Universidade Federal do Rio de Janeiro

Escola Politécnica

Departamento de Eletrônica e de Computação

Algoritmos de Reconhecimento Facial Usando Projeções em Subespaços

Autor:

Rodrigo Leite Prates

Banca Examinadora:

Orientador:

Prof. Eduardo Antônio Barros da Silva, Ph.D.

Co-orientador:

José Fernando Leite de Oliveira, D.Sc.

Examinador:

Prof. Gelson Vieira Mendonça, Ph.D.

Examinador:

Prof. José Gabriel Rodriguez Carneiro Gomes, Ph.D.

DEL

Novembro de 2010

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
Escola Politécnica - Departamento de Eletrônica e de Computação
Centro de Tecnologia, bloco H, sala H-217, Cidade Universitária
Rio de Janeiro - RJ CEP 21949-900

DEDICATÓRIA

À minha família e amigos.

AGRADECIMENTOS

"Viver é acreditar e realizar o impossível."

Agradeço a Deus por ter me dado as oportunidades necessárias para que eu conseguisse atingir e finalizar mais uma etapa da minha vida.

Agradeço a meus pais, Edna Leite Santos e Germano Prates, por terem me proporcionado uma boa formação pessoal e educacional. Agradeço também a eles por estarem ao meu lado em todos os momentos da minha vida. Agradeço também à minha irmã Carolina pelo apoio e ajuda em todos os momentos.

Agradeço especialmente à minha tia Regina Prates, por ter me apoiado e financiado boa parte dos meus estudos.

Agradeço também ao meu orientador Eduardo Antônio Barros da Silva por ter me orientado neste trabalho e por ser um exemplo a ser seguido de realização profissional e pessoal.

Agradeço também ao co-orientador deste trabalho, José Fernando Leite de Oliveira, por estar sempre apto a esclarecer minhas dúvidas e ao Professor Waldir Sabino da Silva Júnior e ao doutorando Gabriel Matos Araújo por também me ajudarem a tornar esse trabalho possível através de seus conhecimentos.

Agradeço a todos os professores que tive, tanto na faculdade quanto no colégio, especialmente ao coordenador do curso de Engenharia Eletrônica, Carlos José Ribas D'Avila, mais conhecido como Casé, ao chefe do departamento, José Paulo Brafman, ao professor Ricardo Rhomberg Martins por terem resolvido os problemas que aconteceram durante o meu curso de graduação. Foi com o conhecimento transmitido por eles que consegui finalizar mais uma etapa!

Agradeço a todos os meus amigos e colegas da faculdade, especialmente ao pessoal do LPS pelos momentos divertidos e descontraídos tanto no laboratório como no Burguesão. Agradeço ao CNPq pelos três anos de bolsa de iniciação científica.

Enfim, gostaria de agradecer a todos que contribuíram para a minha vida. Espero que consiga dar uma contribuição ainda maior para o mundo.

RESUMO

O presente trabalho está inserido nas área de reconhecimento de padrões e processamento de imagens. Em particular, estamos interessados no problema de reconhecimento de faces. Alguns algoritmos de reconhecimento de padrões, tais como o *DLBP* (*Discriminant Localized Binary Projections*), se fundamentam na análise de padrões binários ortogonais, que vem ganhando popularidade devido a sua eficiência computacional e grande poder discriminativo. Estas técnicas possuem resultados promissores em relação aos métodos clássicos. Este trabalho consiste na implementação e teste do algoritmo *DLBP*. Foram usadas duas bases de imagens para treinamento e teste, onde as faces foram extraídas e alinhadas antes do processo de treinamento. Um outro algoritmo, o *CFA* (*Redundant Class-Dependence Feature Analysis*), foi utilizado para fazer uma comparação com os resultados do *DLBP*. Alguns testes foram realizados e os resultados correspondentes compilados usando o Matlab.

Palavras-Chave: reconhecimento, imagens, projeções, DLBP, extração de características.

ABSTRACT

This work addresses the issues of pattern recognition and image processing. In particular, we are interested in the face recognition problem. Some pattern recognition algorithms, such as the *DLBP* (*Discriminant Localized Binary Projections*) are based on the analysis of orthogonal binary patterns, which is gaining popularity due to its computational efficiency and high discriminative power. These techniques have promising results compared to traditional methods. In this work, the *DLBP* algorithm was implemented and tested. Two image databases were used for trainning and testing, where the faces were extracted and aligned before the trainning process. The results obtained by the *DLBP* algorithm were compared with those of a state-of-the-art algorithm, the *CFA* (*Redundant Class-Dependence Feature Analysis*). Some tests were performed and the corresponding results compiled by using Matlab.

Key-words: recognition, images, projections, DLBP, features extraction.

SIGLAS

DLBP - *Discriminant Localized Binary Projections*

CFA - *Redundant Class-Dependent Feature Analysis*

NMF - *Non-negative Matrix Factorization*

PCA - *Principal Component Analysis*

LDA - *Linear Discriminant Analysis*

MFA - *Marginal Fisher Analysis*

OpenCV - *Intel® Open Source Computer Vision Library*

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Neste capítulo, serão apresentados os seguintes tópicos: tema, delimitação, justificativa, objetivos, organização e notações.

1.1 Tema

Este trabalho está inserido nas áreas de reconhecimento de padrões e processamento de imagens. Em particular, estamos interessados no problema de reconhecimento de faces.

Reconhecer um objeto, em muitos casos, ainda é um processo muito complexo e de alto custo computacional. Daí, a cada momento, mais e mais propostas vão surgindo no intuito de simplificar o problema a ponto de tornar possível sua utilização em vários produtos.

Dentre todos os algoritmos existentes, alguns deles são baseados em padrões binários ortogonais, que vem ganhando popularidade devido à sua eficiência computacional e grande poder discriminativo, como o *DLBP*. Essas técnicas parecem então apresentar melhores resultados que os métodos clássicos como a análise de componentes principais.

1.2 Delimitação

O foco do trabalho é então o estudo comparativo do algoritmo de *DLBP* com outro algoritmo de reconhecimento de faces. O método de detecção em si não é o problema, e por isso será usado o algoritmo de Viola-Jones para se encontrar as

faces. Essa escolha se deve ao fato de já existir o algoritmo pronto na biblioteca do *software* [1]. Nesta etapa, o algoritmo estado-da-arte *CFA* foi escolhido como o outro método para a comparação, visto este já ter sido implementado e testado.

1.3 Justificativa

Um sistema de reconhecimento de faces não é utilizado apenas para procurar os dados de uma pessoa através de sua face, existem várias outras aplicações práticas para ele. Dentre elas pode-se citar o controle de acesso a prédios ou outro sistema de segurança, a interação entre homem e computador, a videoconferência, o diagnóstico de doenças que provocam alterações faciais, além de usos sociais, como busca de crianças desaparecidas em locais públicos ou de indivíduos procurados pela lei, além de ajudar a deficientes visuais. O reconhecimento de faces é também um problema científico interessante do ponto de vista da compreensão do sistema de visão humano.

1.4 Objetivos

O objetivo deste trabalho é implementar, testar e analisar o algoritmo *DLBP*, buscando com isto apresentar um método mais simples e de treinamento rápido, mais robusto a variações na qualidade e desalinhamento das imagens, robusto ao escalamento e translação, de baixa complexidade computacional e, principalmente, com funcionamento mais relacionado à forma com que as pessoas reconhecem rostos, ou seja, o reconhecimento baseado em partes da imagens ao invés da imagem toda do rosto.

1.5 Organização do Trabalho

Este trabalho está organizado da seguinte forma:

- no Capítulo 2, serão apresentados conceitos gerais de reconhecimento de padrões, detecção de padrões faciais e o estado da arte em reconhecimento de faces.
- no Capítulo 3, serão apresentados o sistema proposto, uma breve descrição do algoritmo *CFA*, os requisitos necessários para a execução do projeto, a etapa de pré-processamento, os testes realizados e os resultados obtidos.

- no Capítulo 4, serão apresentadas a conclusão do projeto e propostas para trabalhos futuros.

1.6 Notação

Usamos a seguinte notação:

- O símbolo $\{\cdot\}^T$ denota o transposto.
- O símbolo $\mathbf{I}_N$ denota a matriz identidade $N \times N$.
- O símbolo $\mathbb{R}$ representa o espaço real.
- A norma euclidiana é denotada por $\|\cdot\|^2$.
- O símbolo $O(\cdot)$ denota a ordem de complexidade computacional.

Capítulo 2

Fundamentos

Neste capítulo, serão apresentados os seguintes tópicos: conceitos gerais em reconhecimento de padrões, técnicas de detecção de padrões faciais e estado da arte em reconhecimento de faces.

2.1 Conceitos Gerais em Reconhecimento de Padrões

A habilidade de reconhecer padrões é extremamente desenvolvida nos seres humanos e em alguns animais. O ser humano é hábil em reconhecer rostos, vozes, caligrafias e, até mesmo, estados de humor de pessoas conhecidas. Alguns animais também têm algumas destas características desenvolvidas, tais como os cães farejadores. O grau de refinamento do reconhecimento de padrões, por parte do ser humano, pode chegar a ponto de se distinguir uma pintura de um grande mestre daquela feita por um exímio falsário ou, ainda mais, pode estabelecer uma tomada de decisão por parte de um operador em um dia de grande movimento em uma bolsa de valores. Assim sendo, pode-se dizer que padrões são os meios pelos quais o mundo é interpretado e, a partir dessa interpretação, elaboram-se atitudes e decisões [2].

Percebe-se que, nos exemplos citados, tal facilidade no reconhecimento de padrões está diretamente vinculada aos estímulos aos quais o indivíduo foi exposto anteriormente. Isso leva a supor que a estrutura selecionada pela evolução biológica para desempenhar bem a tarefa de reconhecimento de padrões incorpora alguma forma de aprendizado e evolui com a experiência.

Um dos grandes desafios da humanidade neste início de século é o de desenvolver máquinas que tenham tais comportamentos. Tarefas de reconhecimento de voz, imagens e audio, já estão em fase de desenvolvimento há bastante tempo, mas

seu desempenho ainda não se assemelha ao do ser humano.

Algumas aplicações do reconhecimento de padrões são: identificação através de impressões digitais e análise da íris [3], diagnósticos médicos, análise de imagens aeroespaciais [4], visão computacional, diagnósticos pré e pós-natal e certos diagnósticos de câncer, reconhecimento de voz, análise de peças para manutenção preventiva, análise de eletrocardiogramas, sinais de radar dentre outras [2].

Entende-se por padrões as propriedades que possibilitam o agrupamento de objetos semelhantes dentro de uma determinada classe ou categoria, mediante a interpretação de dados de entrada, que permitam a extração das características relevantes desses objetos [2]. Entende-se por classe de um padrão um conjunto de atributos comuns aos objetos de estudo. Assim, reconhecimento de padrões pode ser definido como sendo um procedimento em que se busca a identificação de certas estruturas nos dados de entrada em comparação com estruturas conhecidas e sua posterior classificação dentro de categorias, de modo que o grau de associação seja maior entre estruturas de mesma categoria e menor entre as categorias de estruturas diferentes.

Um sistema para reconhecimento de padrões engloba três grandes etapas: representação dos dados de entrada e sua mensuração, extração das características e finalmente identificação e classificação do objeto em estudo. A primeira etapa refere-se à representação dos dados de entrada que podem ser mensurados a partir do objeto a ser estudado. Esta mensuração deverá descrever padrões característicos do objeto, possibilitando a sua posterior inclusão numa determinada classe: a classificação do objeto. O vetor que caracteriza um objeto seria:

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \tag{2.1}$$

onde: $\{x_1, x_2, x_3, \ldots, x_N\}$ são suas características.

A segunda etapa consiste na extração de características intrínsecas e atributos do objeto e conseqüente redução da dimensionalidade dos dados de entrada. É a fase da extração das características. A escolha das características é de fundamental importância para um bom desempenho do classificador. Esta escolha é feita objetivando os atributos que se pretende classificar. Exige-se, portanto, um conhecimento específico sobre o problema em estudo. Nesta etapa, os objetivos básicos

são: a redução da dimensionalidade do vetor característico, sem que isto implique em perda de informação que possa ser relevante para a classificação, objetivando a redução do esforço computacional e a seleção das características significativas para a tarefa de classificação. A terceira etapa em reconhecimento de padrões envolve a determinação de procedimentos que possibilitem a identificação e classificação do objeto em uma classe de objetos.

O Extrator de Características tem como função determinar e extrair as características mais significativas que contribuam para a descrição do objeto, dentre as várias características que possam descrevê-lo.

A seguir, classifica-se o objeto. Nesta etapa, o classificador "aprende" a distinguir dentre as classes, aquela à qual o objeto pertence. A figura 2.1 ilustra as fases do reconhecimento de padrões.

Se o treinamento do classificador exigir amplo conhecimento "a priori" da estrutura estatística dos padrões a serem analisados e o padrão de entrada for identificado como membro de uma classe pré-definida pelos padrões de treinamento, o classificador será chamado de Classificador Paramétrico [2]. Por outro lado, se o classificador utilizar determinado modelo estatístico, ajustando-se mediante processos adaptativos e a associação entre padrões se fizer com base em similaridades entre os padrões de treinamento, o classificador será chamado de Classificador Não-Paramétrico [2].

A grande dificuldade na implementação de um projeto de reconhecimento de padrões está justamente na escolha da técnica adequada para que as fases do reconhecimento de padrões ocorram de modo a representar satisfatoriamente os fenômenos do mundo real.

Essa escolha passa pelos critérios de abordagem e supervisão [2]. As abordagens estão divididas basicamente em 3 tipos: Casamento de Moldes (do inglês, *Template Matching*), onde a classificação de um certo padrão é feita através da comparação com um modelo previamente armazenado, Casamento de Características (do inglês, *Feature Matching*), onde a caracterização de padrões é feita mediante algumas "características principais" inerentes aos elementos desta classe. Padrões pertencentes a uma mesma classe possuirão propriedades comuns de discriminação dessa classe. Desta forma, quando um padrão desconhecido é observado pelo sistema, suas características são extraídas e comparadas com aquelas armazenadas como discriminates das classes. O sistema então, "classificará" este novo padrão em uma

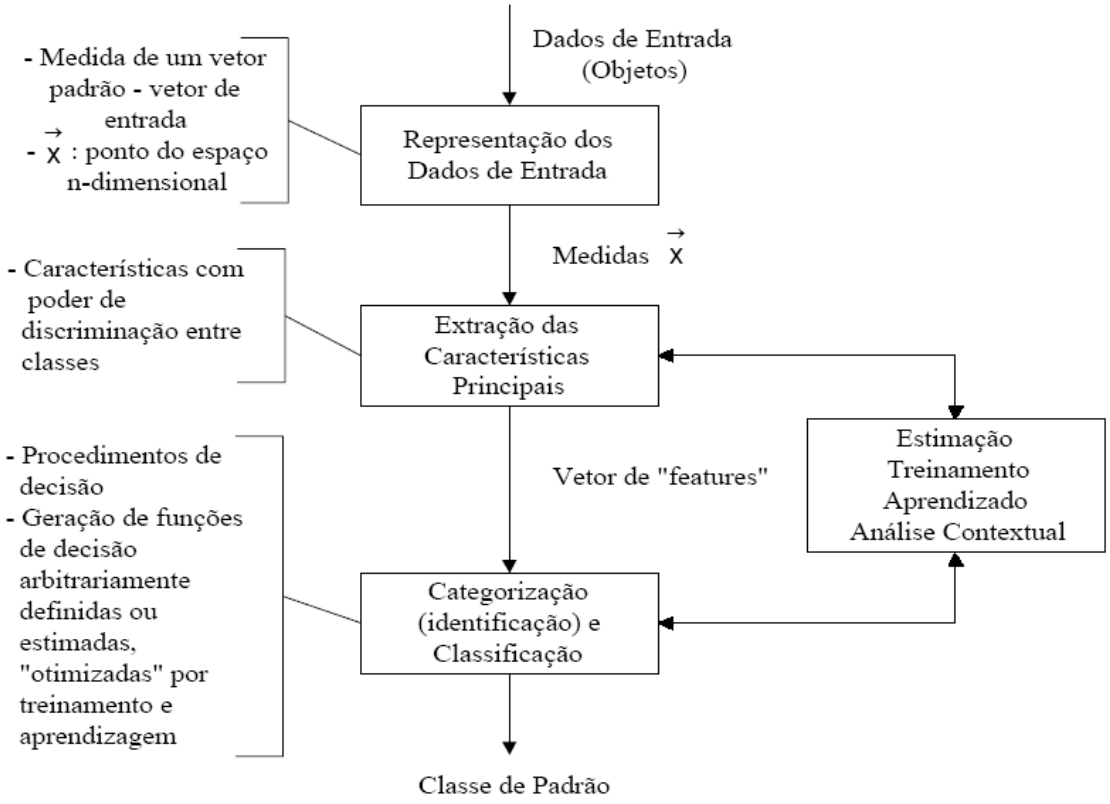

Figura 2.1: Fases do reconhecimento de padrões.

das classes existentes ou então designará o objeto a uma nova classe. Casamento de Agrupamentos (do inglês, *Clustering Matching*), onde os padrões de uma classe são vetores de números reais e a classe do padrão pode ser estabelecida segundo formas do agrupamento, "clusters", desses pontos no plano. Havendo uma separação entre os pontos de forma clara, técnicas simples podem ser empregadas, tais como "distância-mínima". Temos, exemplos para essas três abordagens na tabela 2.1.

