para este conjunto de dados, testando diferentes valores para este.

Assim que o sistema estiver com o *codebook* otimizado, os experimentos seguintes tem como objetivo o ajuste das informações utilizadas na verificação. Sendo assim, são testadas diferentes combinações de probabilidades, assim como o número de hipóteses verificadas.

A mesma otimização do treinamento utilizada no primeiro estágio é testada também para o segundo. Ou seja, a inclusão de dados isolados, com o objetivo de otimizar o treinamento, através de um melhor balanceamento de exemplos entre as classes, também é avaliada para a etapa da verificação.

Para finalizar esta etapa de experimentos, um limiar de decisão é proposto para fazer a verificação das hipóteses. Este limiar tem por objetivo selecionar as palavras que devem ser verificadas.

5.4.1 Adaptação do Segundo Estágio

Esta seção contém o primeiro experimento realizado para analisar o desempenho da verificação nos resultados apresentados pelo primeiro estágio. O objetivo deste experimento é avaliar o desempenho do método proposto sem otimizações, para então em seqüência avaliar passo a passo a inclusão destas.

Os modelos utilizados foram treinados utilizando a base descrita na Seção 5.3, considerando 256 símbolos. Os estados dos MEMs de colunas são os mesmos apresentados na Tabela 5.2, ou seja, os estados estimados nos caracteres segmentados da base IAM. Para os modelos de linhas também foi necessário estimar os estados a partir das observações das linhas. Os estados para os MEMs de linhas podem ser visualizados na Tabela 5.14.

Tabela 5.14: Número de estados para cada MEM utilizado no primeiro estágio.

Classe	**#**	**Classe**	**#**	**Classe**	**#**	**Classe**	**#**
a	8	n	7	A	13	N	10
b	14	o	8	B	13	O	9
c	7	p	10	C	10	P	15
d	16	q	10	D	12	Q	17
e	10	r	8	E	11	R	13
f	10	s	9	F	9	S	11
g	11	t	14	G	8	T	12
h	13	u	7	H	11	U	9
i	8	v	8	I	9	V	11
j	15	w	9	J	6	W	13
k	14	x	9	K	15	X	10
l	15	y	13	L	7	Y	9
m	8	z	14	M	11	Z	10

A verificação, considerando 10 hipóteses, utilizou nesta etapa a informação apresentada na Equação 4.4, ou seja, a combinação de informações das linhas e das colunas dos caracteres. A taxa de reconhecimento nesta etapa, considerando todo o léxico, foi

de 61,67%. Para léxicos de 10, 100 e 1000 palavras, as taxas de reconhecimento foram de 90,80%, 82,60% e 69,91%, respectivamente.

Utilizando apenas esta informação, com estes parâmetros, o estágio de verificação resultou numa queda de 13,09%, utilizando todo o léxico. Ao reduzir o léxico, o sistema também apresentou quedas na taxa de reconhecimento. Em média, a queda na taxa de reconhecimento foi de 10,78%.

5.4.2 Otimização do Codebook

A primeira otimização considerada para esta etapa é a adequação do *codebook*. Esta etapa é importante para ajustar a informação computada pelo estágio de verificação, e maximizar as taxas de reconhecimento antes da avaliação de combinações das informações de ambos os estágios.

Com isso, considerando a combinação de informações de linhas e colunas, a maior taxa de reconhecimento foi alcançada considerando 288 símbolos. Com este número de símbolos, o estágio de verificação possui uma taxa de reconhecimento de 62,92%, ficando com uma taxa 1,25% maior que considerando apenas 256 símbolos. A Figura 5.4 mostra um gráfico sobre os resultados dos diferentes valores testados.

5.4.3 Avaliação de Diferentes Combinações de Informações para a Verificação

Os experimentos já realizados com este estágio do sistema consideraram informações de linhas e colunas dos segmentos. O objetivo desta seção é mostrar a avaliação de outras informações, com o objetivo de aumentar a taxa de reconhecimento do sistema. Todas as taxas de reconhecimento destes experimentos estão na Tabela 5.15.

Primeiramente, avaliou-se o desempenho do sistema utilizando informações de linhas e colunas separadamente. Utilizando apenas características das colunas dos caracteres, a verificação passa a reconhecer 64,88% das palavras. Apenas com características

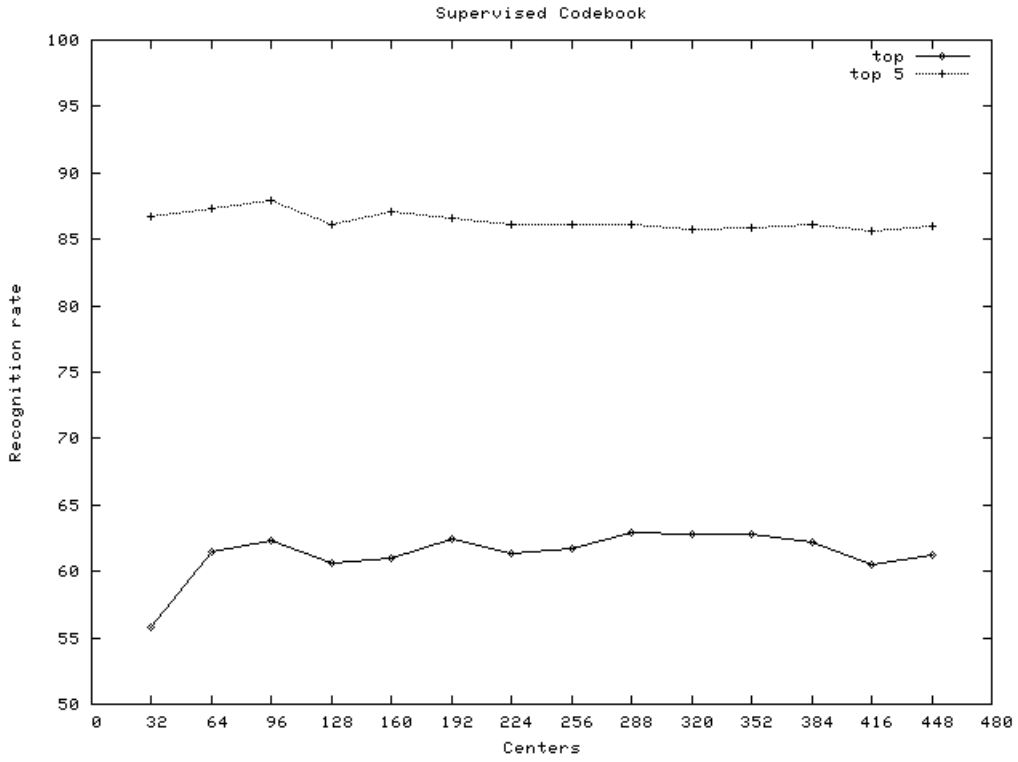


Figura 5.4: Avaliação do codebook do segundo estágio através da maneira supervisionada.

de linhas, a taxa de reconhecimento deste estágio fica em 49,55%. Como mostrado anteriormente, a taxa de reconhecimento para a combinação destas informações foi de 62,96%.

Tabela 5.15: Taxas de reconhecimento para o estágio de verificação com 288 símbolos.

Léxico	**3.717**		**1.000**		**100**		**10**	
Informação	**Top 1**	**Top 5**	**Top 1**	**Top 5**	**Top 1**	**Top 5**	**Top 1**	**Top 5**
Coluna + Linha	62,96	86,13	71,87	90,52	84,06	94,26	91,62	97,75
Coluna	64,88	84,24	73,37	88,95	83,07	93,44	91,16	97,97
Linha	49,55	84,99	55,94	89,45	72,41	92,98	78,79	96,34
Col + Lin + 1º Est.	73,58	87,38	82,46	91,16	90,87	94,94	95,54	98,47
Coluna + 1º Est.	73,51	86,34	81,35	90,74	90,27	94,87	95,01	98,75
Linha + 1º Est.	65,20	86,52	73,41	90,62	87,27	93,97	93,80	97,65
Média	64,95	85,93	73,07	90,24	84,66	94,07	90,99	97,86
Desvio Padrão	8,82	1,13	9,51	0,85	6,78	0,78	6,23	0,84

Outro tipo de informação que pode auxiliar no reconhecimento são as probabilidades computadas no primeiro estágio. Combinando as probabilidades do primeiro estágio com as probabilidades das colunas dos caracteres, computadas no segundo estágio, a taxa de reconhecimento fica em 73,51%. Já combinado a informação do primeiro estágio com probabilidades calculadas através de informações de linhas e colunas, a taxa de reconhecimento é de 73,58%. O pior resultado fica para a combinação de informação das linhas com as informações do primeiro estágio, com apenas 65,20% de reconhecimento.

Esta seção mostrou que a combinação das probabilidades do primeiro estágio com as da verificação aumenta em média 11,63%, porém ainda há uma redução desta em relação ao primeiro estágio.

5.4.4 Otimização do Treinamento

Assim como no primeiro estágio, a inclusão de novos exemplos na base de treinamento pode ser uma alternativa para o aumento do peso da verificação no sistema. O desequilíbrio de exemplos para treinar os modelos pode trazer ruídos para o reconhecimento deste estágio. Estes ruídos são causados por parâmetros mal treinados dos MEM.

Para otimizar o treinamento, os caracteres isolados da base IRONOFF são inseridos nas bases de treinamento e validação. E as duas abordagens utilizadas no primeiro estágio (Seção 5.2.4) são também testadas para o segundo estágio. Com a inserção de exemplos em todas as classes, a taxa de reconhecimento média ficou em 59,08%, mostrando uma redução 5,87% na taxa de reconhecimento. A Tabela 5.16 traz em detalhes todas as taxas de reconhecimento para este experimento.