Tabela 2.1: Os três tipos de abordagem no reconhecimento de padrões e exemplos de artigos que os usaram.

Abordagens	**Exemplos**
Template Matching	*Template Matching Using Fast Normalized Cross Correlation* [5] *Fast Normalized Cross Correlation for Defect Detection* [6]
Feature Matching	*DLBP* [7] *Non-negative Matrix Factorization Methods and their Applications* [8]
Cluster Matching	*K-Means* [2] ou *Linde-Buzo-Gray* (*LBG*) [9]

Quanto ao critério de supervisão, o algoritmo é dito supervisionado quando

os padrões representativos de cada classe estão disponíveis e o sistema reconhece padrões por meio de esquemas de adaptação. Exemplos de algoritmos utilizados no reconhecimento supervisionado são: *perceptron*, gradiente, erro quadrático mínimo, etc [2]. Quando os padrões representativos não estão disponíveis, o algoritmo é dito não-supervisionado. Como exemplo podemos citar o método de *Self-Organizing Maps* baseado em rede neural híbrida [2].

Dada essa introdução aos algoritmos de reconhecimento de padrões, podemos agora nos aprofundar nas técnicas usadas para reconhecer padrões faciais. Para tanto é necessário, antes, uma explicação sobre os métodos de detecção de faces, usados para se extrair da imagem os dados necessários para o posterior tratamento e classificação.

2.2 Técnicas de Detecção de Faces

Umas das tarefas que devem ser realizadas na maioria dos Sistemas de Reconhecimento de Faces é detectar a presença da face em uma determinada imagem. Detectar a face antes de detectar cada característica em particular poupa muito trabalho, uma vez que a maioria dos algoritmos se baseia na procura por tais elementos em toda a imagem. A vantagem de se detectar a face, em um primeiro momento, é que após esta fase a procura pelas características fica limitada apenas a uma determinada região da imagem.

Na Tabela 2.2, conforme [10], apresentamos alguns problemas encontrados para detectarmos uma face. As técnicas de detecção de padrões faciais são classificadas em [10]:

- Métodos Baseados em Conhecimento: são aqueles que utilizam alguma base de regras estabelecida a partir do conhecimento prévio sobre o problema, ou seja, métodos que possuem regras que definem o que é uma face, de acordo com o conhecimento do pesquisador. Este método sofre de algumas desvantagens inerentes a construção do conjunto de regras. Se as regras forem muito gerais, corre-se o risco de o resultado apresentar muitos falsos positivos, ou seja, elementos erroneamente identificados como faces. O inverso também é verdade, ou seja, um conjunto de regras muito específico que não permite detectar faces se estas não satisfizerem todas as regras [10]. Como exemplo dessa abordagem podemos citar a técnica de Yang e Huang [10], a qual utiliza um método de

Tabela 2.2: Exemplos de problemas em detecção de face.

Problema	**Descrição**
Pose	as imagens de face variam de acordo com a posição da câmera que registrou a imagem.
Expressão Facial	a expressão da face influência diretamente na aparência da imagem da face.
Presença de Elementos Estruturais	a presença de elementos como barba, bigode e óculos que podem modificar as características em termos de tamanho, luminosidade, etc.
Oclusão	no caso de imagens feitas em ambientes não controlados as faces podem aparecer, parcial ou totalmente sobrepostas, por objetos ou até mesmo por outras faces.

detecção de faces baseado no conhecimento, implementado com o uso de conjuntos de regras hierárquicas. Essas regras são aplicadas em dois níveis onde no primeiro, o objetivo é detectar os possíveis candidatos a faces, e no segundo nível, tenta-se validar os elementos extraídos do primeiro nível.

- Métodos Baseados em Características Invariantes: são as técnicas que tem por objetivo encontrar características invariantes da face. Particularmente, estes métodos são inspirados na capacidade que os seres humanos possuem de identificar objetos independentes do ponto de vista. A principal desvantagem de tal abordagem é que tais características podem ser corrompidas devido às condições de iluminação ou algum tipo de ruído. A cor da pele e a textura da face são as principais características invariantes que podem ser utilizadas para separar a face de outros objetos presentes em uma cena [10]. Com relação à face humana, constatou-se que a cor da pele, independente de suas variações (branca, negra, amarela, etc), forma um *cluster* no espaço de cores, podendo ser modelado por uma distribuição gaussiana. Já a textura, assim como a cor, é independente do ponto de vista. Portanto, estas características pode ser exploradas para detectar a presença de uma face em uma imagem e classficar regiões como sendo de face ou não [10].

- Métodos Baseados em *Templates*: são as técnicas de busca por um determinado objeto dentro da imagem. Uma das maneiras mais comuns de modelar a forma

de um objeto é descrevê-lo através de seus componentes geométricos básicos, como círculos, quadrados ou triângulos. A detecção consiste então em achar a melhor correspondência, definida através de uma função energia, entre o objeto presente na imagem e o seu molde (*template*).

- Métodos Baseados na Aparência: são os métodos que não utilizam nenhum conhecimento *a priori* sobre o objeto ou caracteristica a ser detectada. Nesta classe de algoritmos, surgem os conceitos de aprendizagem e treinamento, uma vez que as informações necessárias para realizar a tarefa de detecção são retiradas do próprio conjunto de imagens sem intervenção externa. A exemplo temos o método de *eigenfaces* baseado na transformada de Karhunen-Loève (KLT), ou PCA (*Principal Component Analysis*). A KLT é usada para achar os vetores que melhor descrevem a distribuição de imagens dentro do espaço de imagens inteiro. Temos também exemplos usando redes neurais e Modelos de Markov Escondidos (*Hidden Markov Models*).

De tantos métodos existentes para esse tipo de abordagem, o presente trabalho fez uso do detector de **Viola-Jones**, um método muito usado para detecção em tempo real e de alta taxa de acerto [11].

Até a metade dos anos 90, a maior parte dos trabalhos sobre segmentação se concentrou na segmentação de apenas uma face em fundos simples ou complexos. As abordagens incluíam o uso de um modelo (*template*) de um face inteira, modelos baseados em características invariantes e redes neurais. Abordagens recentes podem detectar faces e suas poses em fundos diversos [12].

Em especial, a abordagem de Viola-Jones, é considerada o estado-da-arte em detecção de faces.

Esse método trouxe três grandes contribuições [11]:

- Uma nova representação da imagem, chamada de *Integral Image*, que permite um rápido cálculo das *Features*, características da imagem usadas no classificador.

- Um simples mas eficiente classificador criado através da seleção de poucas *Features*, de um conjunto grande de *Features*, pelo algoritmo de AdaBoost.

- Um método que combina sucessivamente, em uma estrutura em cascata, classificadores cada vez mais complexos, dividindo o processo de detecção em estágios

onde só é reconhecido como uma face, a imagem que passar por todas as etapas (filtros) que descartam o que não fizer parte de uma face.

A figura 2.2 ilustra um detector de faces que utiliza o método Viola-Jones.

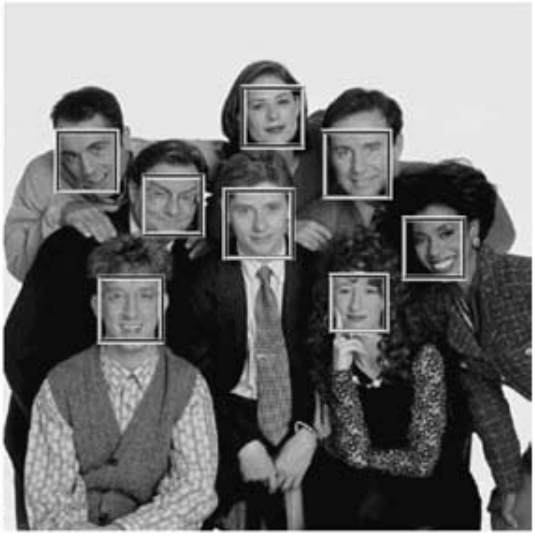

Figura 2.2: Exemplo de uma imagem processada por um detector de faces que utiliza o método Viola-Jones.

2.3 Estado da Arte em Reconhecimento de Faces

Os humanos baseiam-se freqüentemente na face para o reconhecimento de indivíduos. Por sua vez, os avanços obtidos nas últimas décadas na capacidade de computação permitem um reconhecimento similar, mas de forma automática.

Os primeiros algoritmos de reconhecimento da face utilizavam modelos geométricos simples [13], mas o processo de reconhecimento já atingiu um nível de maturidade que lhe permite apresentar-se como uma ciência de representações matemáticas sofisticadas e processos de comparação.

O reconhecimento automatizado da face é um conceito relativamente novo. Desenvolvido na década de 60, o primeiro sistema semi-automatizado para o reconhecimento da face exigia que fossem localizadas características nas fotografias (por exemplo, olhos, orelhas, nariz e boca) antes do sistema calcular distâncias para um ponto de referência comum. Esse ponto de referência era comparado com os dados disponíveis. Na década de 70, utilizaram-se 21 marcadores específicos (incluindo a cor do cabelo e a espessura dos lábios) para automatizarem o reconhecimento. O

problema com estas duas soluções iniciais residia no fato de as medições e localizações terem de ser calculadas manualmente [13]. Em 1988, foi aplicada a Análise de Componentes Principais (*PCA*), uma técnica de cálculo padrão, ao problema do reconhecimento da face. Isto foi um marco, pois demonstrou-se que eram necessários menos de uma centena de valores para codificar com exatidão uma imagem da face adequadamente normalizada e alinhada [13]. Em 1991, foi descoberto que, na utilização de técnicas usando componentes principais, o erro residual, da distância entre uma imagem e sua projeção no *Face Space* (espaço formado pelas *eigenfaces*), podia ser utilizado para detectar faces em imagens [14]. Isso porque as faces, em geral, são mais parecidas entre si do que o *background* da foto. Esta descoberta permitiu a criação de sistemas automatizados de reconhecimento da face em tempo real. Esta abordagem estava limitada de certa forma por fatores ambientais, mas acabou por motivar um grande interesse para o desenvolvimento futuro de tecnologias de reconhecimento automatizado da face. Essa tecnologia começou a chamar a atenção das pessoas devido à reação dos meios de comunicação a uma implementação experimental no *January 2001 Super Bowl*, que capturou imagens de vigilância e as comparou com uma base de dados de conteúdos digitais. Esta demonstração iniciou uma análise sobre a forma de utilizar a tecnologia para suportar determinadas necessidades nacionais, considerando ao mesmo tempo as preocupações sociais e de privacidade [13].

Atualmente, a tecnologia de reconhecimento da face é utilizada para o combate à fraude com passaportes [15], para o reforço de legislação, para a identificação de crianças desaparecidas, para minimizar as fraudes de benefícios/identidade, em câmeras e *webcams*, dentre muitas outras [13]. Uma das novidades em destaque no momento são as câmeras e *webcams* com capacidade para detecção e reconhecimento de faces. Os equipamentos não só detectam os rostos dentro da foto como podem "lembrar" deles. Com isso a câmera pode otimizar o foco e a exposição para que esse rosto detectado apareça bem focado e com brilho. Isso faz com que seja fácil tirar fotos boas de uma pessoa dentro de um grupo.

As tecnologias acima tratam do reconhecimento em imagens de duas dimensões. O mesmo pode ser feito para imagens de três dimensões e para vídeo. Então, dependendo do tipo do dado de entrada, imagens 2D, 3D ou vídeo, temos a disposição uma gama de algoritmos para o reconhecimento, como por exemplo:

1. Imagens 2D:

- O método por PCA (*Principal Component Analysis*), onde dado um vetor s-dimensional representativo da face, tenta-se achar uma nova representação de dimensão t, em um novo subespaço de dimensões menores que s, $t<s$. Esta redução de dimensões remove informação que não é útil e decompõe a estrutura da face em componentes ortogonais (não correlacionados), conhecidos por *eigenfaces*. A figura 2.3 ilustra as imagens de algumas *eigenfaces*. Cada imagem da face pode ser representada como a soma das *eigenfaces* (vetor de características) que estão armazenadas como vetores 1D. Uma dada imagem a pesquisar será então comparada com uma galeria de imagens, medindo a distância entre os seus respectivos vetores de características [14].
- O método de EP (*Evolutionary Pursuit*) [16], uma abordagem onde se tenta achar o melhor conjunto de vetores de projeção que maximizem uma função custo, medindo ao mesmo tempo a acurácia e a generalidade do classificador. Por causa da grande dimensionalidade dos possíveis vetores de projeção, é usado um algoritmo genético para a seleção da melhor base [16].
- Os métodos de *Kernel* que são generalizações dos métodos lineares como o método de PCA, considerando subespaços não lineares. Exemplos são encontrados em *Kernel Independent Component Analysis* [17] e *Nonlinear Component Analysis as a Kernel Eigenvalue Problem* [18].

2. Imagens 3D:

   - No reconhecimento de faces 3D, o desafio está na classificação das faces independentemente das deformações superficiais causadas pelas expressões faciais. Inicialmente características como o mapa de profundidade (*range image*) e a textura são extraídas da face. Depois é feito um pré-processamento no mapa de profundidade para remoção de certas partes como o cabelo, que podem complicar o reconhecimento. Finalmente, uma forma canônica da face é calculada, insensível tanto à orientação da cabeça quanto à expressão da face. Isso simplifica muito o processo de reconhecimento. Exemplos são *3D Face Recognition without Facial Surface Reconstruction* [19] e *Expression-invariant 3D face reconstruction* [20].

3. Vídeo:

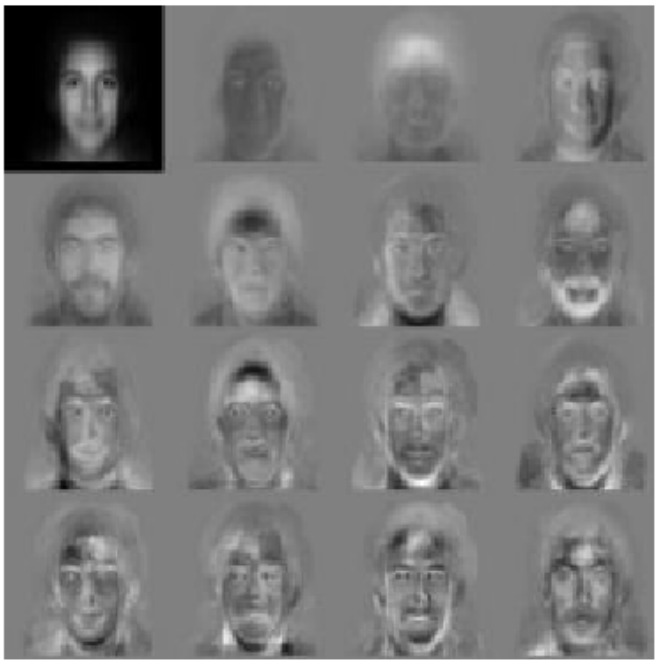

Figura 2.3: Exemplos de vetores de uma base do método de *Eigenfaces*.

- No reconhecimento facial em vídeo, tradicionalmente, este era tratado como uma coleção de imagens, que eram extraídas dele e comparadas com outras imagens usando métodos de reconhecimento de imagens faciais. Como as imagens em vídeo são de baixa qualidade e resolução, novos métodos foram explorados. Atualmente, as técnicas de reconhecimento em vídeo levam em consideração resultados calculados sobre vários quadros ao invés de um único quadro, o que torna o método mais parecido com a forma de reconhecer do seres humanos, e usam mecanismos neuro-associativos eficientes para permitir aprendizado rápido e associação de estímulos a valores sinápticos, permitindo o uso de técnicas como rede neural. Como exemplos temos *Video-Based Framework for Face Recognition in Video* [21] e *Probabilistic Recognition of Human Faces from Video* [22].

Dentro da classe de algoritmos de *feature extraction*, os métodos de *subspace learning* [7] vem sendo muito usados na classificação de padrões, principalmente pela sua simplicidade computacional e analítica. A maioria dessas técnicas, como *Principal Component Analysis* (PCA), *Linear Discriminat Analysis* (LDA) e o *Marginal Fisher Analysis* (MFA), são holísticas [7]. Isso é, todas as entradas dos vetores de projeção são não nulas e o cálculo computacional de cada *feature* no sub-espaço de

dimensões reduzidas deve explorar todas as *features* no espaço de *features* original. Técnicas de representação esparsas foram estudadas com o objetivo de obter vetores de projeção com poucos elementos não nulos. O algoritmo de *Non-negative Matrix Factorization* (NMF) foi o trabalho pioneiro nesse sentido. Ele impõe, no treinamento, que os vetores de projeção sejam positivos. Isto permite que a combinação dos vetores de projeção, ou melhor que, as bases formem imagens positivas [7]. Entretanto, a maioria desses algoritmos, de redução de dimensionalidade com representação dos vetores de forma esparsa ou binária, tiveram como objetivo a reconstrução de imagens. Isso significa que os mesmos não eram ótimos no sentido de maximizar a classificação.

Para superar esse problema, uma nova abordagem surgiu onde as bases são binárias, com cardinalidade (número de elementos não-nulos) definido pelo usuário, o que permite obter as *features* por simples soma de *pixels*, e onde a redução da dimensionalidade é feita de forma supervisionada. Este algoritmo, chamado *Discriminat Localized Binary Projections* (DLBP), foi escolhido para este projeto por apresentar essas inovações que se mostram muito promissoras e que serão detalhadas no próximo capítulo.

Capítulo 3

Metodologia

Neste capítulo, serão apresentados os seguintes tópicos: sistema proposto, descrição do algoritmo *CFA*, requisitos, preprocessamento, testes realizados e resultados.

3.1 Introdução e Sistema Proposto

O presente trabalho é baseado no algoritmo *DLBP*. Esse algoritmo foi extraído do artigo de congresso *Learning Semantic Patterns with Discriminant Localized Binary Projection* [7].

Para a apresentação do algoritmo, assumimos a existência de um conjunto de imagens, de m *pixels*, na forma vetorial $m \times 1$ $\{\mathbf{x}_i \mid \mathbf{x}_i \in \mathbb{R}^m\}_{i=1}^N$ e suas respectivas classes $\{c_i \mid c_i \in \{1,...,n_c\}\}_{i=1}^N$, onde n_c é a n-ésima classe. Como na prática o valor de m é muito grande, é normalmente necessário transformar os dados do espaço de entrada para um espaço de dimensão menor. Este problema de dimensionalidade é facilmente percebido quando observamos que, para uma imagem de tamanho moderado, o valor de m não é menor que 10.000 [7].

Os modelos clássicos de redução de dimensionalidade usualmente determinam a matriz de projeção $\mathbf{P} = [\mathbf{p}_1,\mathbf{p}_2,...,\mathbf{p}_d] \in \mathbb{R}^{m \times d}$, que mapeia as imagens, do espaço de alta dimensionalidade original, $x \in \mathbb{R}^m$, para um outro espaço (de *features*), de dimensionalidade menor, $\mathbf{y} \in \mathbb{R}^d$, através da equação $\mathbf{y} = \mathbf{P}^T\mathbf{x}$. Geralmente, não existe nenhuma restrição quando aos valores das entradas dos vetores de projeção $\mathbf{p}_i$ e assim todas as entradas de $\mathbf{p}_i$ podem ser diferentes de zero. Entretanto, evidências mostram que essa não é a forma mais parecida com a forma humana de reconhecer padrões. Este problema foi então estudado e o algoritmo *Non-negative Matrix*

Factorization foi proposto, onde bases não negativas podem ser geradas [7].

Existem evidências de que o reconhecimento de faces feito pelos seres humanos é baseado somente em partes da face [7]. Assim bastam detalhes como nariz, olhos ou boca para que uma pessoa reconheça a outra. Isso é uma evidência a favor do uso de bases esparsas, localizadas, para a extração das *features*. Outra observação é que para vetores de projeção gerais, com entradas positivas e negativas, as *features* calculadas são facilmente afetadas pelo desalinhamento das imagens, ou em outras palavras, pela translação ou pelo escalamento da imagem [7]. Baseado nas evidências acima, as restrições feitas pelo algoritmo DLBP são:

1. Fazer com que as entradas dos vetores de projeção, ou bases, sejam binárias.
2. As *features* calculadas pela matriz de projeção devem ser ótimas no sentido da classificação.
3. As bases devem ser espacialmente localizadas e ortogonais entre si.

O problema pode ser formalmente apresentado da seguinte forma: dado um conjunto de imagens da forma $\mathbf{X} = \{x_i\}_{i=1}^{N}$ e seu correspondente conjunto, inicial, de classes da forma $\{c_i\}_{i=1}^{N}$, buscamos por um conjunto de vetores de projeção $\mathbf{P} = [\mathbf{p}_1,\mathbf{p}_2,...,\mathbf{p}_d]$ que satisfaça:

$\mathbf{P} = \arg\max F(\mathbf{P})$, onde:

1. $\mathbf{p}_i(k)$ =1 ou 0, $i = 1, 2, ..., d$ e $k = 1, 2, ..., m$
2. $\mathbf{p}_i \perp \mathbf{p}_j$, $\forall i \neq j$
3. $Card(\mathbf{p}_i) \leq N_s, \forall i$

onde $F(\mathbf{P})$ é a função objetivo que mede o desempenho de classificação da matriz de projeção $\mathbf{P}$. $\mathbf{p}_i \perp \mathbf{p}_j$ é a imposição de que os vetores devem ser perpendiculares. $Card(\mathbf{p}_i)$ é a cardinalidade (número de elementos não nulos) dos vetores de projeção $\mathbf{p}_i$, e N_s é o número máximo de elementos não nulos.

A função objetivo pode ter diferentes definições dependendo de seu propósito. Nesse trabalho, esta função contém o cálculo das distâncias euclidianas de *features* de uma classe e de *features* de classes diferentes, como ilustra a figura 3.1. O cálculo final é o somatório sobre todas as *features*, como expresso pela equação (3.1):

$$F(\mathbf{P}) = \sum_{i=1}^{N}(- \sum_{c_j=c_i} \parallel \mathbf{P}^T\mathbf{x}_i - \mathbf{P}^T\mathbf{x}_j \parallel^2 /(n_{c_i} - 1) + \\ + \sum_{c_i \neq c_j} \parallel \mathbf{P}^T\mathbf{x}_i - \mathbf{P}^T\mathbf{x}_j \parallel^2 /(N - n_{c_i})) \tag{3.1}$$

Dessa forma, este é um problema de aprendizado supervisionado clássico, chamado de *Integer Optimization Problem* [7]. Como o número de parâmetros a se otimizar é muito grande, fica muito complicado otimizar (maximizar nesse caso) diretamente a função objetivo, isto é, achar $\mathbf{P}$ que:

- Minimize o somatório dentro de cada classe ($c_j = c_i$), minimizando a distância dentro das classes.
- Maximize o somatório para classes diferentes ($c_j \neq c_i$), maximizando a distância entre classes.

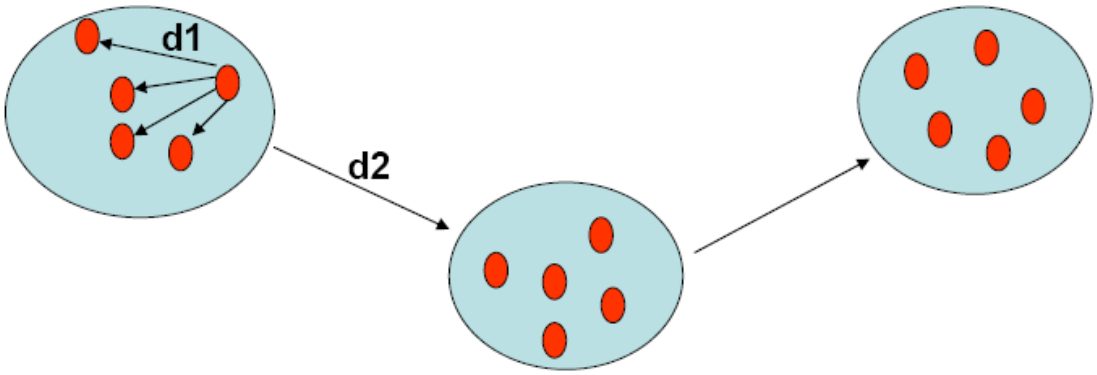

Figura 3.1: Distâncias entre *features*.

Por isso é desejável que se tenha um procedimento que aproxime a solução ótima.

3.1.1 Procedimento para Solução Ótima

Nesta seção, apresentamos um procedimento para uma aproximação da solução do problema da equação 3.1. Dado que os vetores de projeção são ortogonais e

de entradas binárias, deve existir pelo menos uma entrada não nula em cada coluna da matriz de projeção $\mathbf{P}$. Por isso, o problema acima pode ser igualmente reformulado em um problema de *feature clustering*. Cada vetor de projeção $\mathbf{p}_i$ da matriz de projeção $\mathbf{P}$ corresponde a uma classe C_i, isto é, este vetor funciona como um indicador para o *clustering*. Assim, se a entrada k do vetor de projeção i for igual a 1 ($\mathbf{p}_i(k) = 1$), isso equivale a passar a feature k ($\mathbf{f}_k$) para a classe i (C_i). Esse problema de *feature clustering* pode ser formalmente apresentado como:

Feature Clustering Para um conjunto de *features* $\{f_k\}_{k=1}^{m}$, buscamos por um método que separe essas *features* em $(d + 1)$ classes $C_0, C_1, ..., C_d$, através da maximização de uma função objetivo $F(\mathbf{P})$, além de satisfazer as condições $\mathbf{p}_i(k) = 1 \Leftrightarrow f_k \in C_i$, $\mathbf{p}_i(k) = 0 \Leftrightarrow f_k \notin C_i$, $\forall i, k$ e o número de *features* em cada classe não ser maior que N_s.

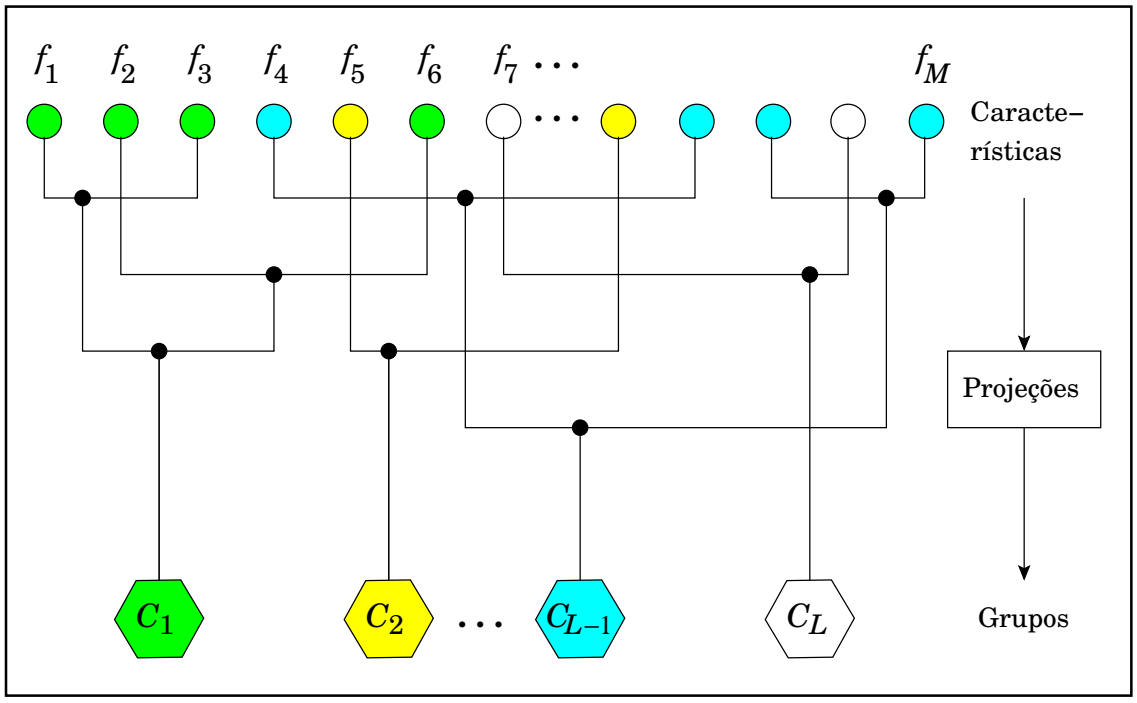

Figura 3.2: *Clustering*.

No processo de classificação descrito acima, as *features* na classe C_0 não aparecem nos vetores de projeção e contribuem pouco para a separação das diferentes classes.

Agora será apresentado um algoritmo Voraz (*Greedy*), usado para, progressivamente, combinar as classes, ao passo que diminui o valor da função objetivo ao longo do processo e satisfaz todas as restrições. Os algoritmos de *greedy* são aqueles

que tentam achar o "ótimo" a cada passo, não se preocupando com passos futuros. Espera-se que, obtendo mínimos ou máximos locais em cada etapa, se chegue a um mínimo ou máximo global ao final do processo [23].

Greedy Solution A função objetivo na equação equação. (3.1) pode ser reescrita na forma

$$F(\mathbf{P}) = Tr(\mathbf{P}^T\mathbf{S}\mathbf{P}) = \sum_{k=1}^{d} \mathbf{p}_k^T\mathbf{S}\mathbf{p}_k \tag{3.2}$$

onde

$$\mathbf{S} = \sum_{i=1}^{N}(\frac{1}{N-n_{c_i}}\sum_{c_j\neq c_i}\mathbf{x}_{ij}\mathbf{x}_{ij}^T - \frac{1}{n_{c_i}-1}\sum_{c_j=c_i}\mathbf{x}_{ij}\mathbf{x}_{ij}^T) \tag{3.3}$$

onde $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$.

Ao invés de maximizar, diretamente, a função objetivo, pode-se usar a solução acima, onde assumimos que a solução aproximada é obtida pela combinação de duas classes, progressivamente, à medida que garantimos que o valor da função objetivo é máximo a cada passo. Neste processo, as duas primeiras restrições (ver pág 17) são naturalmente satisfeitas, e a última pode ser obtida restringindo o número de *features* em cada classe para um valor maior que ou igual a N_s.

Assuma que a matriz de projeção é iniciada como $\mathbf{P}^0 = \mathbf{I}_m \in \mathbb{R}^{m\times m}$, onde $\mathbf{I}_m$ é a matriz identidade de ordem m, isto é, cada *feature* constitui uma classe, inicialmente. No passo $(t+1)$, duas classes são combinadas, isto é, duas colunas da matriz de projeção $\mathbf{P}^t = [\mathbf{p}_1^t, \mathbf{p}_2^t, ..., \mathbf{p}_m^t]$ são somadas gerando $\mathbf{p}_i^{t+1} \Leftarrow \mathbf{p}_i^t + \mathbf{p}_j^t$, e $\mathbf{p}_j^t$ é re-iniciada para $\mathbf{p}_j^{t+1} = 0$. Dessa forma, o valor da função objetivo cresce por:

$$\begin{aligned}(\mathbf{p}_i^t + \mathbf{p}_j^t)^T\mathbf{S}(\mathbf{p}_i^t + \mathbf{p}_j^t) - \mathbf{p}_i^{t^T}\mathbf{S}\mathbf{p}_i^t - \mathbf{p}_j^{t^T}\mathbf{S}\mathbf{p}_j^t \\ = 2\mathbf{e}_i^T(\mathbf{P}^{t^T}\mathbf{S}\mathbf{P}^t)\mathbf{e}_j\end{aligned} \tag{3.4}$$

onde $\mathbf{e}_i$ é um vetor binário de dimensão m com somente uma entrada unitária na posição i.

A análise acima mostra que o máximo da função objetivo é obtido achando o máximo elemento não diagonal da matriz $\mathbf{S}^t = \mathbf{P}^{t^T}\mathbf{S}\mathbf{P}^t$ considerando que o número máximo de *features* em uma classe não passe de N_s. O cálculo computacional da matriz $\mathbf{S}^t$ pode ser feito de forma mais eficiente como:

$$\begin{aligned}\mathbf{S}^{t+1} &= \mathbf{P}^{t+1^T}\mathbf{S}\mathbf{P}^{t+1} \\ &= (\mathbf{P}^t(\mathbf{I} + \mathbf{E}_{ij} - \mathbf{E}_{ii}))^T\mathbf{S}\mathbf{P}^t(\mathbf{I} + \mathbf{E}_{ij} - \mathbf{E}_{ii}) \\ &= (\mathbf{I} + \mathbf{E}_{ij} - \mathbf{E}_{ii})^T\mathbf{S}^t(\mathbf{I} + \mathbf{E}_{ij} - \mathbf{E}_{ii})\end{aligned} \tag{3.5}$$

onde $\mathbf{E}_{ij}$ é uma matriz $m \times m$ binária com somente uma entrada igual a 1 na posição (i,j). Isso simplifica o cálculo da matriz $\mathbf{S}^t$.

Em cada passo, se o elemento (i,i) da matriz $\mathbf{S}^{t+1} = \mathbf{P}^{t+1^T}\mathbf{S}\mathbf{P}^{t+1}$ for um número negativo, os elementos dessa classe são transferidos para a classe C_0 e a classe resultante da combinação é limpa. Se a maior entrada não-diagonal (i,j) da matriz $\mathbf{S}^t$ é negativa ou nula, o algoritmo é terminado. O procedimento detalhado é listado abaixo e na figura 3.2 temos uma ilustração de como são combinadas as *features* nas classes.