A segunda abordagem de inserção de dados, onde apenas as classes com menor número de exemplos recebem dados da base IRONOFF, trouxe um aumento de 0,45% na taxa média de reconhecimento. Porém, ao considerar apenas as informações das colunas para a verificação, há um aumento de 1,75%, e com a combinação de informação das

Tabela 5.16: Taxas de reconhecimento para o estágio de verificação com 288 símbolos, inserindo exemplos em todas as classes selecionadas.

Léxico	**3.717**		**1.000**		**100**		**10**	
Informação	**Top 1**	**Top 5**	**Top 1**	**Top 5**	**Top 1**	**Top 5**	**Top 1**	**Top 5**
Coluna + Linha	56,40	85,63	67,02	90,48	78,22	94,69	90,84	98,50
Coluna	51,94	79,82	60,46	85,13	70,30	91,12	82,74	97,15
Linha	48,12	84,56	55,76	89,95	70,98	93,76	78,32	97,00
Col + Lin + 1º Est.	69,77	87,59	79,96	92,08	90,01	95,79	96,22	99,07
Coluna + 1º Est.	63,74	83,74	73,69	89,27	84,31	94,51	93,37	98,86
Linha + 1º Est.	64,53	87,02	72,44	91,37	88,34	95,26	94,44	98,61
Média	59,08	84,73	68,22	89,71	80,36	94,19	89,32	98,20
Desvio Padrão	8,29	2,81	8,97	2,46	8,56	1,65	7,17	0,89

colunas com as informações do primeiro estágio, há um aumento de 2,96%.

Tabela 5.17: Taxas de reconhecimento para o estágio de verificação com 288 símbolos, inserindo exemplos apenas nas classes selecionadas.

Léxico	**3.717**		**1.000**		**100**		**10**	
Informação	**Top 1**	**Top 5**	**Top 1**	**Top 5**	**Top 1**	**Top 5**	**Top 1**	**Top 5**
Coluna + Linha	64,17	87,17	72,69	91,41	85,09	95,29	93,30	98,93
Coluna	66,63	86,56	77,72	91,16	88,34	95,04	94,19	98,75
Linha	47,24	84,95	53,94	89,73	69,30	93,58	76,72	96,86
Col + Lin + 1º Est.	74,54	88,38	82,81	92,05	91,76	95,79	96,86	99,32
Coluna + 1º Est.	**76,47**	**87,98**	**84,60**	**92,62**	**92,55**	**95,69**	**96,61**	**99,25**
Linha + 1º Est.	63,35	86,95	71,48	91,01	87,09	94,69	94,12	98,54
Média	65,40	87,00	73,87	91,33	85,69	95,01	91,97	98,61
Desvio Padrão	10,42	1,21	11,08	0,99	8,51	0,81	7,61	0,91

5.4.5 Otimização do Número de Hipóteses

O objetivo desta seção é avaliar a verificação de diferentes números de hipóteses. Para isso, considera-se o melhor resultado alcançado pelo estágio de verificação, que utiliza a combinação das probabilidades do primeiro estágio com as probabilidades das colunas dos segmentos, e modelos otimizados tanto no número de símbolos

quanto no treinamento.

Como a lista de hipóteses geradas pelo primeiro estágio contém dez palavras, foram avaliadas as verificações para duas até dez hipóteses. A Figura 5.5 traz em detalhes estes experimentos. Pôde ser observado que o melhor resultado é alcançado através da verificação de dez hipóteses, já que a taxa de reconhecimento cai quando o número de hipóteses é reduzido. A exceção fica para o léxico de 100 palavras, onde o melhor resultado foi apresentado com cinco hipóteses.

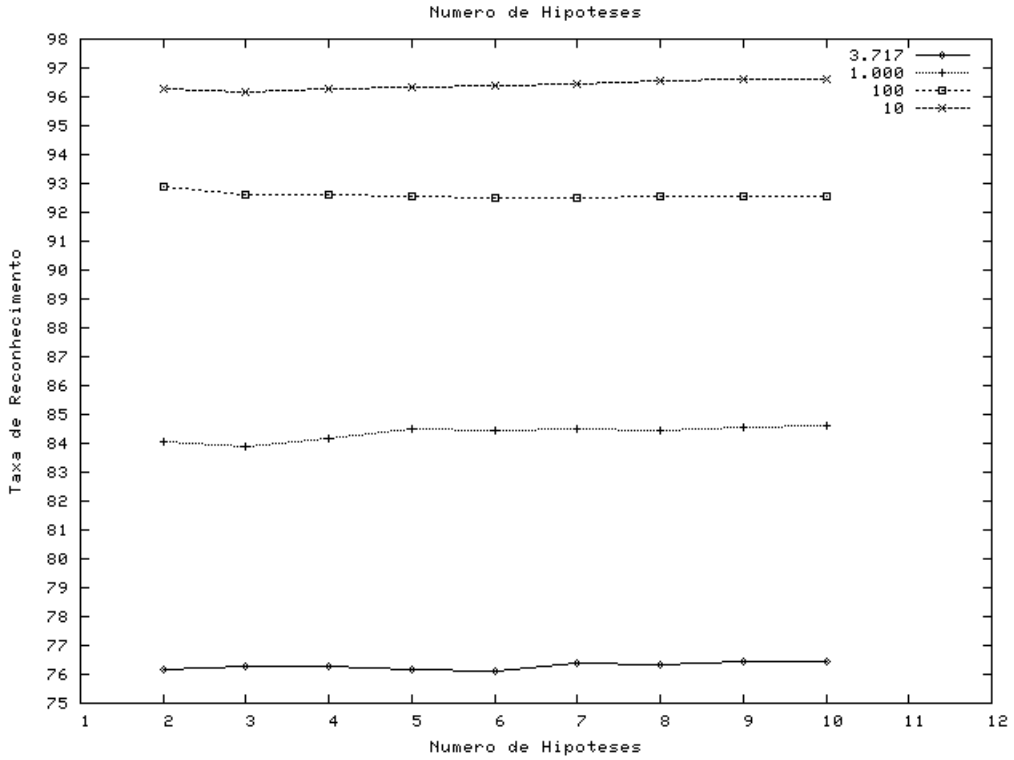


Figura 5.5: Taxas de reconhecimento para o estágio de verificação para diferentes números de hipóteses.

5.4.6 Limiar de Decisão

Como a taxa de reconhecimento apresentada isoladamente pelo segundo estágio é menor que a taxa de reconhecimento do primeiro estágio, foi proposta a utilização de um limiar para selecionar os exemplos a serem verificados. O objetivo deste limiar é verificar apenas as palavras onde o reconhecimento realizado pelo primeiro estágio não possui um certo nível de certeza, tornando necessário a utilização do estágio de verificação.

O limiar de decisão proposto é aplicado entre a primeira e a segunda hipótese. Se a diferença entre estas duas for maior que o valor do limiar, então a palavra possui o nível de certeza necessário e não é verificada. Caso contrário, é feita a verificação de todas as N hipóteses.

Tabela 5.18: Taxas de reconhecimento para o estágio de verificação utilizando o limiar de decisão.

Léx. \ **Limiar**	**0,001**	**0,002**	**0,003**	**0,004**	**0,005**
10	96,93	97,22	97,43	97,43	97,40
100	93,44	93,55	93,73	93,87	93,83
1.000	85,67	85,92	86,02	86,06	85,92
3.717	77,22	**78,00**	77,68	77,79	77,57

Tabela 5.19: Porcentagem de palavras verificadas com a utilização do limiar de decisão.

Léx. \ **Limiar**	**0,001**	**0,002**	**0,003**	**0,004**	**0,005**
10	1,39	2,42	3,35	4,13	4,92
100	3,74	6,67	9,48	12,16	14,65
1.000	10,55	17,65	24,21	30,55	37,61
3.717	16,40	28,66	39,96	50,80	60,29

Dados os resultados apresentados na Tabela 5.18, foi demonstrado que o limiar pode aumentar em até 1,53% a taxa de reconhecimento do sistema, se considerarmos o léxico de 3.717 palavras. O limiar que apresentou as maiores taxas de reconhecimento foi 0,002, sendo que foram testados valores entre 0,001 e 0,005. Com isso, a taxa de

reconhecimento do sistema, utilizando a verificação selecionada por um limiar, é de 78%. Isto faz com que o estágio de verificação aumente em 3,24% a taxa de reconhecimento final do sistema. A Tabela 5.19 traz um resumo do número de palavras verificadas com a utilização do limiar de decisão.

Para os outros léxicos, pode-se dizer que o melhor valor de limiar foi 0,004. Com isso, a taxa de reconhecimento para o léxico com 10 palavras foi de 97,43%, para 100 palavras foi de 93,87%, e para 1.000 palavras foi de 86,06%.

5.4.7 Resumo e Avaliação da Verificação

A Tabela 5.20 traz um resumo dos resultados apresentados pelo método, tornando possível visualizar o impacto da verificação no sistema. Para todos os léxicos a etapa de verificação trouxe aumentos nas taxas de reconhecimento, com este aumento ficando na média de 1,80%.

Tabela 5.20: Taxas de reconhecimento alcançadas apenas pelo primeiro estágio, e com a inclusão do segundo estágio.

	Primeiro Estágio		**Segundo Estágio**	
Léx. (#)	**Top 1**	**Top 5**	**Top 1**	**Top 5**
10	96,68	98,93	**97,43**	99,18
100	93,01	96,47	**93,87**	96,04
1000	83,64	92,48	**86,06**	93,01
3717	74,76	87,81	**78,00**	88,16

Pode ser notado que o impacto da verificação cresce proporcionalmente ao tamanho do léxico. Os maiores ganhos em reconhecimento foram com todo o léxico, sendo que a taxa de reconhecimento aumentou 3,24%. Para os léxicos de 10, 100 e 100 palavras, os aumentos foram de 0,75%, 0,81% e 2,42%, demonstrando que a influência da verificação aumenta conforme o léxico.