1. Inicializar cada feature f_k como uma classe C_k, ou seja, $\mathbf{P}^0 = \mathbf{I}_m$, e iniciar a classe $C_0 = \emptyset$.
2. Calcule a matriz $\mathbf{S}$ como na equação. (3.3).
3. para $t = 1, 2, ..., m - d$,
   - Selecione a maior entrada não diagonal (i,j) da matriz $\mathbf{S}^{t-1} = (\mathbf{P}^{t-1^T}\mathbf{S}\mathbf{P}^{t-1})$ que satisfaça a condição de que o número de elementos não nulos da classe resultante da soma das classes $\mathbf{p}_i$ e $\mathbf{p}_j$ seja menor ou igual a N_s.
   - Se $(\mathbf{p}_i^{t-1} + \mathbf{p}_j^{t-1})^T\mathbf{S}(\mathbf{p}_i^{t-1} + \mathbf{p}_j^{t-1}) \leq 0$, então coloque essas duas classes na classe C_0, ou seja, $\mathbf{p}_i^t = \mathbf{p}_j^t = 0$; ou se a entrada (i,j) da matriz $\mathbf{S}^{t-1}$ não for maior que zero, pare; caso contrário, faça $\mathbf{p}_i^t = \mathbf{p}_i^{t-1} + \mathbf{p}_j^{t-1}$, $\mathbf{p}_j^t = 0$, ou seja, combine as classes C_i e C_j na classe C_i e faça $C_j = \emptyset$;

- $\mathbf{S}^t = \mathbf{P}^{t^T}\mathbf{S}\mathbf{P}^t$

4. Re-ordene os vetores de projeção de acordo com o valor de $\mathbf{p}_i^{t^T}\mathbf{S}\mathbf{p}_i^t$ do maior para o menor. A saída é então $\mathbf{P} = [\mathbf{p}_1,\mathbf{p}_2,...,\mathbf{p}_d]$.

3.1.2 Análise de Complexidade

O custo computacional do algoritmo de *DLBP* pode ser dividido em 2 etapas:

- **Etapa de Aprendizagem** Nessa etapa, o custo computacional é divido em duas partes principais:
  - Custo no cálculo da matriz $\mathbf{S}$, que tem uma complexidade de $O(N^2m^2)\times T_X$ onde $T_\times$ é o tempo gasto para a operação de multiplicação.
  - Custo para o busca da maior entrada não diagonal da matriz $\mathbf{P}^{t^T}\mathbf{S}\mathbf{P}^t$ em todos os $m-d$ passos, que tem uma complexidade de $O(m^3)\times T_+$ onde T_+ é o tempo para a operação de soma (contagem do número de *features*).

  Então, a complexidade total para a etapa de aprendizado é de $O(N^2m^2)\times T_\times + O(m^3)\times T_+$, que é menor que a complexidade do algoritmo de PCA [7], $O(Nm^2)\times T_\times + O(m^3)\times T_\times$, visto que $N \ll m$ e $T_+ \leq T_\times$. Além disso, a complexidade pode ficar ainda menor se restringirmos a distância média na equação. (3.1) para os k vizinhos mais próximos de cada amostra. Nesse caso, o custo computacional para o cálculo da matriz S é de $O(Nkm^2)\times T_\times$. Isso mostra que o algoritmo de *DLBP* é muito eficiente no estágio de aprendizagem.

- **Etapa de Classificação** Nessa etapa, após a obtenção da matriz de projeção, a representação das imagens no espaço de *features* é uma operação de complexidade $O(md\times T_+)$, muito menor que a do algoritmo de PCA e outros algoritmos holísticos [7]. Isso se deve ao fato de a matriz de projeção ser binária e, desta forma, somente operações de soma são necessárias, ao contrário dos demais algoritmos.

O algoritmo de *DLBP* apresenta então diversas vantagens com relação aos demais algoritmos do gênero. É eficiente na classificação e consistente com a forma com a qual os humanos se reconhecem. Além disso, o fato de ter uma matriz de projeção binária faz com que o ao algoritmo tenha baixa complexidade computacional como visto pela análise acima. Isso permite um cálculo rápido das *features*.

3.1.3 Sistema

O algoritmo apresentado acima foi implementado tal como foi descrito. Somente uma pequena mudança foi feita na equação. (3.3) para que a mesma pudesse ser implementada dessa forma. Ocorre que na primeira iteração do algoritmo, $n_{c_i} - 1 = 0, \forall i$, pois só existe uma *feature* em cada classe inicialmente, ou de outra forma, cada *feature* é uma classe no começo do processo. Esse valor nulo não pode ocorrer, e considerando que essa equação só é usada nesse ponto, a equação foi ligeiramente alterada para:

$$\mathbf{S} = \sum_{i=1}^{N} (\frac{1}{N} \sum_{c_j \neq c_i} \mathbf{x}_{ij} \mathbf{x}_{ij}^T - \sum_{c_j = c_i} \mathbf{x}_{ij} \mathbf{x}_{ij}^T) \tag{3.6}$$

A implementação desse algoritmo foi feita na linguagem C, com o auxilio da biblioteca *OpenCV* (*Open Computer Vision Library*) [1]. Essa é uma biblioteca multiplataforma, livre para uso acadêmico e comercial, para o desenvolvimento de aplicativos na área de **Visão Computacional**. Ela foi escolhida por possuir módulos de **Processamento de Imagens e Video I/O, Estrutura de Dados, Álgebra Linear**, centenas de algoritmos de visão computacional como: filtros de imagens, calibração de câmeras, reconhecimento de objetos, etc. O seu processamento de imagens é em tempo real [1]. A escolha pela linguagem C se deve a facilidade em incorporar trechos de códigos prontos em *OpenCV* ao projeto, e ao tempo de treinamento, pequeno se comparado a algumas ferramentas como o Matlab, por exemplo. A implementação do algoritmo foi feita em ambiente **Linux Ubuntu 9.04**, usando o editor **GEDIT** e a compilação usando o compilador da **GNU** (conhecido como **GCC**). Os códigos para a etapa de testes (após o cálculo da matriz de projeção) foram implementados usando o **Matlab 7**. Isso se deve a facilidade que o Matlab proporciona, além do fato de que o tempo deixa de ser uma fator crítico nessa etapa do projeto.

O projeto pode então ser visto, estaticamente, em duas etapas:

1. A etapa onde se busca obter a Matriz de Projeção. Nessa etapa, parâmetros como imagens, o número de iterações e a cardinalidade podem ser variados.

2. A etapa onde se faz o mapeamento das imagens para a classificação. Nessa etapa, é gerada a Matriz de Confusão, ou Tabela de Contingência, que permite

a geração da curva *Receiver Operating Characteristic* (curva ROC) [24], que permite validar os resultados de forma a quantificar o poder discriminativo do classificador.

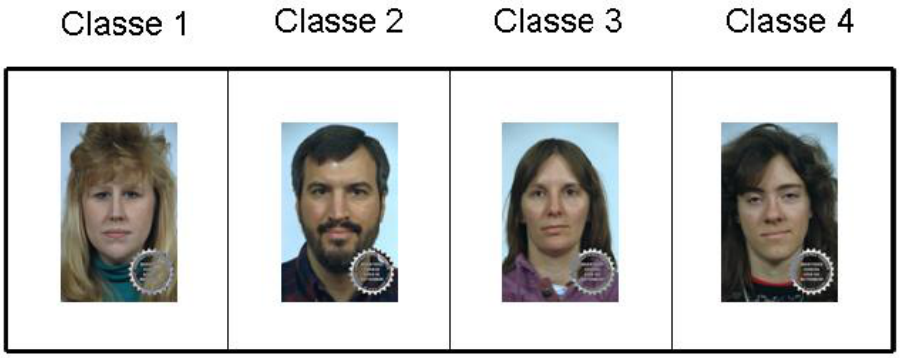

Figura 3.3: Classes no início do algoritmo.

Essas etapas interagem entre si da seguinte forma:

1. Calcular a Matriz de Projeção dado um conjunto de imagens de treinamento, número de iterações e a cardinalidade.
2. Aplicar a Matriz de Projeção obtida às imagens de um conjunto de teste, obtendo assim as *features* no novo subespaço.
3. Classificar as imagens de teste (*features*) usando algum classificador e limiares de classificação, para gerar a curva de ROC. Esses limiares são valores, acima dos quais uma imagem não é considerada de uma pessoa do conjunto treinamento/teste, e por isso não deve ser classificada. As pessoas que não fazem parte do conjunto treinamento/teste, fazem parte do conjunto de validação e são usadas aqui para testar o classificador.
4. Analisar a curva e selecionar o melhor limiar de classificação.
5. Para um novo conjunto de imagens e de parâmetros, refazer as etapas acima, dado o limiar escolhido.

A quinta etapa é realizada para todos os testes propostos, que são apresentados na seção 3.5.

As imagens neste trabalho são dividas em três conjuntos: o de treinamento, usado na primeira etapa e os de teste e validação, usados na segunda etapa.

Questões referentes às imagens usadas na primeira e na segunda etapa, suas origens e seu pré-processamento, serão tratadas respectivamente nas seções 3.3 e 3.4. O parâmetro número de iterações, que aparece na primeira etapa, representa o número de vezes que o algoritmo será treinado. Como já foi dito, haverá $m - d$ iterações, onde m é o número de pixels da imagem e d o número de classes que se quer ao final do treinamento. Como o algoritmo tenta juntar *features* de classes diferentes em uma única classe, em cada interação, o algoritmo precisa de $m - d$ iterações para alcançar d classes. Podemos então variar d e ver como se comporta o algoritmo. Esse teste e também o de variação da cardinalidade, se encontram na seção 3.5.

O classificador usado é o de vizinho mais próximo *Nearest Neighbor*, visto ser esse um dos mais simples classificadores que existe, pois calcula simplesmente a distância euclidiana entre as *features*. Ele também foi o escolhido no artigo [7].

A importância de se traçar a curva de ROC está no fato de que não podemos confiar que a simples quantificação de acertos num grupo de imagens de teste refletirá o quão eficiente esse sistema é, pois essa quantificação dependerá fundamentalmente da qualidade e distribuição dos dados neste grupo de imagens. Para exemplificar, suponha que algum sistema de reconhecimento facial tenha conseguido reconhecer todas as imagens de um conjunto de imagens de teste. Isso, a princípio, parece ser um ótimo resultado, mas se o conjunto for formado também por imagens do conjunto de validação, o sistema não deveria ter reconhecido essas imagens, a menos que o propósito do sistema fosse de reconhecer qualquer indivíduo classificando-o como sendo parecido com alguém de alguma forma. A exceção desse caso, já não teríamos como saber se o sistema é bom ou não. Por isso, outras medidas tiveram de ser criadas, como as medidas de: acurácia, sensibilidade, especificidade, etc. Elas são medidas extraídas da Matriz de Confusão e utilizadas na criação da Curva ROC.

A medida de acurácia é a proporção de predições corretas [25]. Esta medida é altamente suscetível a desbalanceamentos do conjunto de dados e pode induzir a uma conclusão errada sobre o desempenho do sistema. A acurácia, ACC, é dada pela razão entre o Total de Acertos e o Total de Dados no Conjunto, ou seja:

$$ACC = \frac{V_p + V_n}{N_p + N_n}, \tag{3.7}$$

onde V_p e V_n são as quantidades de verdadeiros positivos, pessoas que fazem parte do conjunto treinamento/teste e foram reconhecidas, e verdadeiros negativos, pessoas

que não fazem parte do conjunto anterior e não foram reconhecidas, respectivamente. Assim como N_p e N_n são as quantidades totais de positivos, pessoas reconhecidas, e de negativos, pessoas não reconhecidas, respectivamente.

A medida de sensibilidade fornece a proporção de verdadeiros positivos, isto é, a capacidade do sistema em predizer corretamente a condição para casos que realmente a têm [25]. A sensibilidade, *SENS*, é dada pela razão entre os Acertos Positivos e o Total de Positivos, ou seja:

$$SENS = \frac{V_p}{N_p} = \frac{V_p}{V_p + F_n}, \tag{3.8}$$

onde F_n é a quantidade de falsos negativos, pessoas que deveriam ter sido reconhecidas mas não foram.

A medida de especificidade é a proporção de verdadeiros negativos, isto é, a capacidade do sistema em predizer corretamente a ausência da condição para casos que realmente a não têm [25]. A especificidade,*SPEC*, é dada pela razão entre os Acertos Negativos e o Total de Negativos:

$$SPEC = \frac{V_n}{N_n} = \frac{V_n}{V_n + F_p}, \tag{3.9}$$

onde F_p é a quantidade de falsos positivos, pessoas que não deveriam ter sido reconhecidas mas foram.

Existem ainda as medidas de: Eficiência, que é a média entre sensibilidade e especificidade, medida de Preditividade Positiva, que é a proporção de verdadeiros positivos em relação a todas as predições, medida de Preditividade Negativa, que é a proporção de verdadeiros negativos em relação a todas as predições negativas [25].

Para traçar a curva de ROC, são calculadas antes as matrizes de confusão para cada limiar ou parâmetro do sistema. Por exemplo, variando o parâmetro Cardinalidade (N_s) n vezes, obtemos n matrizes de confusão. De cada matriz, são extraídos os valores de sensibilidade e especificidade. A curva de ROC é a curva da sensibilidade versus o complemento da especificidade ($1 - SPEC$), que considerando o exemplo anterior, terá n pontos [25]. A abscissa da curva de ROC também é conhecida como *False Acceptance Rate* (*FAR*), que é a proporção de falsos positivos. A figuras 3.4 e 3.5 ilustram, respectivamente, a matriz de confusão e a curva de ROC.

Na figura 3.5, a reta vertical e depois horizontal indica um classificador perfeito, já a reta diagonal indica um classificador aleatório, mesma proporção entre

verdadeiros positivos e falsos negativos. Busca-se então um classificador como o da curva acima da reta diagonal, o mais próximo possível das retas horizontal - vertical.

Valor Previsto (predito pelo teste)		Valor Verdadeiro (confirmado por análise)	
		positivos	negativos
	positivos	**VP** Verdadeiro Positivo	**FP** Falso Positivo
	negativos	**FN** Falso Negativo	**VN** Verdadeiro Negativo

Figura 3.4: Matriz de confusão

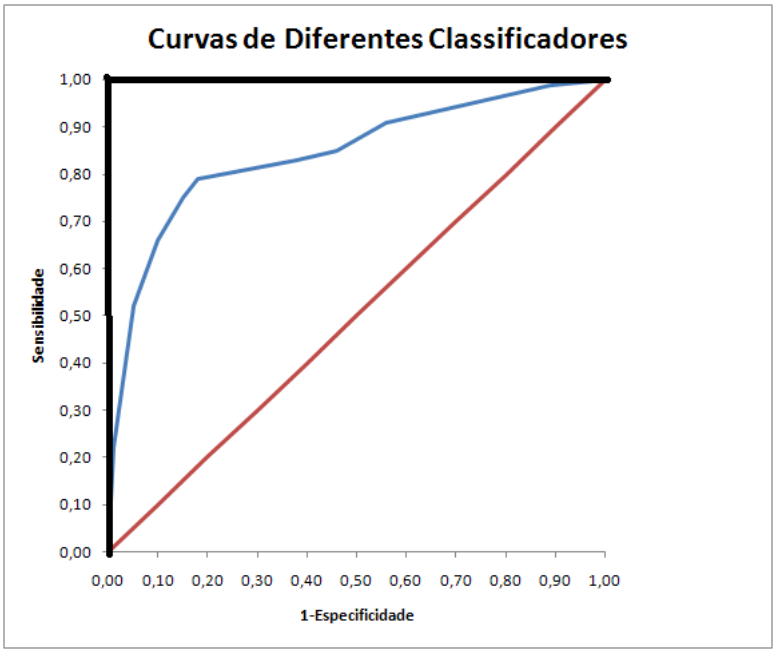

Figura 3.5: Curva ROC.

Uma outra medida muito usada na comparação do desempenho de algoritmos biométricos é a medida de *Equal Error Rate* (*EER*) [26]. Ela fornece a localização na curva de ROC onde a proporção de falsos positivos é igual a proporção de falsos negativos. Quanto mais baixo for o valor de *EER* melhor, o que não significa que seja bom para o sistema funcionar nesse ponto. Isso porque, dependendo da aplicação, pode ser necessário uma taxa de falsos positivos muito mais baixa do que de falsos

negativos ou vice-versa. Por isso que o valor de *EER* só é comumente usado para comparação e não como ponto de funcionamento.

Como foi dito antes, então, o sistema proposto não classifica qualquer pessoa do conjunto de teste. Somente aquelas cujas imagens também apareceram no conjunto de treinamento. Por exemplo, para uma pessoa que tenha doze imagens, podem-se separar seis para treinamento, quatro para teste e duas para validação. Caso essas duas imagens apareçam no conjunto de teste de uma outra pessoa, o sistema não deve reconhecê-las.

A figura 3.3 ilustra o início do algoritmo, onde temos cada imagem em sua própria classe. O mapeamento ocorre como na figura 3.6.

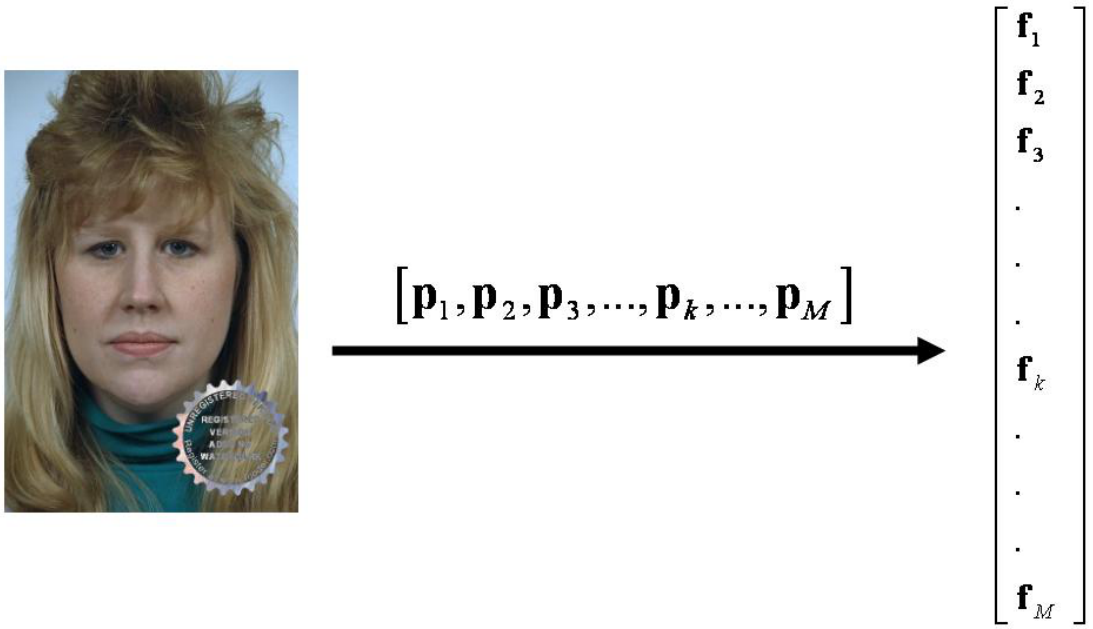

Figura 3.6: Mapeamento das Imagens.

Antes de abordar os assuntos acima, será apresentado na seção 3.2 o algoritmo *CFA*, também usado em reconhecimento facial, cujo desempenho será usado como uma referência na comparação com o do algoritmo *DLBP*.

3.2 Descrição do Algoritmo CFA

A sigla *CFA* significa *Class-Dependence Feature Analysis* e um dos métodos onde esse algoritmo é usado, para reconhecimento facial, se chama *Redundant Class-Dependence Feature Analysis Based on Correlation Filters* [27]. Nesse método, um

banco de filtros de correlação é treinado baseado em um conjunto de imagens de treinamento, que contém várias imagens por classe. Esse banco é então usado para a extração de *features* para o reconhecimento, através do produto interno entre a imagem de teste e os filtros. As *features* então são vetores onde cada componente é obtido pelo produto interno dos vetores pela *DFT* da imagem. O tamanho desse vetor vai depender do número de filtros usados. A comparação das *features* é feita através do ângulo entre elas (*Nearest Neighbor*), para a classificação.

As imagens usadas em [27] foram tiradas da base *Face Recognition Grand Challenge data set* (*FRGC*) [28], uma base de dados grande formada por 12.800 imagens de treinamento, 16.028 imagens para validação, 8.014 imagens não-controladas (onde há variação de iluminação, ruído, embaçamento etc.) e 4.007 imagens 3D.

Muitas técnicas de reconhecimento de faces são no domínio espacial (das imagens), como o *DLBP*, enquanto outras técnicas são aplicadas no domínio da freqüência. Nesse segundo caso, como a informação está no domínio da freqüência, vantagens como tolerância a ruído e a distorções podem ser obtidas facilmente. A técnica *CFA* é uma ferramenta para reconhecimento no domínio da freqüência.

Esse método está dividido em duas etapas, chamadas de fase de inscrição (*enrollment stage*) e fase de verificação (*verification stage*).

Fase de Inscrição Na fase de inscrição, uma ou várias imagens de um mesmo indivíduo são adquiridas. Elas devem refletir as possíveis variações (de rotação, escala e iluminação) que a qualidade de uma imagem de rosto pode sofrer. As transformadas de Fourier dessas imagens de treinamento são usadas por um algoritmo de projeto de filtro para a achar o respectivo filtro de correlação (vetor no domínio da freqüência), desse conjunto de imagens.

Fase de Verificação Nessa fase, é aplicada a transformada de Fourier sobre a imagem de entrada, e está é multiplicada pelos filtros obtidos na fase anterior. A transformada inversa é então calculada sobre cada um desses produtos.

Se os filtros foram corretamente construídos, um pico de correlação pode ser observado caso façamos a correlação de uma imagem com seu respectivo filtro. A figura 3.7 ilustra esse resultado. A posição do pico indica a posição da imagem. Se ela estiver deslocada de alguns *pixels* para a esquerda, o pico de correlação também estará, ou seja, o método é invariante ao deslocamento e isso significa também que não há necessidade de centralizar a imagem de entrada antes de aplicar o método.

A figura 3.8 ilustra o cálculo das features para uma imagem.

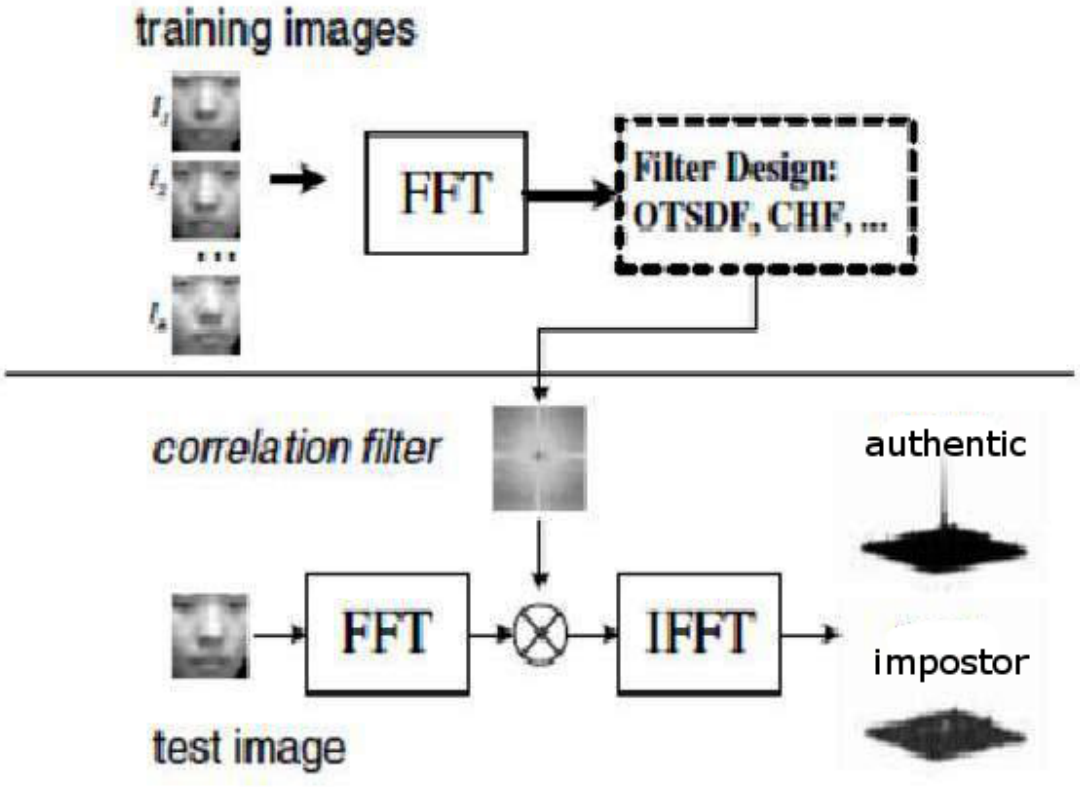

Figura 3.7: Diagrama de blocos do algoritmo de CFA

Uma das formas de se projetar um filtro de correlação é através da otimização de um ou mais critérios da correlação de saída, sob restrições do pico de correlação de saída $c(0,0)$, que é o produto de uma imagem de treinamento e do filtro a ser determinado:

$$c(0,0) = \mathbf{h}^T \mathbf{x}_i$$

onde $\mathbf{h}^T$ é o filtro e $\mathbf{x}_i$ a i-ésima imagem. A idéia em [27] é o projeto de um filtro chamado *optimal tradeoff filter* (OTF), que mistura dois critérios de otimização, o de mínima variação do ruído na correlação da imagem com o filtro ($MVSDF$) e o de mínima média de energia na correlação da imagem com o filtro ($MACE$), e tenta minimiza-los de uma única forma. Então, dado o critério que queremos minimizar $\mathbf{h}^T\mathbf{T}\mathbf{h}$, onde $\mathbf{T} = \alpha\mathbf{D} + \beta\mathbf{C}$ e $0 \leq \alpha, \beta \leq 1$. O OTF é expresso na equação:

$$\mathbf{h}_{OTF} = \mathbf{T}^{-1}\mathbf{X}(\mathbf{X}^T\mathbf{T}^{-1}\mathbf{X})^{-1}c^*$$

onde $\mathbf{D}$ ($MACE$) é o valor médio de $\mathbf{D}_i$, o espectro de potência da i-ésima imagem, e $\mathbf{C}$ ($MVSDF$) é uma matriz diagonal onde seus $C(k,k)$ elementos repre-

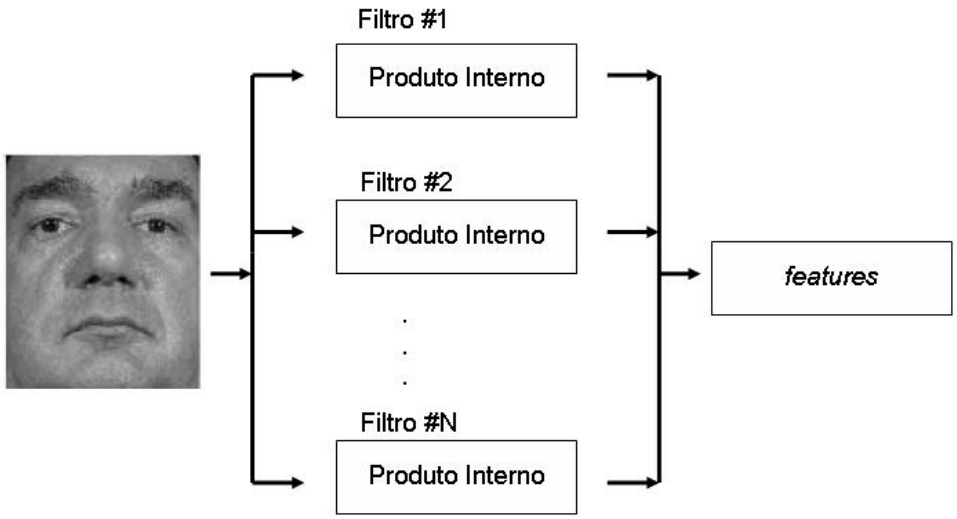

Figura 3.8: Cálculo das features na CFA

sentam a densidade espectral de potência do ruído de correlação na freqüência k, $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N]$ é uma matriz $d \times N$, e cada $\mathbf{x}_i$ é um vetor de dimensão d da transformada de Fourier da i-ésima imagem de treinamento.

3.3 Requisitos

Esse projeto teve os seguintes requisitos:

- Um *laptop* pessoal para escrita dos códigos.
- Computadores do laboratório de processamento de sinais (LPS), usados para simulação dos códigos criados acima.
- A biblioteca gratuita (*OpenCV*), para auxiliar na criação dos códigos.
- O *software* Matlab para a etapa de classificação.
- O sistema operacional Linux Ubuntu, onde o código para a primeira etapa foi criado e compilado.
- Os *softwares* WinEdit e Gimp para a edição do relatório do projeto e manipulação das figuras para o mesmo.

- Bases de dados com fotos de pessoas.
- Uma *webcam* do LPS, Logitech QuickCam Chat, para a etapa de testes.

Para esse projeto foram usadas duas bases de fotos, a **BioID** [29] e a **FRD-ITJRSC-1.7**. A base BioID é uma base disponibilizada pela empresa de segurança digital BioID, composta por 1521 imagens frontais, em tons de cinza, com resolução de 384×286 pixels, de 23 pessoas diferentes. A quantidade de imagens por pessoa é variado, indo de 6 imagens por pessoa a 15 imagens por pessoa. A base FRD-ITJRSC-1.7 é uma base montada na sua maior parte em Manaus, no Instituto de Tecnologia José Rocha Sérgio Cardoso (ITJRSC), e uma pequena parte no Rio de Janeiro. Ela é composta por 4920 imagens coloridas, com resolução de 320x240 pixels, de 45 indivíduos diferentes. Ela foi montada sob diferentes níveis de iluminação e posicionamento do rosto na imagem, apresentando imagens frontais e laterais. Por fim, ela conta com 60 imagens por pessoa e foi adquirida através de uma *webcam* Logitech QuickCam Chat.

Da base BioID, foram usadas 155 imagens, contendo 15 pessoas diferentes, para compor os conjuntos de treinamento, teste e validação. Esse número de imagens se deve ao fato de que só uma pequena parte dessa base estava disponível no começo do projeto, e considerando a necessidade de se ter um número mínimo de imagens por pessoa, somente 155 imagens puderam ser usadas. Já da base FRD-ITJRSC-1.7, foram usadas 500 imagens, contendo 25 pessoas diferentes, nos conjuntos de treinamento, teste e validação. Essa outra base foi usada para verificar o desempenho do sistema para um conjunto maior de indivíduos e de imagens por indivíduo. As figuras 3.9 e 3.10 ilustram imagens extraídas, respectivamente, das bases BioID e FRD-ITJRSC-1.7.

Inicialmente, tentou-se usar a base de dados **FERET** [30], uma das bases mencionada no artigo sobre o *DLBP*. Entretanto, devido à dificuldade em se extrair dessa base uma quantidade grande de imagens frontais, devido à sua organização e devido à marca d'água presente em todas as suas imagens, o que poderia dificultar a detecção da face, optou-se então pelas outras duas bases citadas anteriormente.

3.4 Preprocessamento das Imagens

Para que as imagens das bases pudessem ser usadas em cada etapa desse projeto, elas tiveram que passar por um pré-processamento que envolveu as seguintes

Figura 3.9: Exemplo de imagem da base BioID.

Figura 3.10: Exemplo de imagem da base FRD-ITJRSC-1.7

etapas:

1. Etapa de Alinhamento: onde a imagem é alinhada (normalizada) com relação a duas coordenadas, tipicamente as do centro dos olhos. O processo foi feito de forma automática através de um *script* Linux. O *script* recebe um arquivo texto contendo, em uma certa formatação, o nome das imagens e as coordenadas aproximadas dos olhos. As imagens resultantes têm dimensão de 130×150 *pixels*.

2. Etapa de Escalamento: onde a imagem é reduzida, em suas duas dimensões, de um fator que depende do teste que está sendo realizado. Esse fator foi de 3 (43×50 *pixels*) e 5 (26×30 *pixels*).

Com relação aos fatores de escalamento, foram aplicados valores de 3 e 5, onde se teve como objetivo observar se seria possível trabalhar com imagens menores e

assim diminuir o tempo de processamento sem impactar muito nos resultados. Por exemplo, para imagens de dimensão original 320×240 *pixels*, e que após a etapa de alinhamento passam a ter dimensão 130×150 pixels, para um fator de escala de 5 geram imagens de dimensão 26×30 *pixels*.

A figura 3.11 ilustra uma imagem do banco FRD-ITJRSC-1.7. Já a figura 3.12 ilustra essa mesma imagem alinhada.

Figura 3.11: Imagem original do banco FRD-ITJRSC-1.7

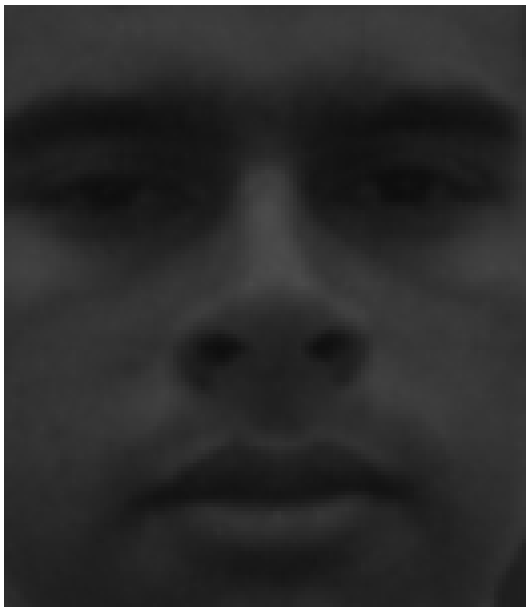

Figura 3.12: Ilustrando o alinhamento da imagem mostrada na figura 3.11

3.5 Testes Realizados

Neste projeto, foi realizada uma série de testes para avaliar o desempenho do algoritmo de *DLBP* quanto aos seus parâmetros (número de classes d e cardinalidade N) e quanto à sua robustez a quantidade de classes (pessoas) e imagens por classe. Também foram incluídas imagens extras, para formar o grupo de avaliação, em cada teste. Como as imagens da base FRD-ITJRSC-1.7 variam quanto às características de iluminação e translação, e as da BioID também, a capacidade em reconhecer, nessas condições, também foi testada. Foram então realizados quatro testes no total, repetidos 2 vezes. Essa repetição se deve ao escalamento, já citado, realizado sobre todas as imagens usadas. Os testes, suas descrições e demais detalhes serão apresentados a seguir.