5.5 AVALIAÇÃO DO CONJUNTO DE CARACTERÍSTICAS GEOMÉTRICAS

Na Seção 4.2.2.2 foi apresentado um outro conjunto de características para ser utilizado pelo primeiro estágio. Estas primitivas já apresentaram bons resultados em outros trabalhos, e o objetivo é avaliá-las para a segmentação e a classificação das palavras no primeiro estágio.

Como está sendo avaliado um novo conjunto de primitivas, os experimentos realizados na Seção 5.2, com algumas exceções, deverão ser realizados também para este conjunto. Devem ser avaliados os impactos dos pré-processamentos, assim como a adaptação do *codebook* com o valor que melhor se adapta a estas primitivas. Para finalizar, será demonstrado o impacto da inclusão de caracteres da base IRONOFF para treinar as classes com menor número de exemplos, e os resultados da verificação para esta implementação do primeiro estágio.

Para a avaliação dos resultados, um primeiro experimento foi realizado sem considerar otimizações. Isto quer dizer que as imagens utilizadas não passaram por qualquer pré-processamento, e o *codebook* foi gerado considerando 256 símbolos. Estes resultados estão apresentados na Tabela 5.21.

Comparando estes resultados com os demonstrados na Tabela 5.5, pode ser observado que este conjunto de características possui um desempenho inferior que o conjunto de características apresentado anteriormente. A próxima seção mostrará os resultados no desempenho do sistema extraindo estas características após alguns pré-processamentos.

5.5.1 Avaliação dos Pré-processamentos

Nesta seção são demonstradas as aplicações de algoritmos de pré-processamento para correções de imperfeições na escrita das palavras utilizadas no reconhecimento. Como foi feito para o conjunto de características CT, os mesmo pré-

Tabela 5.21: Taxas de reconhecimento para o primeiro estágio do sistema com 256 símbolos e sem pré-processamentos.

Léx. (#)	**Top 1**	**Top 5**	**Top 10**
10	94,22	97,90	100,00
100	87,33	93,05	94,15
1000	74,22	85,99	88,98
3717	53,62	72,26	77,08

processamentos serão avaliados para os conjunto CG.

Assim como no outro conjunto de características, apenas a correção da inclinação vertical mostrou melhoras no reconhecimento. O aumento na taxa de reconhecimento com esta correção foi de 10,62%, fazendo com que a taxa de reconhecimento para todo o léxico fique em 64,24%. A Tabela 5.22 traz em detalhes os resultados para estes experimentos.

Tabela 5.22: Taxas de reconhecimento para o primeiro estágio do sistema com correção de inclinação vertical.

Léx. (#)	**Top 1**	**Top 5**	**Top 10**
10	95.33	99,07	100,00
100	86,49	94,33	96,11
1000	71,48	82,96	85,99
3717	64,24	79,57	83,96

A correção da inclinação da linha de base e a normalização do corpo da palavra foram pré-processamentos que não trouxeram aumentos nas taxas de reconhecimento. Com a correção da linha de base, o sistema teve uma queda 0,14%, ficando com a taxa de reconhecimento em 64,10%. Já para a normalização do corpo da palavra, a queda foi de 4,42%, com a taxa de reconhecimento ficando em 59,82%. Os resultados completos destes experimentos estão expostos na Tabela 5.23.

Tabela 5.23: Taxas de reconhecimento para o primeiro estágio do sistema com correção da linha de base e normalização do corpo da palavra, separadamente.

	Linha de base			**Norm. corpo**		
Léx. (#)	**Top 1**	**Top 5**	**Top 10**	**Top 1**	**Top 5**	**Top 10**
10	94,22	98,11	100,00	93,91	97,92	100,00
100	87,13	93,40	94,76	86,95	92,54	94,71
1000	73,73	85,70	88,98	71,19	85,49	88,91
3717	64,10	80,18	83,74	59,82	78,00	83,03

5.5.2 Otimização do Codebook

Em seqüência aos experimentos anteriores, a otimização do *codebook* torna-se necessária para este conjunto de características. Como as informações extraídas, assim como a quantidade destas, é diferente do conjunto de características reportado anteriormente, então há possibilidades do valor ótimo de símbolos tenha valores distintos no dois conjuntos.

Como foi demonstrado que a seleção supervisionada possui melhores resultados que os métodos propostos para a seleção não-supervisionada, então apenas a primeira abordagem foi considerada para este conjunto de características. A Figura 5.6 traz um gráfico dos resultados apresentados nestes experimentos. Pode ser observado que o melhor resultado foi apresentado pelo *codebook* com 192 símbolos, com uma taxa de reconhecimento de 65,03%. Com este número de símbolos houve um aumento de 0,79% na taxa de reconhecimento, comparando com os resultados apresentados com 256 símbolos. Para os léxicos de 10, 100 e 1.000 palavras, as taxas de reconhecimento foram de 97,07%, 93,33% e 84,24%, respectivamente.

5.5.3 Otimização do Treinamento

Outro fator de otimização avaliado para este conjunto de características foi a inserção de novos exemplos para o treinamento. Assim como para a otimização do

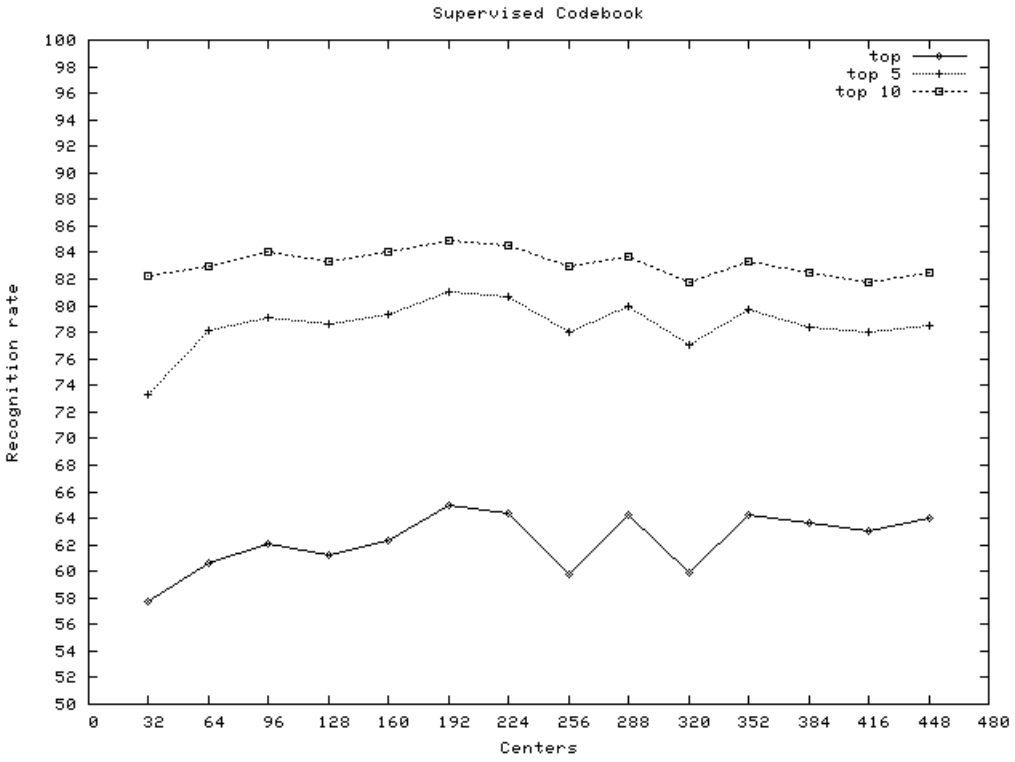


Figura 5.6: Avaliação supervisionada do codebook do primeiro estágio, considerando o conjunto CG.

codebook, foi considerado apenas o método que apresentou melhores resultados com o outro conjunto de características.

Foram inseridos exemplos da base IRONOFF para o treinamento das classes com menor número de exemplos, e os resultados estão demonstrados na Tabela 5.24. Assim como no conjunto de características CT, este método trouxe aumentos nas taxas de reconhecimento para este conjunto de características também. Considerando todo o léxico, o aumento foi de 1,17%, fazendo com que a taxa de reconhecimeto fique em 66,20%.

Tabela 5.24: Taxas de reconhecimento para o primeiro estágio do sistema, com as características CG, e utilizando exemplos da base IRONOFF para treinar as classes com menor número de exemplos.

Léx. (#)	**Top 1**	**Top 5**	**Top 10**
10	95,15	98,64	100,00
100	88,77	94,76	95,90
1000	76,29	87,91	90,87
3717	66,20	82,00	85,99

5.5.4 Avaliação da Verificação

Nesta seção são apresentados os experimentos referentes à avaliação da verificação utilizando as hipóteses geradas pelo conjunto de características CG. Com isso torna-se possível avaliar tanto a segmentação do primeiro estágio com estas características, assim como o impacto da verificação nesta implementação do primeiro estágio.

São consideradas aqui a implementação da verificação que alcança os melhores resultados, conforme é apresentado na Seção 5.4. Além disso, é também avaliada a aplicação do limiar de decisão com as probabilidades geradas por este conjunto de características. Como os resultados podem trazer resultados diferentes dos apresentados pelo outro conjunto de características, os valores do limiar são reavaliados.

Tabela 5.25: Taxas de reconhecimento para o estágio de verificação utilizando as hipóteses geradas com o conjunto de características CG.