Teste de Validação Cruzada Nesse teste, temos o cruzamento das imagens de treinamento e teste em sete grupos diferentes, originados de um mesmo conjunto inicial de imagens. Isto é, a partir de um conjunto inicial de vinte pessoas, com dez imagens por pessoa, foram selecionados sete grupos de subconjuntos contendo sete e três imagens para cada pessoa, para compor os conjuntos de treinamento e teste, respectivamente. Os sete grupos foram criados respeitando a seguinte dinâmica:

1. Selecionar quaisquer sete imagens, em seqüência, das dez disponíveis para cada pessoa, e compor o primeiro grupo de treinamento. Selecionar, por conseguinte, as três imagens restantes para o grupo de teste. Por exemplo, para uma pessoa do grupo $G_{treinamento} = [I_1,I_2,I_3,I_4,I_5,I_6,I_7]$ e $G_{teste} = [I_8, I_9, I_{10}]$.
2. Para o próximo grupo, trocar três imagens do primeiro grupo por três novas imagens, tanto para treinamento quanto para teste. Essas três novas imagens também estão em seqüencia. Seguindo o exemplo, $G_{treinamento} = [I_4,I_5,I_6,I_7,I_8,I_9,I_{10}]$ e $G_{teste} = [I_1, I_2, I_3]$. A figura 3.13 ilustra a forma como foi organizada as imagens para esse teste.

Esta é uma variação de um teste tipicamente usado em métodos de predição, onde o objetivo é a seleção dos melhores conjuntos de parâmetros para uma predição "ótima". Esses testes de validação cruzada são chamados de *K-folds* [31]. Eles também são usados para mostrar o quanto os resultados de um

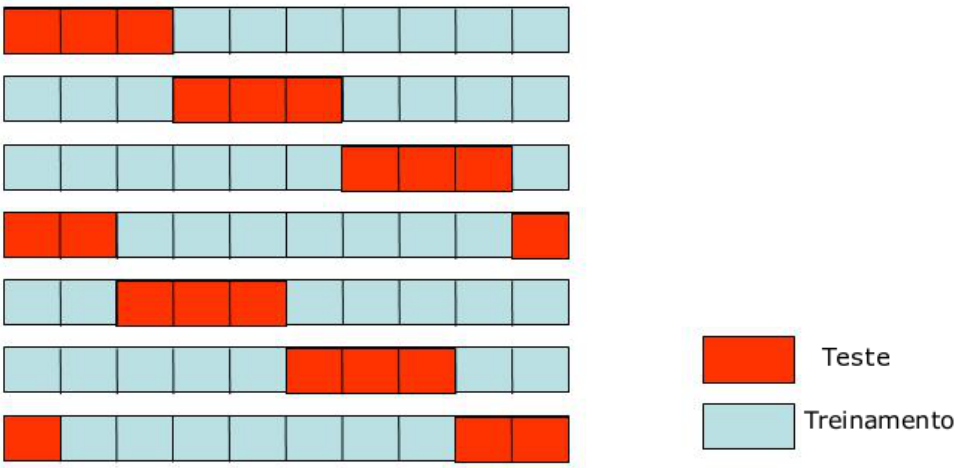

Figura 3.13: Organização das imagens para o teste de validação cruzada.

método são estatisticamente independentes dos seus dados de entrada, e assim avaliar melhor a capacidade de generalização do método. Assim, se espera que a acurácia não mude muito de um conjunto de dados para o outro. O valor de cardinalidade dos vetores de projeção usados aqui foi de $N_s = 5$.

Teste de Variação de Cardinalidade Neste teste, o treinamento é realizado para um número diferente de cardinalidades dos vetores de projeção. O valor de N_s é variado de: $N_s = [5, 10, 20, 25, 30, 40, 50, 100, 140, 180]$. O número de imagens nesse teste foi de 175 para o treinamento, 25 pessoas com 7 imagens por pessoa, e 75 para teste, 25 pessoas com 3 imagens por pessoa. A escolha dos valores de cardinalidade se devem ao valor inicial de 5, usado em [7], e ao interesse em observar o efeito para valores maiores mas com crescimento gradual, até o valor de 180, também mencionado em [7].

Teste de Variação de Classes Nesse teste, mantendo o valor de $N_s = 5$, variamos o número de pessoas de $C = [1, 3, 5, 8, 12, 15, 20]$, para a base FRD-ITJRSC-1.7 e $C = [1, 3, 5, 8, 10, 12]$ para a base BioID. O número de imagens por pessoa para treinamento e teste foi de 7 e 3, respectivamente. O objetivo desse teste é saber como o algoritmo se comporta desde um número mínimo de pessoas diferentes, até um número maior, com crescimento gradual. Para manter um padrão nos testes que não envolvem a variação do parâmetro N_s, resolveu-se fixar essa variável em 5.

Teste de Variação de Imagens por Classe Nesse teste, para $N_s = 5$, variamos o número de imagens por pessoa em $IC = [1, 3, 4, 5, 6, 8, 10, 15]$ para a base FRD-ITJRSC-1.7 e $IC = [1, 3, 4, 5, 6, 8]$ para a base BioID. O número de pessoas é fixo em 25 para a base FRD-ITJRSC-1.7 e 13 para a base BioID, tanto para treinamento quanto para teste. O objetivo desse teste é saber se o algoritmo responderá bem para um número muito pequeno de imagens por pessoa e se aumentando muito esse valor, a resposta será igualmente melhor.

No início desta seção, foi mencionado um teste sobre o parâmetro d. Esse parâmetro controla o número de classes que se quer ao final da etapa de treinamento. Como já foi explicado, na subseção 3.1.3, para termos d classes é preciso $m - d$ iterações do algoritmo de treinamento. O número de classes corresponde ao número de colunas da matriz de projeção, ou seja, as bases ou vetores de projeção da matriz. O objetivo é ter o menor valor possível de d, mas que ainda assim proporcione um bom resultado para o reconhecimento. Então, testes foram realizados com valores de $m - d$ em torno de 1800 a 3600, isto porque o valor de $m \cong 2160$ para um escalamento de 3, e por isso, tentou-se valores em torno desse m. Para um valor de escalamento maior, o valor de m é menor, e cai na razão $\frac{m}{s^2}$, onde s é o valor do escalamento e m o número de *pixels* da imagem. Entretanto, foi observado que mantendo um número de iterações alto ($m - d = 3000$), independente do valor do escalamento, o algoritmo de treinamento ainda funcionava. Isso porque o algoritmo funciona juntando classes a cada iteração, ou melhor, somando colunas da matriz de projeção em cada iteração, mas ele também encerra caso ocorra a condição citada na subseção 3.1.1. Essa situação de encerramento ocorre a uma taxa inversamente proporcional ao valor do escalamento, ou seja, quanto maior o escalamento usado, mais rápido o algoritmo se encerra. Esse é o comportamento esperado e equivale a se ter usado um valor menor de m. Por isso, foi usado um valor fixo de $m-d = 3000$ para todos os testes já que, para valores maiores que esse, o algoritmo continuava encerrando na mesma iteração, e nenhuma mudança de desempenho era observada.

A justificativa acima pode ser vista de outra forma, dado que a iteração que gera a matriz de projeção só é influenciada pelos parâmetros $m-d$ quanto ao número de vezes que ela irá ocorrer. Isso significa que $m - d$ não é usado dentro da iteração e, considerando o processo de união de classes mencionado antes, poderíamos, por exemplo, executar $m - d = 20$ iterações diretamente, ou executar $m - d = 10$, e depois, usando os vetores obtidos nas iterações anteriores, executar mais uma vez

$m - d = 10$ para completar vinte interações, e chegaríamos ao mesmo resultado. Por fim, considerando o critério de parada, podemos concluir que deixar um valor alto de $m - d$ não resulta em erros pois ele irá gerar o número máximo de bases que a cardinalidade e o conjunto de imagens usado permitem, sendo essas bases as mesmas, salvo a quantidade, independente do valor de $m - d$ usado.

Com relação aos testes realizados em [7], a principal diferença com os testes realizados aqui, está no número de imagens usadas nas etapas de representação (geração das bases) e reconhecimento (treinamento e teste). Neste artigo, foram usadas duas bases de dados na etapa de representação: XM2VTS (base com 295 pessoas com 4 imagens frontais por pessoa), CMU PIE (com 68 pessoas com 9 imagens por pessoa). Essa grande quantidade de imagens garantiu uma boa representação das bases, isto é, bases esparsas ressaltando partes do rosto, como boca e nariz. Já na etapa de reconhecimento, o artigo usa as bases: FERET (com 70 pessoas 6 imagens para treinamento e teste) e CMU PIE (com 68 pessoas com 9 imagens para treinamento e 12 para teste). Para esta etapa, o número de imagens foi menor e conseqüentemente a representatividade das bases foi menor, o que ainda assim permitiu bons resultados de taxa de reconhecimento. Esse resultado também é evidenciado nas comparações com outros algoritmos, onde o *DLBP* foi melhor, alcançando taxas superiores.

Com relação a testes comparativos, aqui também foram realizados testes entre os algoritmos de *DLBP* e *CFA*. A comparação foi feita aplicando no método *CFA* os mesmos testes feitos para o algoritmo *DLBP*. Tudo pode ser feito da mesma forma, visto que o algoritmo *CFA* também gera, após o treinamento, uma matriz usada para obter as *features* das imagens de teste. A representatividade das bases e a acurácia serão apresentados na seção 3.6 a seguir.

3.6 Resultados

Os resultados serão apresentados seguindo uma ordem. Primeiro são apresentados os resultados de representatividade das bases e depois são apresentados os gráficos de acurácia.

A figura 3.14 mostra as 12 primeiras bases, ou vetores de projeção, geradas pelo algoritmo de *DLBP* para a primeira matriz gerada pelo teste de validação cruzada na FRD-ITJRSC-1.7. As bases, em geral, apresentam esta mesma aparência

para todos os testes, independentemente da escala usada.

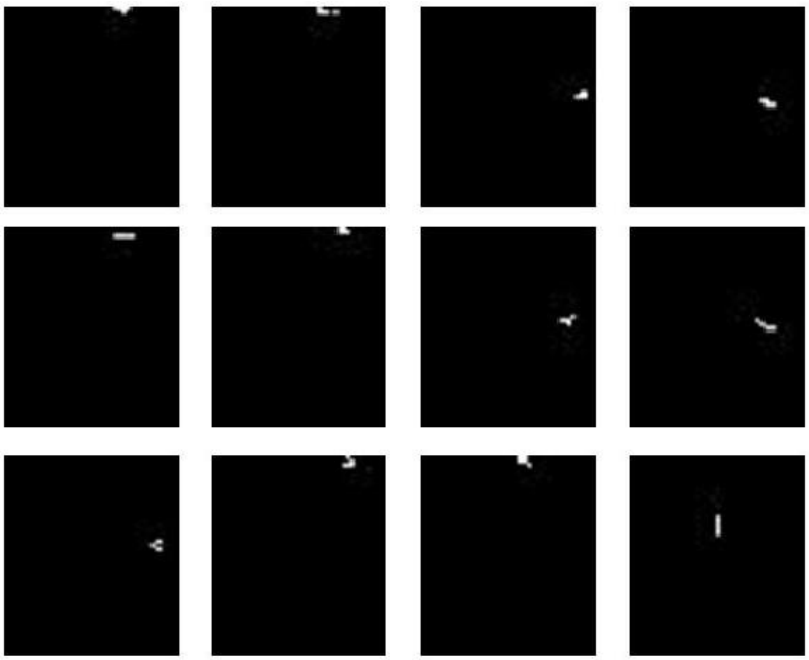

Figura 3.14: As 12 primeiras bases do *DLBP* para $Ns = 5$.

Em seguida, a figura 3.15 mostra 12 filtros gerados pelo algoritmo *CFA* para o mesmo teste na mesma base acima mencionados.

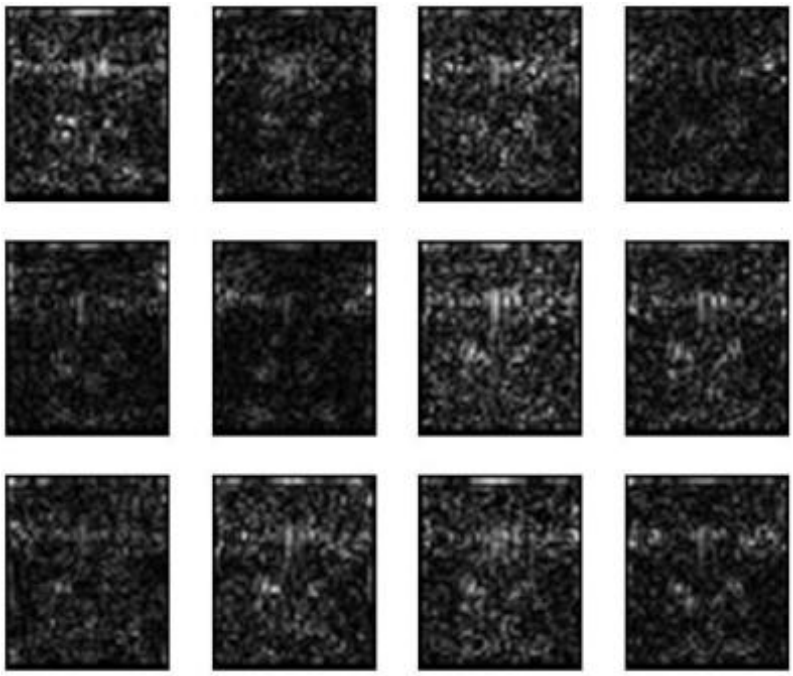

Figura 3.15: A magnitude dos 12 primeiros filtros do *CFA*.

Estas figuras servem para demonstrar a quantidade de informação que cada

uma dessas bases carrega. O algoritmo *DLBP* carrega muito menos informação pois as bases são bem esparsas, além de serem binárias. Por outro lado, a pouca informação, espalhada pelas bases, não traz informação de face, diferentemente no caso do *CFA*, onde podemos observar traços de faces nos filtros. Essa última observação se deve ao fato de que como estamos trabalhando com faces, podemos esperar ver algum padrão nas imagens dos vetores, já que as faces apresentam padrões como testa, nariz, boca etc.

A seguir, na figura 3.16, temos as 4 primeiras bases do algoritmo *DLBP* para valores de cardinalidade do vetores de projeção (Ns) entre 5 e 140, referentes ao teste onde se varia Ns, citado na seção 3.5.

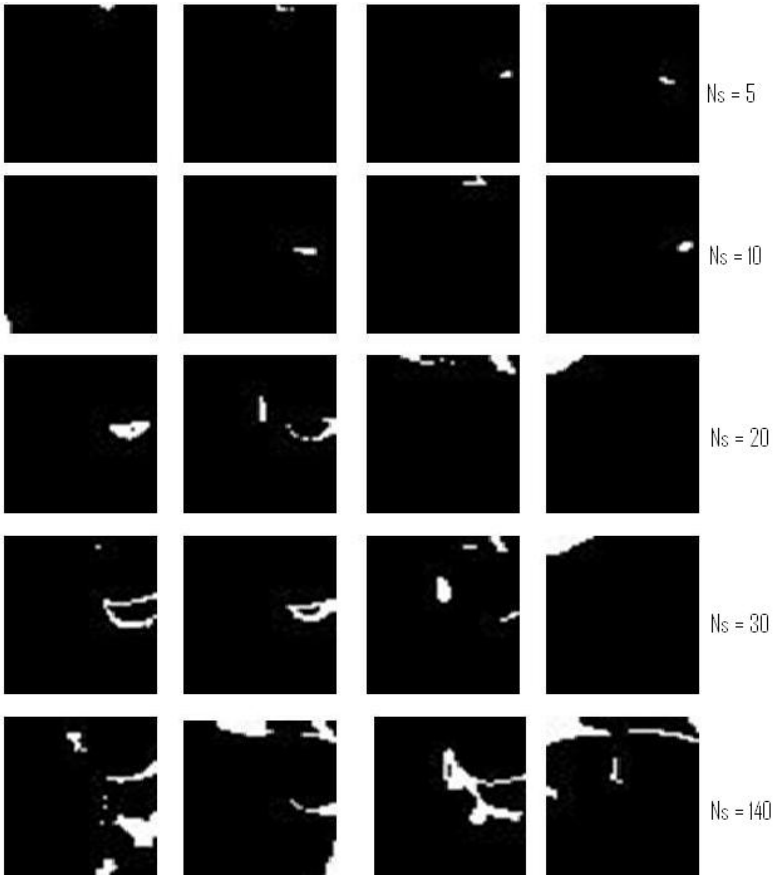

Figura 3.16: As 4 primeiras bases do *DLBP* para Ns entre 5 e 140.