Léxico	**3.717**		**1.000**		**100**		**10**	
Informação	**Top 1**	**Top 5**	**Top 1**	**Top 5**	**Top 1**	**Top 5**	**Top 1**	**Top 5**
Coluna + Linha	61,75	82,85	69,05	87,98	83,24	94,15	92,83	98,43
Coluna	65,28	82,74	75,19	87,91	86,02	93,69	93,15	98,32
Linha	45,60	81,00	51,73	86,20	67,52	92,51	77,40	97,14
Col + Lin + 1º Est.	70,16	83,49	78,79	88,56	89,55	94,65	95,90	98,68
Coluna + 1º Est.	71,51	83,71	80,68	88,63	90,34	94,37	95,12	98,86
Linha + 1º Est.	57,40	82,32	64,74	87,34	82,64	93,65	92,87	98,11
Média	61,95	82,68	70,03	87,77	82,22	93,84	91,21	98,26
Desvio Padrão	9,57	0,97	10,78	0,90	8,31	0,76	6,89	0,61

O limiar de decisão também foi avaliado para selecionar as palavras a serem verificados. Os valores deste foram re-avaliados, já que as probabilidades podem estar em intervalos diferentes do apresentado pelo outro conjunto de características. A Tabela 5.26 traz os resultados apresentados e na Tabela 5.27 encontram-se as taxas de palavras que foram passadas para o estágio de verificação. Foi observado que nestes experimentos o limiar não trouxe aumentos nas taxas de reconhecimento na maioria dos casos. Apenas para os léxicos menores (dez e cem palavras) foram observados aumentos nas taxas de reconhecimento.

Tabela 5.26: Taxas de reconhecimento para o estágio de verificação utilizando o limiar de decisão e o primeiro estágio com CG.

Léx. \ Limiar	**0,001**	**0,002**	**0,003**	**0,004**	**0,005**
10	95,65	96,01	95,86	95,65	95,68
100	89,95	90,45	90,59	90,55	90,09
1.000	79,57	80,32	80,61	80,28	79,64
3.717	70,52	71,41	71,48	71,09	70,59

Tabela 5.27: Porcentagem de palavras verificadas com a utilização do limiar de decisão e o primeiro estágio com CG.

Léx. \ Limiar	**0,001**	**0,002**	**0,003**	**0,004**	**0,005**
10	1,85	3,6	5,13	7,23	8,84
100	6,38	11,26	15,29	19,29	24,74
1000	16,18	26,95	35,94	44,14	52,16
3.717	23,78	40,50	53,01	63,24	72,55

Considerando que os melhores resultados alcançados pela verificação combinam as probabilidades do primeiro estágio e das colunas dos segmentos, a Tabela 5.28 traz um resumo das taxas de reconhecimento, com e sem a verificação. Com exceção do sistema com léxico de dez palavras, todos os sistemas apresentaram aumento na taxa de reconhecimento utilizando a verificação. O aumento médio na taxa de reconhecimento foi de 2,81%, sendo que o maior ganho de desempenho ficou com o sistema utilizando todo

o léxico. Isto mostra mais uma vez que a influência da verificação aumenta conforme o tamanho do léxico.

Tabela 5.28: Taxas de reconhecimento alcançadas apenas pelo primeiro estágio (com características CG), e com a inclusão do segundo estágio.

	Primeiro Estágio		**Segundo Estágio**	
Léx. (#)	**Top 1**	**Top 5**	**Top 1**	**Top 5**
10	95,15	98,64	**95,12**	98,82
100	88,77	94,96	**90,34**	94,37
1000	76,29	87,91	**80,68**	88,63
3717	66,20	82,00	**71,51**	83,71

5.6 COMBINAÇÃO DE CONJUNTOS DE CARACTERÍSTICAS

Foram propostos dois conjuntos de características para o primeiro estágio do sistema, e o desempenho destes foi avaliado individualmente. Nesta seção é mostrado o desempenho destes conjuntos de características aplicados conjuntamente, para que consiga-se assim melhoras nas taxas de reconhecimento do sistema.

A avaliação da combinação de conjuntos de características dá-se de duas maneiras. Na primeira, métodos clássicos da literatura são aplicados para a combinação dos resultados final dos sistemas implementados com conjuntos de características diferentes. Na segunda maneira, métodos são propostos para que a combinação seja efetuada antes do estágio de verificação, buscando assim melhoras no desempenho da segmentação.

5.6.1 Avaliação da Combinação Utilizando Métodos da Literatura

A utilização dos métodos clássicos de combinação avaliou as três categorias de problemas de combinação. Isto torna-se possível já que tem-se tanto a informação de *ranking* de cada sistema, como a probabilidade para cada hipótese. Primeiramente foi avaliado a combinação apenas das palavras classificadas como primeira opção dos classificadores.

O método de votação fui avaliado para isto, onde duas alternativas foram avaliadas para a decisão em caso de empate. Escolhendo a palavra com maior probabilidade, as taxas de reconhecimento com esta combinação foram de 96,15%, 91,87%, 83,46%, e 73,87%, para os léxicos com 10, 100, 1.000 e 3.717 palavras, respectivamente. Já escolhendo a palavra com menor probabilidade, as taxas de reconhecimento ficaram em 96,18%, 92,51%, 83,28%, e 75,65%, seguindo a mesma seqüência de léxicos do experimento anterior.

Posteriormente à avaliação com o método de votação, o método *Borda Count* foi aplicado para realizar a combinação das saídas dos sistemas. Novamente as probabilidades foram utilizadas para a decisão em caso de empates, e além da avaliação da utilização das probabilidades mínimas e máximas e a média as probabilidades também foram avaliadas. A Tabela 5.29 mostra as taxas de reconhecimento destes experimentos, sendo que os melhores resultados foram alcançados com a utilização da média das probabilidades como critério de desempate.

Tabela 5.29: Taxas de reconhecimento para a combinação das duas implementações do sistema utilizando a combinação Borda Count.

Desempate	**Máximo**		**Mínimo**		**Média**	
Léx. (#)	**Top 1**	**Top 5**	**Top 1**	**Top 5**	**Top 1**	**Top 5**
10	95,54	99,50	95,86	99,50	95,97	99,50
100	90,91	96,61	90,84	96,61	91,16	96,61
1000	82,25	93,87	82,39	93,87	82,81	93,87
3717	73,51	89,48	74,33	89,45	74,69	89,48

O último método de combinação avaliado foi através da combinação das probabilidades. Neste método foi formada uma nova lista a partir das ocorrências nos dois *rankings*, e três critérios foram avaliados para o cálculo da nova probabilidade. Estes critérios foram a probabilidade máxima, a probabilidade mínimas, e a média das probabilidades. Os resultados constam na Tabela 5.30, onde as melhores taxas de reconhecimento foram alcançadas ao considerar a probabilidade máxima para cada palavra.

Tabela 5.30: Taxas de reconhecimento para a combinação das duas implementações do sistema utilizando a combinação das probabilidades.

Desempate	**Máximo**		**Mínimo**		**Média**	
Léx. (#)	**Top 1**	**Top 5**	**Top 1**	**Top 5**	**Top 1**	**Top 5**
10	96,15	99,46	96,33	99,14	97,04	99,43
100	91,87	96,29	72,98	95,97	88,27	96,22
1000	83,46	91,91	61,14	91,12	78,39	92,12
3717	73,87	88,13	51,59	84,92	66,95	87,41

5.6.2 Avaliação da Combinação do Primeiro Estágio Utilizando MEMs Distintos

Na Seção 4.4.2 foi proposto um método para a combinação dos conjuntos de características do primeiro estágio. Os resultados apresentados por este método contribuíram em um aumento de 0,53% na taxa de reconhecimento do primeiro estágio, considerando todo o léxico. Entretanto, esta abordagem representou um aumento médio 3,61%, considerando todos os tamanhos de léxicos. O maior aumento na taxa de reconhecimento foi para o léxico de mil palavras, onde o aumento foi de 9,52%. A Tabela 5.31 traz os detalhes destes experimentos.

Tabela 5.31: Taxas de reconhecimento para a combinação das duas implementações do primeiro estágio do sistema.

Léx. (#)	**Top 1**	**Top 5**	**Top 10**
10	98,11	99,25	100,00
100	95,97	97,68	98,18
1000	93,12	95,69	96,29
3717	75,29	87,52	90,05

Os resultados apresentados pelo primeiro estágio esta combinação foram os melhores já apresentados até aqui por esta fase do sistema. A Tabela 5.32 traz os resultados da verificação, desta vez combinando os pontos de segmentação de ambos conjuntos de características. Essa combinação é dada pela soma da probabilidade final de cada hipótese, sendo que agora cada hipótese possui dois conjuntos de pontos de segmentação. O único

tamanho de léxico que apresentou aumento na taxa de reconhecimento foi o léxico de 3.717 palavras. Este aumento foi de 0,82%.

Tabela 5.32: Taxas de reconhecimento para o estágio de verificação utilizando as hipóteses geradas com a combinação dos conjuntos de características.

Léxico	**3.717**		**1.000**		**100**		**10**	
Informação	**Top 1**	**Top 5**	**Top 1**	**Top 5**	**Top 1**	**Top 5**	**Top 1**	**Top 5**
Coluna + Linha	64,42	86,06	80,36	94,58	87,56	96,93	93.80	98,97
Coluna	67,09	85,99	84,21	94,05	89,80	96,79	94,29	99,07
Linha	48,66	84,03	66,06	93,33	75,37	95,39	82,25	97,75
Col + Lin + 1º Est.	74,01	87,41	90,77	95,19	94,15	97,36	97,33	99,29
Coluna + 1º Est.	76,11	87,20	92,12	94,62	94,33	97,33	96,72	99,39
Linha + 1º Est.	63,88	86,02	85,81	94,37	90,98	96,58	95,11	98,72
Média	65,69	86,11	83,22	94,36	88,70	96,66	93,25	98,87
Desvio Padrão	9,74	1,20	9,45	0,63	7,03	0,72	5,56	0,60

Em busca melhorar os resultados da verificação, também foi avaliada nesta implementação a utilização do limiar de decisão, proposto na Seção 4.3.2.1. A Tabela 5.33 traz os resultados da avaliação dos valores entre 0,001 e 0,005 para o limiar, enquanto a Tabela 5.34 mostra a porcentagem de palavras selecionadas para serem verificadas. O melhor resultado apresentado para o léxico total foi de 77,68%, com o limiar de 0,003, e esta taxa de reconhecimento é 0,32% menor que o resultado apresentado pelo sistema com o conjunto de características CG. Entretanto, se considerarmos todos os tamanhos de léxico, esta implementação atingiu taxas de reconhecimento, em média, 2,6% maiores que a outra implementação. O destaque fica para o léxico de mil palavras, com uma taxa de reconhecimento 7,39% maior.

Um resumo dos experimentos apresentados por esta combinação podem ser vistos na Tabela 5.35. Neste caso a verificação não trouxe um impacto muito grande ao sistema, já que o primeiro estágio atingiu altas taxas de reconhecimento. O aumento médio utilizando a verificação foi de 0,91%, mas este aumento tende a ser maior com léxicos maiores.

Tabela 5.33: Taxas de reconhecimento para o estágio de verificação utilizando o limiar de decisão e os pontos de segmentação gerados pelos dois conjuntos de características.

Léx. \ Limiar	**0,001**	**0,002**	**0,003**	**0,004**	**0,005**
10	98,32	98,22	98,08	98,18	98,18
100	96,08	96,29	96,15	95,90	95,79
1.000	93,40	93,83	93,65	93,72	93,51
3.717	77,25	77,50	77,68	77,36	77,25

Tabela 5.34: Porcentagem de palavras verificadas com a utilização do limiar de decisão e os pontos de segmentação gerados pelos dois conjuntos de características.

Léx. \ Limiar	**0,001**	**0,002**	**0,003**	**0,004**	**0,005**
10	0,82	1,89	2,67	3,28	3,99
100	2,46	3,60	5,03	6,17	7,27
1000	4,10	7,09	8,77	10,69	12,29
3.717	15,90	27,56	37,68	46,63	54,87

Tabela 5.35: Taxas de reconhecimento alcançadas apenas pelo primeiro estágio (combinando os dois conjuntos de características através da soma das saídas dos sistemas), e com a inclusão do segundo estágio.

	Primeiro Estágio		**Segundo Estágio**	
Léx. (#)	**Top 1**	**Top 5**	**Top 1**	**Top 5**
10	98,11	99,25	**98,32**	99,32
100	95,97	97,68	**96,29**	97,50
1000	93,12	95,69	**93,83**	95,72
3717	75,29	87,52	**77,68**	83,71

5.6.3 Avaliação da Combinação no Primeiro Estágio Utilizando MEMs Compartilhados

Para finalizar as avaliações das combinações dos conjuntos de características, a avaliação da combinação proposta na Seção 5.6.2 foi avaliada. A Tabela 5.36 traz os resultados apresentados pelo primeiro estágio desta implementação. Pode ser observado que houveram quedas na taxas de reconhecimento.

Tabela 5.36: Taxas de reconhecimento para a combinação das seqüências de observações no primeiro estágio do sistema.

Léx. (#)	**Top 1**	**Top 5**	**Top 10**
10	89,63	94,54	100,00
100	84,70	88,98	90,66
1000	76,43	84,42	85,74
3717	68,56	81,14	83,53

Além das quedas nas taxas de reconhecimento do primeiro estágio, esta abordagem não demonstrou bons resultados para a segmentação. A avaliação desta, através da verificação, demonstrou que todos os léxicos tiveram redução nas taxas de reconhecimento. A Tabela 5.37 traz os resultados da verificação.

Tabela 5.37: Taxas de reconhecimento para o estágio de verificação utilizando as hipóteses geradas com a combinação dos conjuntos de características treinados no mesmo MEM.

Léxico	**3.717**		**1.000**		**100**		**10**	
Informação	**Top 1**	**Top 5**	**Top 1**	**Top 5**	**Top 1**	**Top 5**	**Top 1**	**Top 5**
Coluna + Linha	59,64	79,79	66,45	82,74	77,40	88,56	89,23	97,33
Coluna	62,42	79,71	71,34	82,82	81,75	88,59	90,37	96,79
Linha	45,17	78,15	50,55	81,25	57,93	85,81	72,23	94,97
Col + Lin + 1º Est.	60,10	80,04	67,27	82,78	78,61	88,66	89,02	96,72
Coluna + 1º Est.	62,78	79,82	71,66	82,85	82,17	88,63	89,69	96,79
Linha + 1º Est.	45,81	78,22	51,55	81,39	60,14	85,95	78,97	95,29
Média	55,99	79,29	66,13	82,30	73,00	87,70	84,91	96,31
Desvio Padrão	8,22	0,86	9,60	0,76	10,99	1,41	7,54	0,93

5.7 ANÁLISE DE ERROS

A Tabela 5.38 traz alguns exemplos de palavras que não foram reconhecidas corretamente pelo sistema, considerando os experimentos apresentados na Seção 5.6.2. Para o léxico de 3.717 palavras, a taxa de erro do sistema foi de 22,32%. Ao observar estes exemplos, pode ser observado que alguns dos erros apresentados são devido a ruídos na

base de dados. Estes ruídos são causados por erros do escritor, que dificultam o reconhecimento do sistema. Outro fator que trouxe confusão ao reconhecimento é a variabilidade na forma do caracter, sendo que alguns casos, este pode ser confundido com outro caracter, ou até mesmo com seqüências de caracteres.

A espessura da caneta utilizada pode fazer com que alguns componentes importantes do caracteres, como concavidades, não estejam presentes em alguns casos. Isto traz confusão tanto para o primeiro estágio, quanto para o segundo. Com relação à classificação de caracteres maiúsculos e minúsculos, um fator que pode trazer confusão é não ter a presença de caracteres com tamanhos diferentes na mesma palavra, ou seja, caracteres que demonstrem claramente a presença de um caracter maiúsculo.

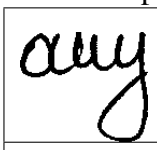
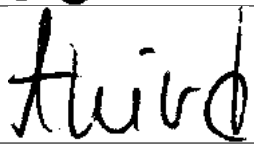
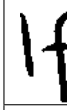
Dois outros fatores observados também trouxeram confusão ao sistema. Ambos estão relacionados à posição dos componentes dos caracteres. No primeiro, o deslocamento de componentes, que por natureza são desconectados do caracter, trouxe falhas para o reconhecimento. No segundo, a confusão foi causada pela desconectividade de componentes que por natureza deveriam estar conectados. Este tipo de caso faz com que um caracter seja confundido com seqüências de caracteres.

Para finalizar esta seção, a Tabela 5.39 traz exemplos de palavras classificadas corretamente pelo sistema. Nesta podem ser observados os casos em que o sistema obteve sucesso no reconhecimento, apesar dos fatores de complexidades apresentados por estas palavras, como inclinações, descontinuidades na formas dos caracteres, variabilidades nas formas de escritas, e caracteres conectados.

5.8 DISCUSSÃO

Este capítulo abordou todos os processos realizados para otimizar o desempenho do sistema proposto, partindo da implementação do primeiro estágio até a combinação de sistemas. Em todos os experimentos quatro tamanhos diferentes de léxicos foram ava-

Tabela 5.38: Exemplos de palavras não reconhecidas corretamente. O valor correto está entre parênteses.

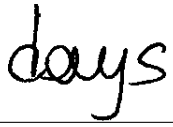
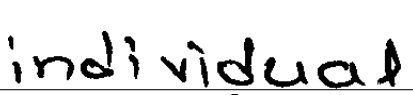
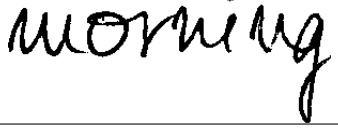
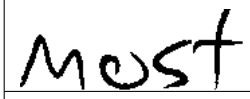
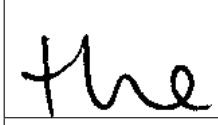
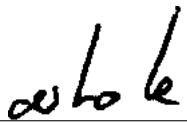
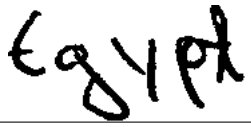
ally (any) 	ben (been) 
grasp (group) 	in (him) 
port (regard) 	So (so) 
fleet (that) 	feeling (third) 
do (to) 	walls (wake) 
if (If) 	it (It) 
last (laugh) 	let (Let) 

liados, para demonstrar o desempenho do método em diferentes grandezas de problema.

Primeiramente foi avaliado o conjunto de características CT como ferramenta de reconhecimento e segmentação. A Figuras 5.7 representa o gráfico das taxas de reconhecimento do método com este conjunto de características, demonstrando a evolução do desempenho do sistema para os léxicos de 10, 100, 1.000 e 3.717 palavras. Com isto torna-se possível a visualização dos pontos onde houveram progressos no desempenho.