Pela figura 3.16, podemos concluir que quanto maior for o valor de Ns, mais representativas são as bases, já que maior é a cardinalidade. Isso chega ao ponto de aparecer traços faciais nas bases dos últimos valores de Ns. Além disso, quanto maior for Ns, menor é a quantidade de bases geradas, começando com 430 bases para $Ns = 5$ e chegando a 12 para $Ns = 180$. O que demonstra, mais uma vez, que

a informação vai se concentrando a medida que Ns aumenta.

Para concluir os resultados de representatividade, segue os mapas de importância do algoritmo *DLBP*, na figura 3.17, para Ns igual a 5, 10 e 20. Um mapa de importância é formado pela soma de uma certa quantidade de bases. Na primeira linha temos 3 mapas para as bases de 1 a 60, 121 a 180 e 1 a 240 respectivamente. Para os demais valores de Ns, as bases também foram divididas em 3 grupos. Essas são as bases geradas do teste mencionado na figura 3.16.

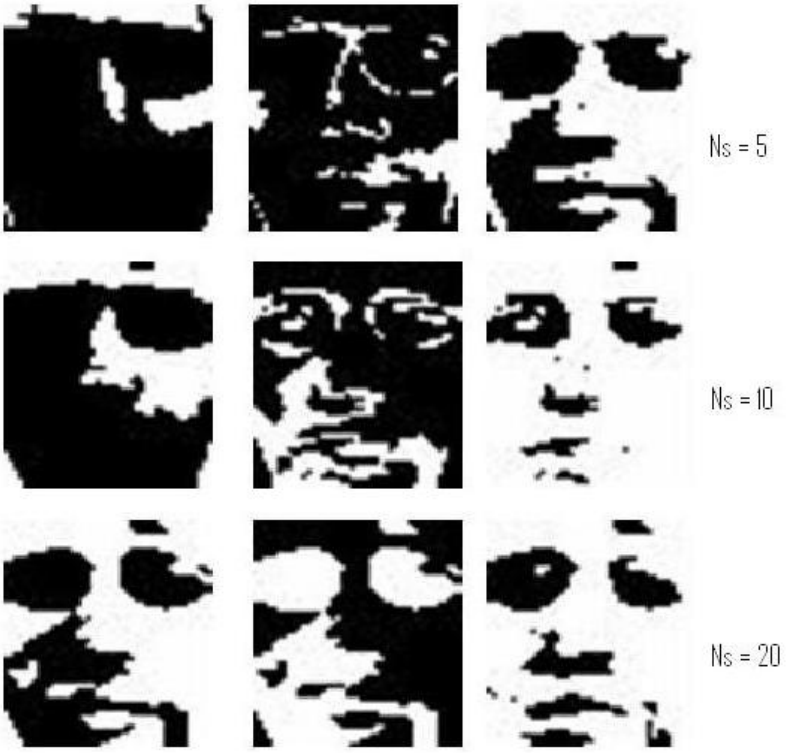

Figura 3.17: Três mapas de importância para Ns igual 5, 10 e 20.

Pela figura 3.17, podemos concluir que as informações de menor variação, como a testa e bochecha, ficam concentrados nas primeiras bases, enquanto as bases seguintes ficam encarregadas de pequenos detalhes e entornos. Mais a frente será mostrado que as primeiras bases são as mais relevantes para o reconhecimento.

Terminados os resultados de representatividade, seguem os resultados de acurácia para o algoritmo *DLBP*, nas bases FRD-ITJRSC-1.7 e BioID, e para o algoritmo *CFA* na base FRD-ITJRSC-1.7, comparativamente aos resultados do *DLBP* na mesma base. Para a FRD-ITJRSC-1.7, os resultados apresentados serão tanto para escala 3 (imagens de 43×50 *pixels*) como para escala 5 (imagens de 26×30 *pixels*). Foram usadas 15 imagens de 5 pessoas, 3 por pessoa, para compor o grupo

de validação na FRD-ITJRSC-1.7 e na BioID.

Antes de apresentar os resultados para cada um dos testes propostos, são apresentados os resultados de um teste para avaliar quantas e quais bases *DLBP* devem ser usadas ao longo dos testes para se obter a maior acurácia possível. O resultado foi obtido para $Ns = 5$ e escala 3. Com esta cardinalidade são geradas 430 bases. Além disso, também para esse teste, serão apresentadas as taxas de falsos positivos e negativos obtidas.

Na figura 3.18, está sendo avaliada a acurácia para diferentes quantidades de bases. Assim, pelo gráfico, temos um pico de abscissa 86. Isso significa que, usando as primeiras 86 bases geradas para $Ns = 5$ e para esse conjunto de faces, obtemos a melhor acurácia. Quanto à figura 3.19, a escolha anterior também parece correta, pois gera as mais baixas taxas de falsos positivos e negativos.

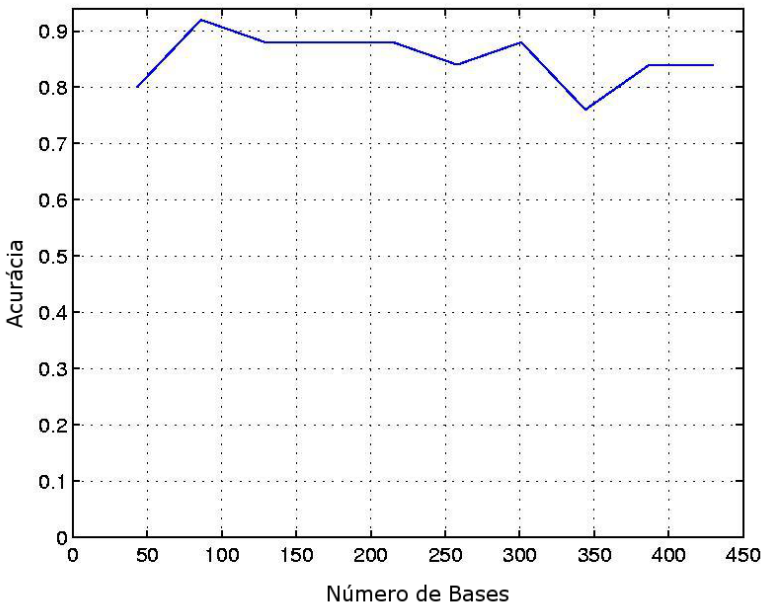

Figura 3.18: Acurácia para diferentes quantidades de bases na FRD-ITJRSC-1.7 para o algoritmo *DLBP*.

Já na figura 3.20, se avalia a faixa de bases que deve ser usada. As 430 bases foram dividias em 7 faixas com igual quantidade de bases em cada uma delas. Pelo gráfico, observamos que a primeira e a segunda faixa apresentam os melhores resultados. Elas correspondem as bases de 1 a 60 e de 61 a 120, respectivamente. Olhando para a figura 3.21, a segunda faixa parece ser a de melhor compromisso entre taxa de falsos positivos e taxa de falsos negativos. Por esses 4 gráficos, decidiu-se por usar as primeiras 86 bases para os demais testes, a menos dos testes onde se

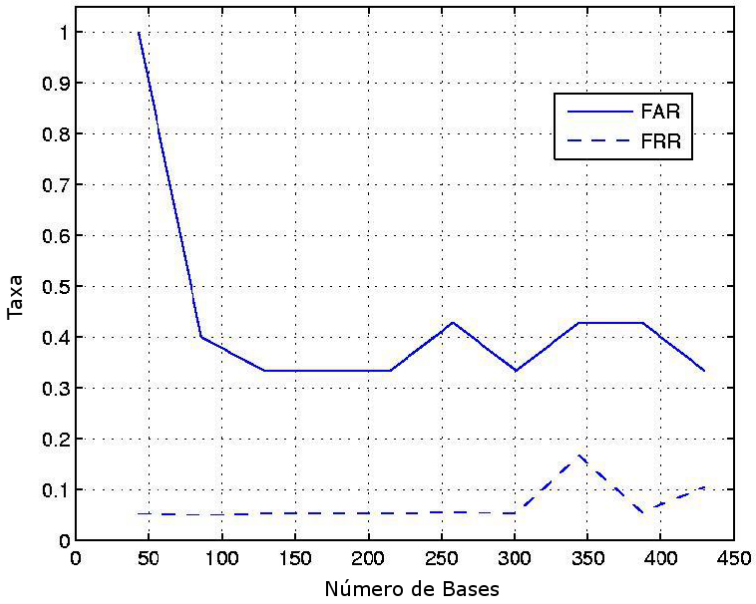

Figura 3.19: Taxa de falsos positivos e negativos para diferentes quantidades de bases na FRD-ITJRSC-1.7 para o algoritmo *DLBP*.

varia a cardinalidade.

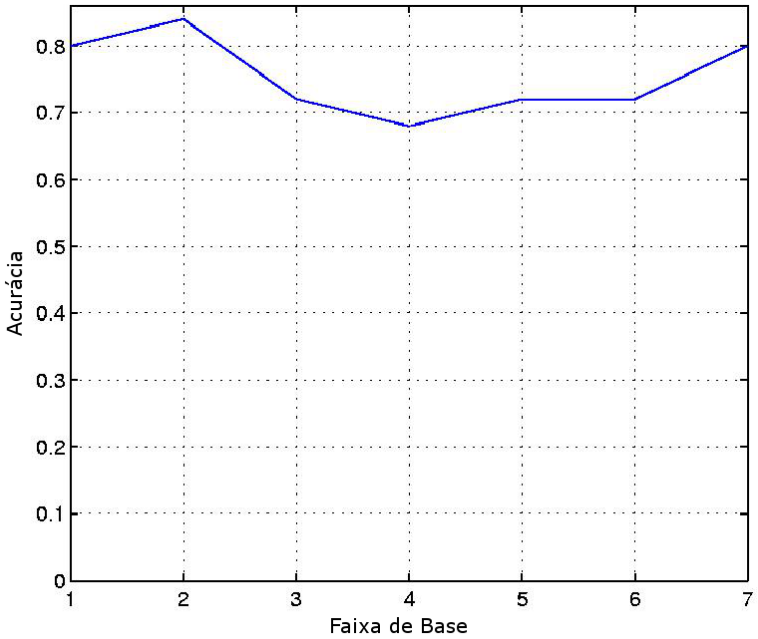

Figura 3.20: Acurácia para diferentes faixas de bases na FRD-ITJRSC-1.7 para o algoritmo *DLBP*.

A seguir serão apresentados os resultados para o teste de validação cruzada, na base FRD-ITJRSC-1.7, para os algoritmos *DLBP* e *CFA*. Nesse teste, como já

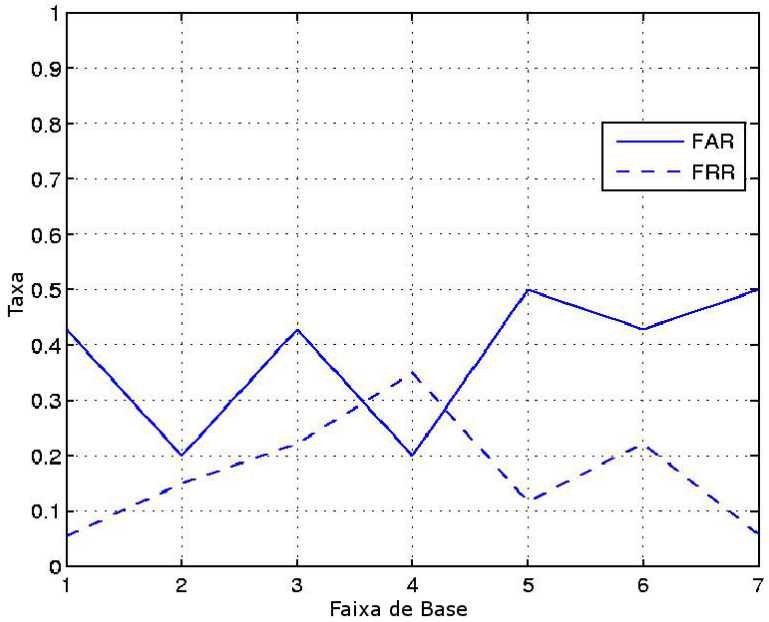

Figura 3.21: Taxa de falsos positivos e negativos para diferentes faixas de bases na FRD-ITJRSC-1.7 para o algoritmo *DLBP*.

mencionado, espera-se que o valor de acurácia não varie muito com as amostras de faces usadas. Para esse teste foram calculados os intervalos de confiança para ambos os algoritmos e em ambas as escalas. Esses intervalos foram calculados através da equação: $intervalo = \mu \pm 2.447 \times \sqrt{\frac{\sigma^2}{N}}$, considerando uma distribuição t - *student* visto que $N < 30$ [32]. μ e σ^2 são a média e a variância das medidas e os seus valores são estimados usando $m = \frac{\sum_i x_i}{N}$ e $s^2 = \frac{\sum_i (x_i - m)^2}{N-1}$, respectivamente.

A figura 3.22, mostra que o algoritmo *DLBP* não só obteve uma acurácia maior para o conjunto de amostras citado anteriormente, como também não variou muito ao longo das mesmas.

Na figura 3.23 é apresentado o resultado comparativo entre os algoritmos *DLBP* e *CFA*, tal como antes, só que na escala 5 (imagens de 26×30 *pixels*). Os resultados são semelhantes, com uma acurácia um pouco menor em alguns pontos. Mais uma vez o algoritmo de *DLBP* apresentou resultados melhores quanto a acurácia para o mesmo conjunto de amostras.

Agora os resultados para o teste de variação de cardinalidade, para a base FRD-ITJRSC-1.7, para os dois algoritmos e em todas as escalas. Para o caso do algoritmo de *CFA*, foi variada a quantidade de filtros usados. Espera-se que o resultado melhore de forma proporcional a quantidade de filtros usados. Para o algoritmo

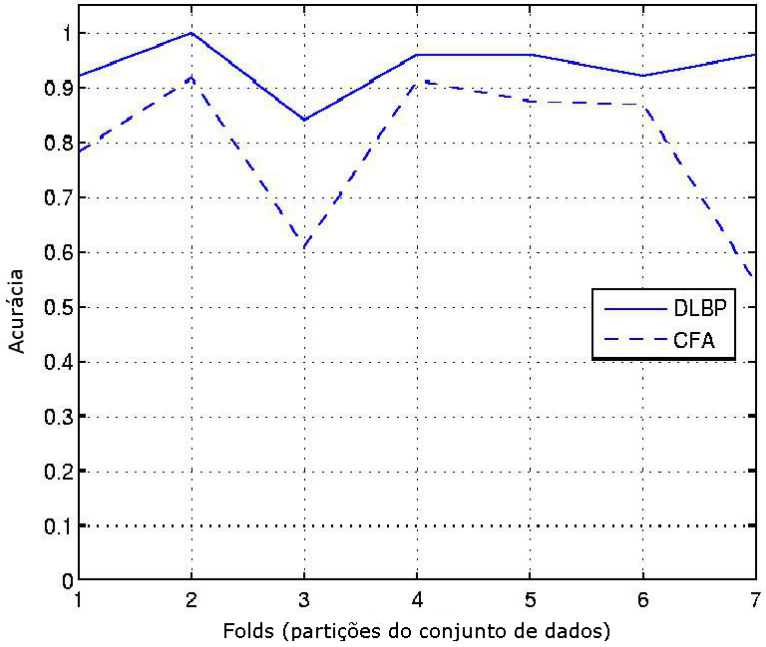

Figura 3.22: Acurácia no teste de validação cruzada para FRD-ITJRSC-1.7 escala 3. Comparando *DLBP* e *CFA*.

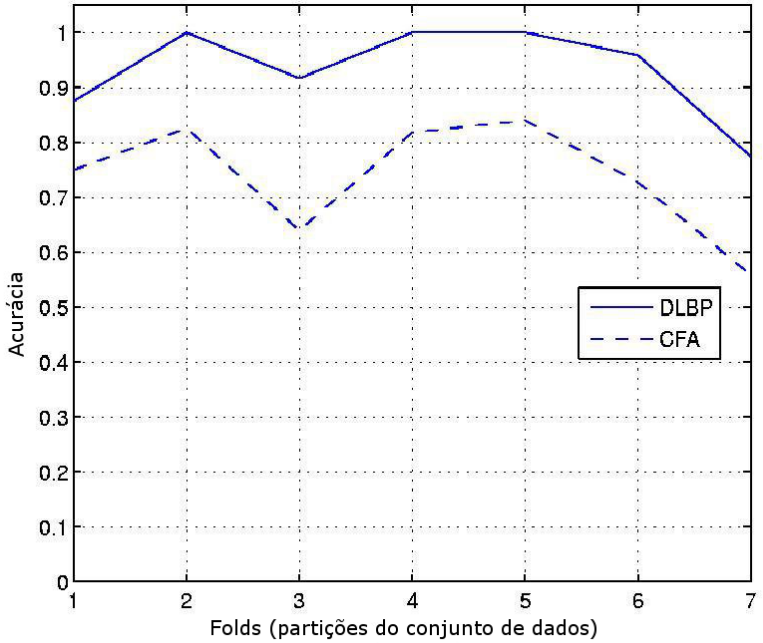

Figura 3.23: Acurácia no teste de validação cruzada para FRD-ITJRSC-1.7 escala 5. Comparando *DLBP* e *CFA*.

de *DLBP*, usou-se o primeiro quinto das bases disponíveis para cada cardinalidade.