Neste gráficos os primeiros experimentos apresentados foram realizados para avaliar a primeira etapa do método (experimentos 1 a 9). Partindo da avaliação com

Tabela 5.39: Exemplos de palavras reconhecidas corretamente.

and 	days 
from 	individual 
modern 	morning 
most 	second 
the 	whole 
Gundpowder 	Egypt 

parâmetros apresentados em trabalhos relacionados, diversas estratégias foram propostas para a otimização. Esta busca não apenas aumentos na taxa de reconhecimento, mas também melhores resultados para a segmentação das palavras. Do primeiro experimento realizado, até o melhor resultado alcançado, foi atingido um aumento de 13,69% na taxa de reconhecimento deste estágio. É possível ser observado nos experimentos 2, 5 e 9 a evolução devido à inclusão da correção de inclinação vertical, à otimização do número de símbolos no *codebook*, e à inclusão de caracteres isolados da base IRONOFF nas classes com menor número de exemplos.

Após a conclusão dos experimentos com o primeiro estágio do sistema, este foi utilizado para segmentar a base de palavras utilizado. Com isso, foi criada uma base de

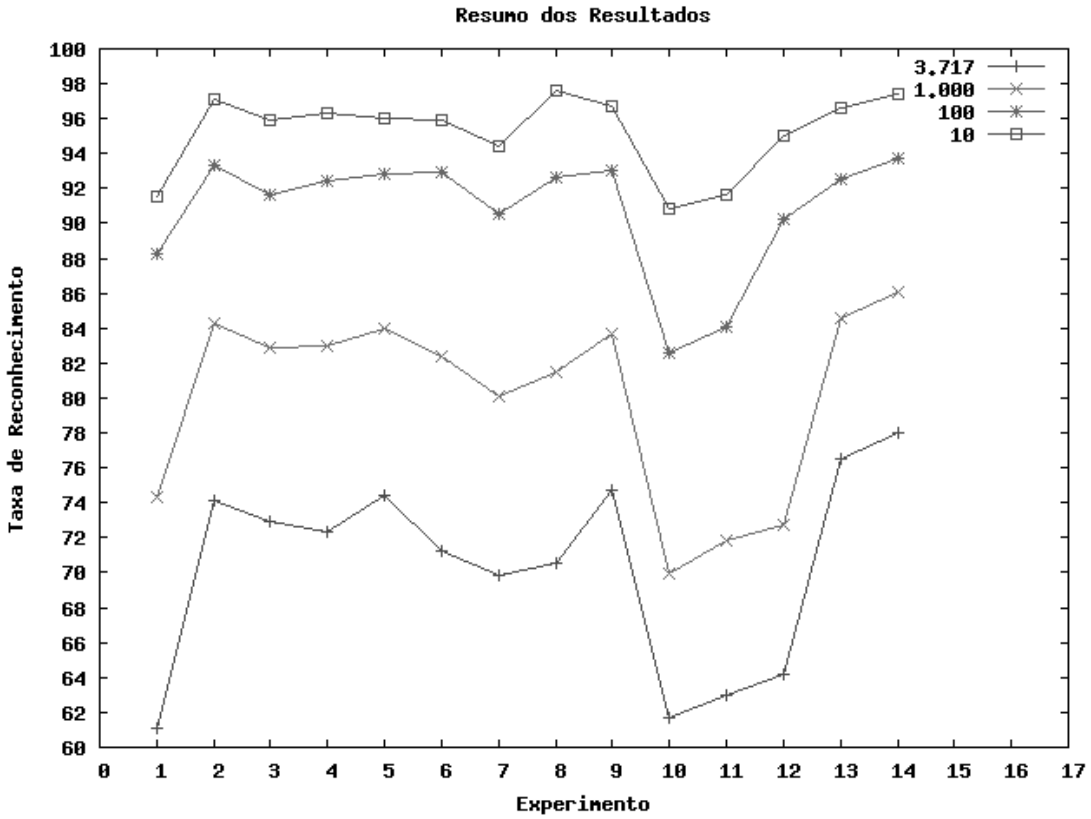


Figura 5.7: Resumo das taxas de reconhecimento utilizando o conjunto de características CT no primeiro estágio. Os experimentos 1 a 9 estão relacionado ao primeiro estágio, sendo que: 1, sem pré-processamentos; 2, correção de inclinação vertical; 3, correção de inclinação vertical e linha de base; 4, correção de inclinação vertical e normalização do corpo da palavra; 5, com 288 símbolos; 6, estados computados na base IRONOFF; 7, estados computados na base IAM; 8, inclusão de dados em todas as classes; 9, inclusão de dados nas classes com poucos exemplos. Os experimentos 10 a 14 corresponde à verificação, sendo que: 10, resultados apenas do segundo estágio; 11, com 288 símbolos; 12, inclusão de dados nas classes com poucos exemplos; 13, combinação com o primeiro estágio; 14, limiar de decisão.

caracteres isolados, que é utilizada no treinamento do segundo estágio do sistema. Além da segmentação automática, foi necessária a avaliação manual dos caracteres segmentados. Nesta avaliação, 18,73% dos caracteres foram desconsiderados para diminuir o ruído de

caracteres mal segmentados.

Com a base de caracteres criada, foi então avaliado o segundo estágio do sistema, que é utilizado para verificar as hipóteses computadas pelo primeiro. Assim como o primeiro estágio, este também foi primeiro avaliado com os parâmetros propostos em trabalhos relacionados, e foi otimizado até alcançar as maiores taxas de reconhecimento (experimentos 10 a 14 na Figura 5.7). Nesta etapa, foi proposto e avaliado um limiar para selecionar as palavras a serem verificadas, o qual trouxe aumentos nas taxas de reconhecimento (experimento 14 na Figura 5.7). O estágio de verificação trouxe aumentos nas taxas de reconhecimento, sendo que para todo o léxico, este aumento foi de 3,24%.

Dados os resultados apresentados pelo sistema, outro conjunto de características, chamado de CG, foi avaliado para a segmentação e a classificação das palavras no primeiro estágio. As taxas de reconhecimento, para todo o léxico, foram de 66,20% para o primeiro estágio, e 71,51% com o segundo estágio. Este conjunto de características não demonstrou resultados tão bons quanto o outro conjunto CT, sendo que este alcançou taxas de reconhecimento 6,49% maiores, considerando a taxa final do sistema e todo o léxico.

Apesar da diferença apresentada pelos dois conjuntos avaliados para o primeiro estágio, a combinação destes conjuntos de características demonstrou que dois conjuntos diferentes podem complementar a discriminação entre as classes, e aumentar o resultado do sistema. No experimento da Seção 5.6.2, em que são combinadas as saídas de dois sistemas distintos, a taxa de reconhecimento final do sistema para todo o léxico teve uma pequena queda de 0,38%. Porém, considerando todos os léxicos, esta implementação demonstrou um aumento significativo de 2,6% em média, demonstrando que a combinação de conjuntos de características é um caminho promissor. Além disso, os experimentos com o método *Borda Count* demonstraram melhores resultados que a combinação de probabilidades, e pode trazer melhores resultados se aplicado no método proposto na Seção 4.4.2. A Figura 5.8 traz um resumo das taxas de reconhecimento apresentadas pelos métodos

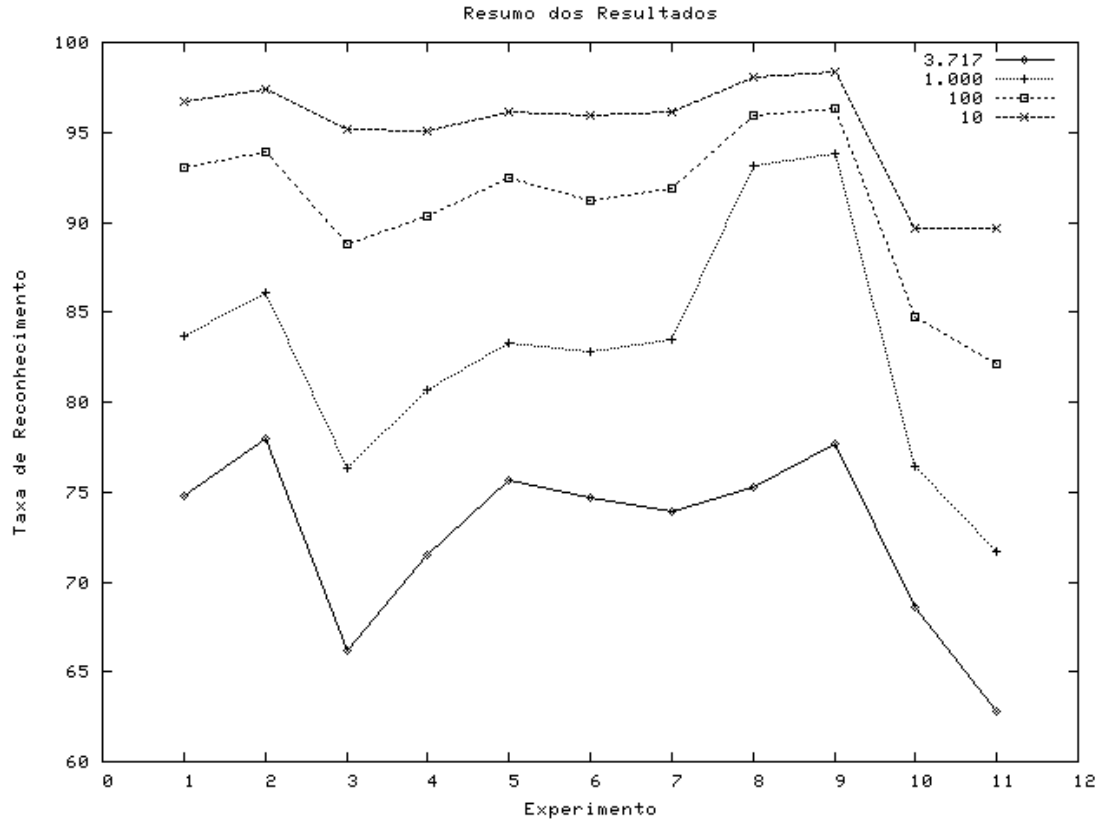


Figura 5.8: Resumo das taxas de reconhecimento de todas as implementações propostas. 1 e 2, primeiro e segundo estágio com CT. 3 e 4, primeiro e segundo estágio com CG. 5, 6 e 7, votação, BC e combinação de probabilidades. 8 e 9, combinação de CT e CG no primeiro estágio com MEMs distintos e verificação. 10 e 11, combinação de CT e CG no primeiro estágio com MEMs compartilhados e verificação.

propostos neste trabalho.

Em comparação ao trabalho de Günter [GÜNTER, 2004b], que utiliza o conjunto de características CG como um de seus conjuntos de características, o método aqui proposto, em sua melhor implementação, atinge uma taxa de reconhecimento cerca de 7% menor que o método de Günter. O método aqui proposto ainda necessita de diversas otimizações, as quais podem trazer maiores taxas de reconhecimento. Além disso, este método considera apenas informações das palavras isoladamente, tanto nos pré-

processamentos como na extração de características, enquanto o método de Günter considera informações de toda uma linha de texto (um conjunto de palavras), o que pode trazer um diferença favorável ao último método.

6 CONCLUSÕES

Este capítulo traz as conclusões e os trabalhos futuros elaborados a partir da análise dos experimentos realizados. Devido à complexidade do método proposto, podemos dividi-lo em três fases, onde faz-se basicamente alterações no primeiro estágio do sistema. A primeira fase avaliou-se o conjunto de características originalmente aplicado para o reconhecimento de cadeias numéricas. Na segunda é avaliado um conjunto de características proposto para o reconhecimento de palavras, e que tem demonstrado bons resultados em trabalhos relacionados. E por último, a combinação dos dois conjuntos de características para otimizar a etapa de segmentação implícita também foi avaliada.

Os experimentos foram realizados em tamanhos diferentes de léxico, para que seja possível avaliar o método em diferentes grandezas de problemas. O comportamento das três implementações do sistema foram proporcionais aos tamanhos dos léxicos, ou seja, os melhores resultados em cada léxico foram apresentados pela mesma implementação. As diferenças entre as implementações do método, com relação ao tamanho do léxico, fica apenas no ajuste do limiar de decisão, sendo este varia conforme o tamanho do léxico, com a confusão tendendo a ser menor com léxicos menores.

Em termos de taxa de reconhecimento, as melhores implementações foram com o conjunto de características CT e para a combinação dos conjuntos. A taxa de reconhecimento final destes sistemas, para o léxico de 3.717 palavras, foi de 78% e 77,68%, respectivamente. Houve apenas uma diferença de 0,32% em favor do sistema com apenas um conjunto de características. Porém, a combinação dos conjuntos de características trouxe um aumento médio de 2,6% ao se considerar todos os léxicos, o que demonstra um aumento de desempenho. Já o sistema com o conjunto de características CG apresentou 71,48%, mas futuras otimizações no sitema podem melhorar o desempenho deste, já que esta taxa de reconhecimento está abaixo do apresentado por outros métodos na literatura que utilizam este conjunto de características.

Um ponto observado é que o conjunto de primitivas CT, comparado com o conjunto CG, atinge uma taxa de reconhecimento 8,56% maior, se considerarmos apenas o primeiro estágio. Tendo em vista que outros métodos que utilizam o conjunto CG apresentam maiores taxas de reconhecimento, novas otimizações no sistema podem fazer com que a implementação utilizando o conjunto CT, ou até mesmo a combinação dos dois conjuntos, tornem-se método promissores para atingirem melhores resultados que os apresentados na literatura.

Diversos pontos ainda necessitam ser avaliados em trabalhos futuros, principalmente no primeiro estágio, com o objetivo de melhorar o seu desempenho de segmentação. Novos métodos de pré-processamentos devem ser avaliados, como por exemplos, a normalização das regiões das palavras utilizando tamanho fixo. A variabilidade entre os tamanhos dos caracteres é um fator que que pode ser diminuídos através de métodos de pré-processamento. Os números de estados dos MEMs também devem ser re-avaliados, com a possibilidade de melhores resultados. Nos experimentos realizados, foi observado que a taxa de reconhecimento foi proporcional ao número médio de estados. Ou seja, as maiores taxas de reconhecimento foram observadas no conjunto com maior média de estados.

O aumento dos dados de treinamento, nas classes com poucos exemplos, foi uma estratégia que apresentou bons resultados em todos os experimentos em que foi avaliado. Um trabalho futuro pode avaliar a utilização de outras bases de dados para esta tarefa, para avaliar a inclusão de um maior número de exemplos nas classes. Um maior número de exemplos pode melhorar a re-estimação dos parâmetros dos MEM.

Os resultados de reconhecimento apenas utilizando características extraídas das linhas dos caracteres, no segundo estágio, demonstraram que a segmentação do primeiro estágio necessita de otimizações. As baixas taxas de reconhecimento com as observações das linhas demonstram que os caracteres possuem excesso ou falta de fragmentos, devido à má segmentação. Isto ficou mais claro com a inclusão de exemplos da

base IRONOFF, que possui caracteres originalmente isolados. Com a inclusão destes, a taxa de reconhecimento para as observações diminuiu, demonstrando que há diferenças entre os caracteres das palavras testadas e os caracteres isolados da base IRONOFF.

A combinação de características que apresentou os melhores resultados foi proposta neste trabalho. Com base na avaliação dos resultados apresentados por métodos de combinação já existentes, podem ser avaliadas diferentes abordagens de combinação dos classificadores tanto no primeiro estágio, como no segundo. O método *Borda Count* apresentou melhores resultados que a combinação das probabilidades, e pode trazer aumentos nas taxas de reconhecimento do método de combinação proposto.

Além dos trabalhos futuros já propostos, a avaliação de outros conjuntos de características pode ser útil. Tanto no primeiro estágio quanto no segundo, a avaliação de outros conjuntos de características podem trazer um melhor desempenho na segmentação do segundo estágio, assim como um maior poder de verificação para o segundo estágio.

REFERÊNCIAS

[ARICA & YARMAN-VURAL, 2002] ARICA, N., YARMAN-VURAL, F. T.. *Optical Character Recognition for Cursive Handwriting*, IEEE Transactions on Pattern Analysis and Machine Inteligence, vol. 24, n. 6, p. 801-813, jun. 2002.

[BOZINOVIC & SRIHARI, 1989] BOZINOVIC, R.M., SRIHARI, S.. *Off-line Cursive Script Word Recognition*. IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 11, n. 1, p. 68-83, jan. 1989.

[BRITTO, 2000] BRITTO JR., A. S. et al, *Improvement in Handwritten Numeral String Recognition by Slant Normalization and Contextual Information*, Seventh International Workshop on Frontiers in Handwriting Recognition (IWFHR-7), Amsterdam, p. 323-332, set. 2000.

[BRITTO, 2001] BRITTO JR., A. S.. *A Two-Stage HMM-Based Method For Recognizing Handwritten Numeral Strings*. Curitiba, 142 p., Tese de Doutorado - Departamento de Informática, Pontifícia Universidade Católica do Paraná, 2001.

[BRITTO et al., 2003] BRITTO JR., A. S. et al, *Complementary Features Combined in an HMM-based System to Recognize Handwritten Digits*. Proceedings of the International Conference on Image Analysis and Processing - ICIAP03, Mantova, p. 676-680, 17 a 19 de set. de 2003.

[BRITTO et al., 2004] BRITTO JR., A. S. et al, *Foreground and Background Information in an HMM-Based Method for Recognition of Isolated Characters and Numeral Strings*. 9th International Workshop on Frontiers in Handwriting Recognition (IWFHR-9), Tokio, Japão, p. 371-376, 26 a 29 de out. de 2004.

[BUNKE et al., 1995] BUNKE, H., ROTH, M., SCHUKAT-TALAMAZZINI, E. G., *Off-Line Cursive Handwriting Recognition Using Hidden Markov Models*, Pattern Recognition, vol. 28, n. 9, p. 1399-1413, 1995.

[BUNKE & ZIMMERMANN, 1998] BUNKE, H., HA, T. M., ZIMMERMANN, M., *Off-Line Handwritten Numeral String Recognition by Combining Segmentation-based and Segmentation-free Methods*, Pattern Recognition, vol. 31, n. 3, p. 257-272, 1998.

[BUNKE & MARTI, 1999] BUNKE, H., MARTI, U., *A Full English Sentence Database For Off-Line Handwriting Recognition*. Proc. of the 5th Int. Conf. on Document Analysis and Recognition, ICDAR 99. Bangalore, p. 705-708, 1999.

[BUNKE & MARTI, 2000] BUNKE, H., MARTI, U., *Handwritten Sentence Recognition*. Proc. of 15º ICPR, Barcelona, Espanha, vol. 3, p. 467-470, 2000.

[BUNKE & MARTI, 2001] BUNKE, H., MARTI, U., *Text Line Segmentation and Word Recognition in a System for General Writer Independent Handwriting Recognition*. Proc. of 6^o ICDAR, p. 159-163, 2001.

[BUNKE & ZIMMERMANN, 2002a] BUNKE H., ZIMMERMANN M., *Hidden Markov Model Length Optimization for Handwriting Recognition Systems*, International Workshop on Frontiers in Handwriting Recognition (IWFHR), Niagra-on-the-Lakes, p. 369-374, 2002.

[BUNKE & ZIMMERMANN, 2002b] BUNKE H., ZIMMERMANN M., *Automatic Segmentation of the IAM Off-line Database for Handwritten English Text*, IEEE, 2002.

[CAI & LIU, 1998] CAI, J., LIU, Z.. *Integration of Structural and Statistical Information for Unconstrained Handwritten Numeral Recognition*. Proceedings of the 14th Internation Conference on Pattern Recognition, vol. 1, p. 378-380, 1998.

[CHEUNG & YEUNG, 1998] CHEUNG, K., YEUNG, D.. *A Bayesian Framework for Deformable Pattern recognition with Application to Handwritten Character Recognition*. IEEE Transactions on Pattern Analysis and Machine Inteligence, vol. 20, n. 12, p. 1382-1388, dez. 1998.

[CONNELL, 1996] CONNEL, S..*A Comparison of Hidden Markov Model Features for the Recognition of Cursive Handwriting*, Dissertação de Mestrado, Departamento de Ciência da Computação, Universidade Estadual de Michigan, 1996.

[EL-YACOUBI, 1996] EL-YACOUBI, A.. *Modélisation Markovienne de L'ecriture Manuscrite*: Application à la Reconnaissance des Adresses Postales. Rennes, 307 p.. Tese de Doutorado, Universidade de Rennes, 1996.

[EL-YACOUBI et al., 1999] EL-YACOUBI, A. et al., *An HMM-Based Approach for Off-Line Unconstrained Handwritten Word Modeling and Recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21, n. 8, p. 752-760, ago. de 1999.

[GOVINDARAJU & MADHVANATH, 2001] GOVINDARAJU, V., MADHVANATH, S., *The Role of Holistic Paradigms in Handwritten Word Recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, n. 2, p. 149-164, fev. de 2001.

[GUILLEVIC, 1995] GUILLEVIC, D.. *Unconstrained Handwritten Recognition Applied to The Processing of Bank Cheques*. Montreal, 183 p.. Tese de Doutorado, Universidade Concórdia, 1995.

[GÜNTER, 2004a] GÜNTER, S.. *Multiple Classifier Systems in Offline Cursive Handwriting Recognition*. Bern, 184 p.. Tese de Doutorado, Universidade de Bern, 2004.

[GÜNTER, 2004b] GÜNTER, S., BUNKE, H.. *Combination of Three Classifiers with Diferent Architectures for Handwritten Word Recognition*. 9th International Workshop on Frontiers in Handwriting Recognition (IWFHR-9), Tokio, Japão, p. 63-68, 26 a 29 out. 2004.

[HALKIDI et al, 2001] HALKIDI, M., BATISTAKIS, Y., MAULIK, U.. *On Clustering Validation Techniques*. Journal of Intelligent Information Systems, vol. 17, 2001.

[HEUTE et al, 1998] HEUTE, L., et al. *A Structural/Statistical Feature-based Vector for Handwritten Character Recognition*. Pattern Recogniton Letters, vol. 19, p. 629-641, 1998.

[HIRANO et al, 1997] HIRANO, T., OKADA, Y., YODA, F.. *Structural Character Recognitoin Using Simulated Annealing*. Proceedings of the 4th International Conference on Document Analysis and Recognition, vol. 2, p. 507-510, 1997.

[HO et al, 1994] HO, T. K., HULL, J. J., SRIHARI, S. N.. *Decision Combination in Multiple Classifier Systems*. IEEE Transactions on Pattern Analisys and Machine Intelligence, vol. 16, n. 1, p. 66-75, jan. 1994.

[IAM, 2004] *The Homepage of the IAM Database*: A Database for Off-line Handwriting Recognition Research. Disponível em http://www.iam.unibe.ch/~zimmerma/iamdb/iamdb.html. Acesso em 2004.

[KAPP et al, 2004] KAPP, M.N., FREITAS, C., SABOURIN, R.. *Handwritten Brazilian Month Recognition*: An Analysis of Two NN Architectures and a Rejection Mechanism, 9th International Workshop on Frontiers in Handwriting Recognition (IWFHR-9), Tokio, Japão, p. 209-214, 26 a 29 out. 2004.

[KIM et al, 2000] KIM, J. H., KIM, K. K., NADAL, C. P., SUEN, C. Y.. *A Methodology of Combining HMM and MLP Classifiers for Cursive Word Recognition*. Proceedings of 15th International Conference on Pattern Recognition (ICPR2000), p. 319-322, 2000.

[KIMURA & SHIDHAR, 1992] KIMURA, F., SHRIDHAR, M.. *Segmentation-Recognition Algorithm for Handwritten Numeral Strings*. Machine Vision Applications, n. 5, p. 199-210, 1992.

[KIMURA et al, 1993] KIMURA, F., SHRIDHAR, M., CHEN, Z.. *Improvements of a Lexicon Directed Algorithm for Recognition of Unconstrained Handwritten Words*. Proceedings of the International Conference on Document Analysis and Recognition (ICDAR'93), p. 18-22, 1993.

[KOERICH et al, 2002] KOERICH, A. L., et al. *A Hybrid Large Vocabulary Handwritten Word Recognition System Using Neutal Networks with Hidden Markov Models*, International Workshop on Frontiers in Handwriting Recognition (IWFHR 2002), Niagara-on-the-Lake, Canada, p. 99-104, ago. 2002.

[KOERICH et al, 2003a] KOERICH, A. L., SABOURIN, R., SUEN, C. Y., *Large Vocabulary Off-line Handwritten Recognition: A Survey*, Pattern Analysis & Applications, vol. 6, n. 02, p. 100-125, 2003.

[KOERICH et al, 2003b] KOERICH, A. L., SABOURIN, R., SUEN, C. Y., *Lexicon-Driven HMM Decoding for Large Vocabulary Handwriting Recognition With Multiple Character Models*, International Journal on Document Analysis and Recognition (IJDAR), vol. 6, n. 02, p. 126-144, 2003.

[KOERICH et al, 2003c] KOERICH, A. L.. *Unconstrained Handwritten Character Recognition Using Different Classification Strategies*. IAPR International Workshop on Artificial Neural Networks in Pattern Recognition (ANNPR 2003), Firenze, Itália, set. 2003.

[MARTI & BUNKE, 2001] MARTI, U., BUNKE, H.. *Using a Statistical Language Model to Improve the Performance of an HMM-Based Cursive Handwriting Recognition System.*, Int. Journal of Pattern Recognition and Artificial Intelligence, vol 15, p. 65-90, 2001.

[MORITA, 1998] MORITA, M. E., *Estudo para a Melhoria da Correção da Inclinação da Linha De Base de Palavras Manuscritas*, Curitiba, 1998. Dissertação de Mestrado, Centro Federal de Educação Tecnológica do Paraná, 1998.

[MORITA et al, 2004] MORITA, M.E., et al.. *Segmentation and Recognition of Handwritten Dates*: An HMM-MLP Hybrid Approach, International Journal on Document Analysis and Recognition, vol. 6, p. 248-262, 2004.

[OLIVEIRA et al, 2002] OLIVEIRA, L.S., et al. *Automatic Recognition of Handwritten Numerical Strings:* A Recognition and Verification Strategy. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, n. 11, p. 1438-1454, 2002.

[OLIVEIRA & SABOURIN, 2004] OLIVEIRA, L.S., SABOURIN, R.. *Support Vector Machines for Handwritten Numerical String Recognition*, 9th International Workshop on Frontiers in Handwriting Recognition (IWFHR-9), Tokio, Japão, p. 39-44, 26 a 29 out. 2004.

[OLIVEIRA JR. et al, 2002] OLIVEIRA JR, J. J., CARVALHO, J. M., FREITAS, C. O. A., SABOURIN, R.. *Evaluating NN and HMM Classifiers for Handwritten Word Recognition*. 15th Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI´2002), Fortaleza. p. 210-217, 2002.

[OTSU, 1979] OTSU, N., *A Threshold Selection Method from Gray-level Histograms*, IEEE Transactions on Systems, Man and Cybernetics, vol. 9, n. 1, pág. 62-66, 1979.

[RABINER, 1989] RABINER, L. R., *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*, Proceedings of IEEE, vol. 77, n. 2, p. 257-286, fev. de 1989.

[STEINHERZ et al, 1998] STEINHERZ, T., RIVLIN, E., INTRATOR, N., *Offline Cursive Script Word Recognition - A Survey*, Internation Journal on Document Analysis and Recognition (IJDAR), vol. 2, p. 90-110, 1993.

[SUEN et al, 1999] SUEN, C. Y., KIU, K., STRATHY, N. W.. *Sorting and Recognizing Cheques and Financial Documents*. Document Analysis Systems: Theory and Practice, Springer, Belin, p. 173-187, 1999.

[TAPPERT et al, 1990] TAPPERT, C. C., SUEN, C. Y., WAKAHARA, T.. *The State of Art in Online Handwriting Recognition*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 12, n. 8, p. 787-808, 1990.

[TRIER et al, 1995] TRIER, O. D., JAIN, A. K., TAXT, T., *Feature Extraction Methods for Character Recognition - A Survey*, Pattern Recognition, vol. 29, pág. 641-662, 1996.

[XU et al, 1992] XU, L., KRZYZAK, A., SUEN, C. Y.. *Methods of Combining Multiple Classifiers and Their Applications to Handwriting Recognition*. IEEE Transactions on Systems, Man, and Cybernetics, vol. 22, n. 3, p. 418-435, mai. 1992.

[WANG, 1994] WANG, X. *Durationally Constrained Training of HMM Without Explicit State Duration PDF*. Instituto de Ciências Fonéticas, Universidade de Amsterdam, Proceedings, vol. 18, p. 111-130, 1994.