Na figura 3.24, são mostrados os gráficos de variação de cardinalidade nas duas escalas. Há uma piora, não esperada, para a escala 5, nos valores de cardina-

Tabela 3.1: Intervalos de confiança do teste de validação cruzada.

Gráficos	**Intervalos de Confiança**
DLBP escala 3	$0{,}95 \pm 0{,}04$
CFA escala 3	$0{,}8 \pm 0{,}1$
DLBP escala 5	$0{,}92 \pm 0{,}07$
CFA escala 5	$0{,}73 \pm 0{,}09$

lidade mais altos.

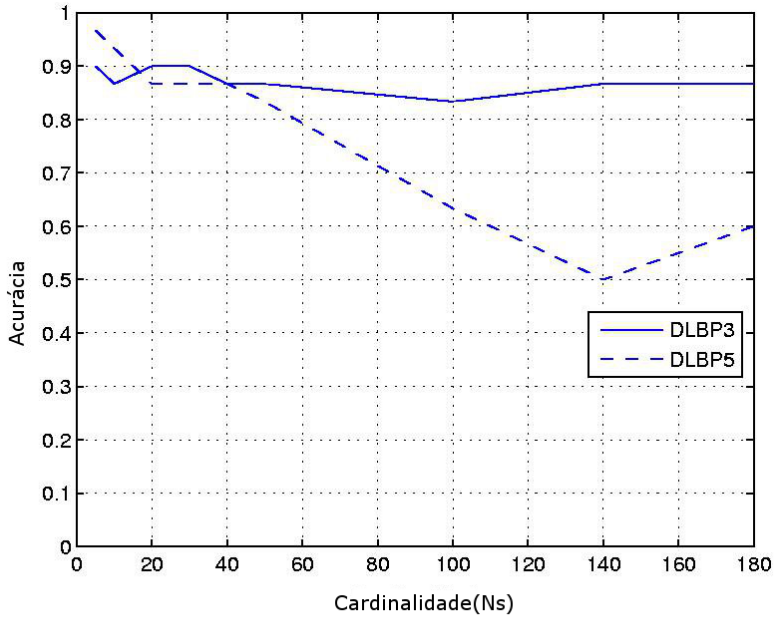

Figura 3.24: Acurácia no teste de variação de cardinalidade na FRD-ITJRSC-1.7 para *DLBP* escala 3 e 5.

Na figura 3.25, são mostrados gráficos semelhantes aos anteriores, só que agora sobre o *CFA*, variando a quantidade de filtros usados.

Por esse gráficos, concluímos que o algoritmo *DLBP* continua com resultados superiores aos obtidos pelo algoritmo *CFA*. O *CFA*, como esperado, apresenta resultados melhores para o uso de todos os filtros, não apresentando, contudo, bons resultados para a escala 5, nas altas cardinalidades.

Em seguida, seguem os resultados para o teste onde se varia a quantidade de pessoas, ou classes, usadas da base FRD-ITJRSC-1.7.

A figura 3.26 e a 3.27 demonstram que a acurácia aumenta, para o referido conjunto de imagens, na medida em que se aumenta a quantidade de classes. Além

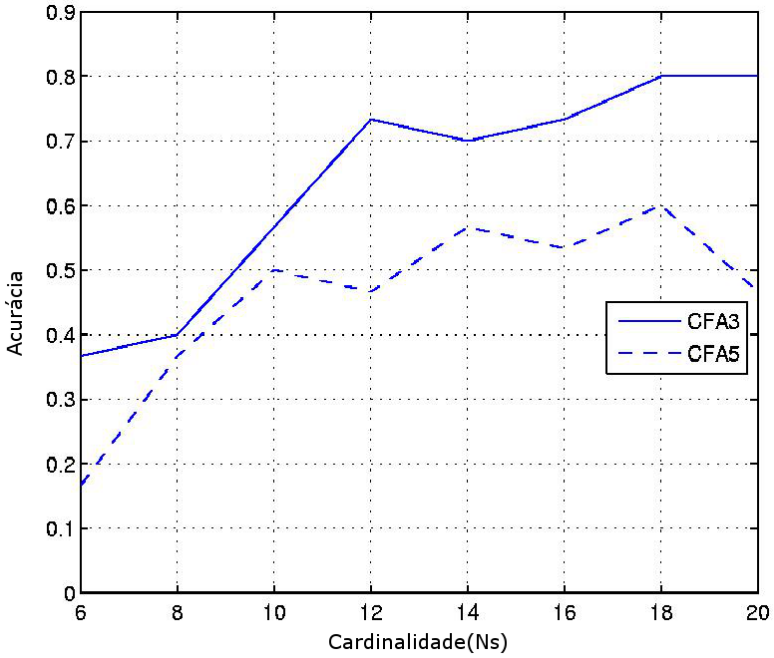

Figura 3.25: Acurácia no teste de variação de cardinalidade na FRD-ITJRSC-1.7 para *CFA* escala 3 e 5.

disso, em ambos os casos, o algoritmo *DLBP* resultou em uma acurácia maior que a do algoritmo *CFA*. Na figura 3.26, observamos uma acurácia de 1 para uma classe. Uma possível explicação para esses resultados se deve ao valor de limiar de classificação usado, que é diferente para cada ponto da curva, e que, no caso desses dois pontos, foi capaz de separar bem as classes.

Por fim, seguem os resultados para o teste onde se varia a quantidade de imagens por classe.

Pelos gráficos das figuras 3.28 e 3.29, observamos que tanto o *DLBP* quanto o *CFA* apresentam resultados semelhantes para ambas as escalas. O *DLBP* continua resultando em gráficos com acurácia superior aos do *CFA*.

Após a apresentação dos resultados para a base FRD-ITJRSC-1.7, serão apresentados os resultados para a base BioID. A ordem com a qual os resultados serão mostrados é a mesma para o caso anterior. Como já foi dito, os resultados foram gerados somente para o algoritmo *DLBP*, pois o interesse está em avaliar o desempenho do *DLBP* em outra base.

Como antes, os dois primeiros gráficos são usados para se saber quais bases devem ser usadas, para o caso de $Ns = 5$. Por eles concluímos, e principalmente pelo gráfico 3.31, que devem ser usadas quase que metade do número total de bases

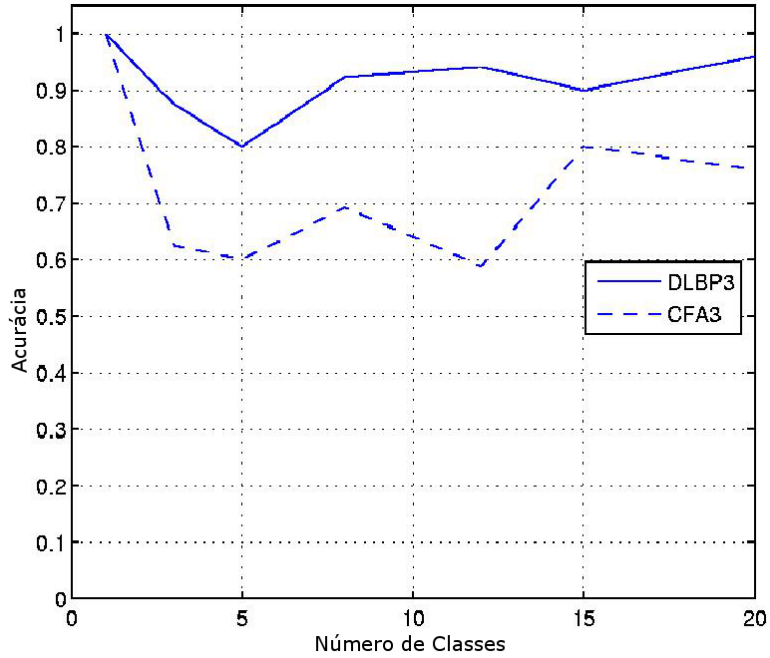

Figura 3.26: Acurácia no teste de variação do número de classes na FRD-ITJRSC-1.7 escala 3.

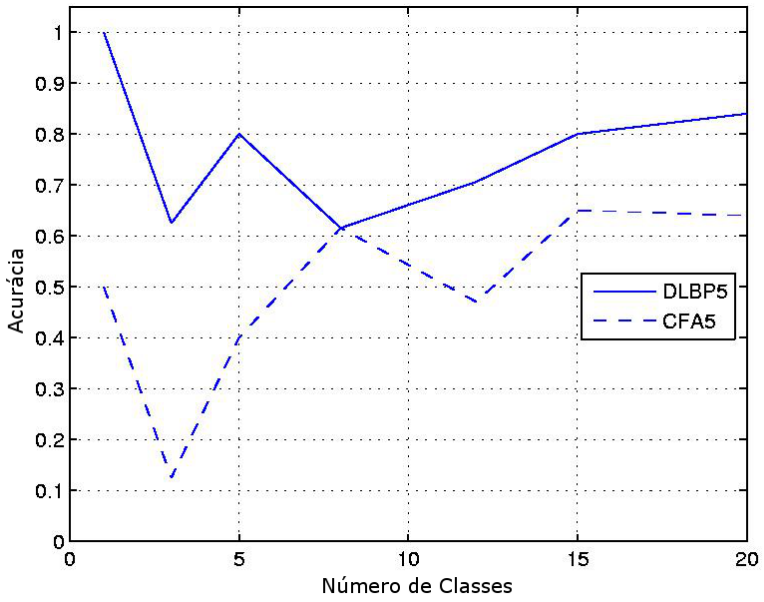

Figura 3.27: Acurácia no teste de variação do número de classes na FRD-ITJRSC-1.7 escala 5.

geradas. Como antes, foram geradas 430 bases (caso onde $Ns = 5$ e a $escala = 3$), e por isso resolveu-se usar 172 bases (2/5 do total).

Seguem os demais resultados para os 4 testes aplicados. No caso do teste de validação cruzada para a base BioID, o intervalo de confiança foi de $0{,}5 \pm 0{,}1$.

Todos os resultados apresentados pelo algoritmo *DLBP*, para a base BioID,

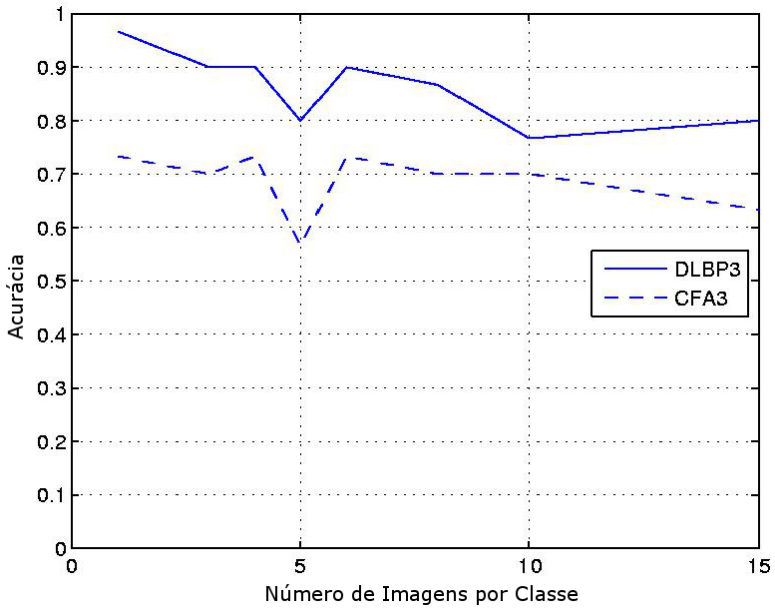

Figura 3.28: Acurácia no teste de variação do número de imagens por classe na FRD-ITJRSC-1.7 escala 3.

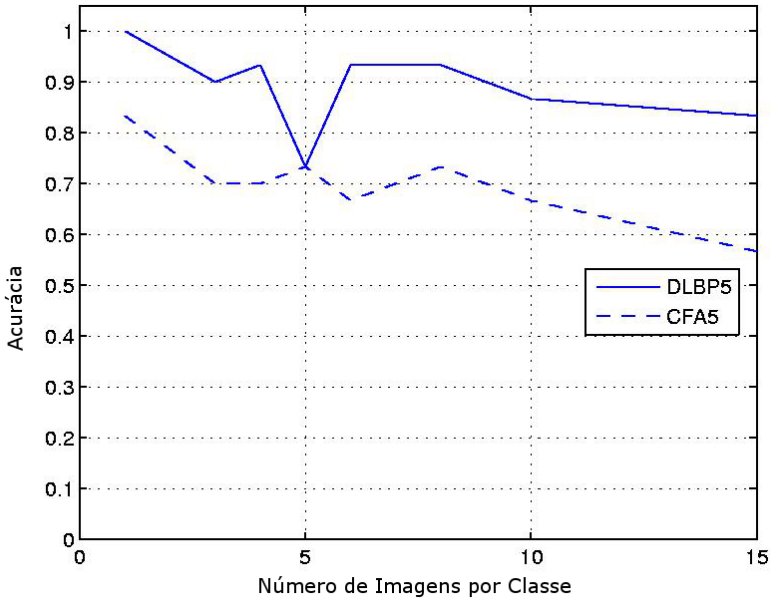

Figura 3.29: Acurácia no teste de variação do número de imagens por classe na FRD-ITJRSC-1.7 escala 5.

apresentam acurácia abaixo do valor obtido para a base FRD-ITJRSC-1.7. Uma possível justificativa para esses últimos resultados estão no fato de ser a base BioID bem menos "comportada" que a base FRD-ITJRSC-1.7. Isso porque a BioID apresenta imagens de pessoas em condições bem diferentes, com grande variação na abertura da boca, na abertura dos olhos, e assim por diante. Isso parece resultar

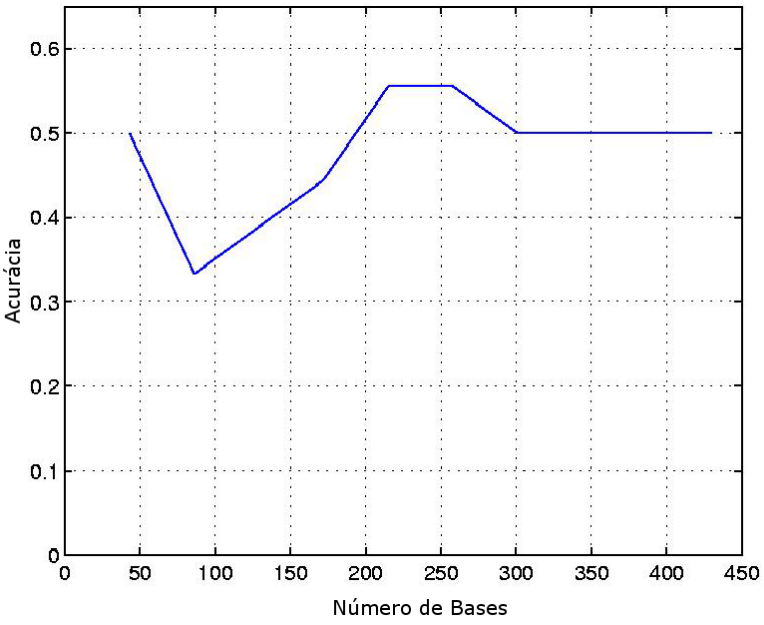

Figura 3.30: Acurácia para diferentes quantidades de bases na BioID.

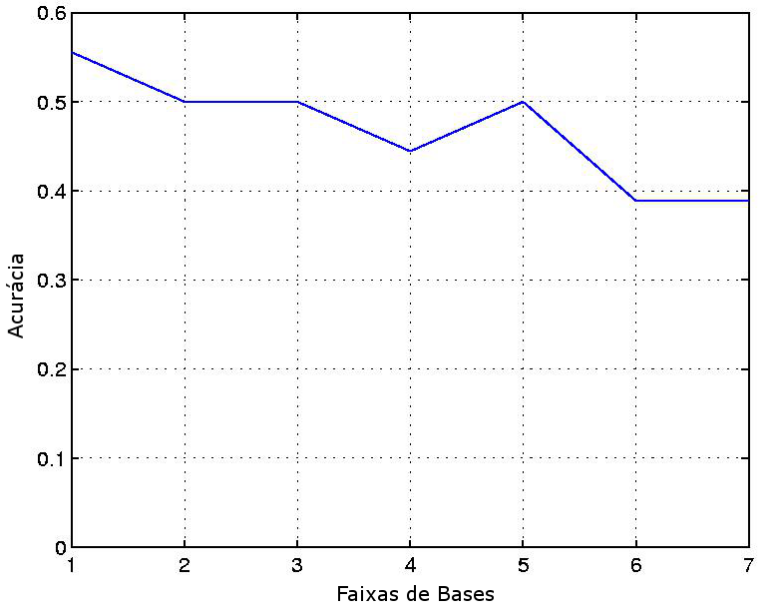

Figura 3.31: Acurácia para diferentes faixas de bases na BioID.

em um impacto bem negativo para a análise de acurácia.

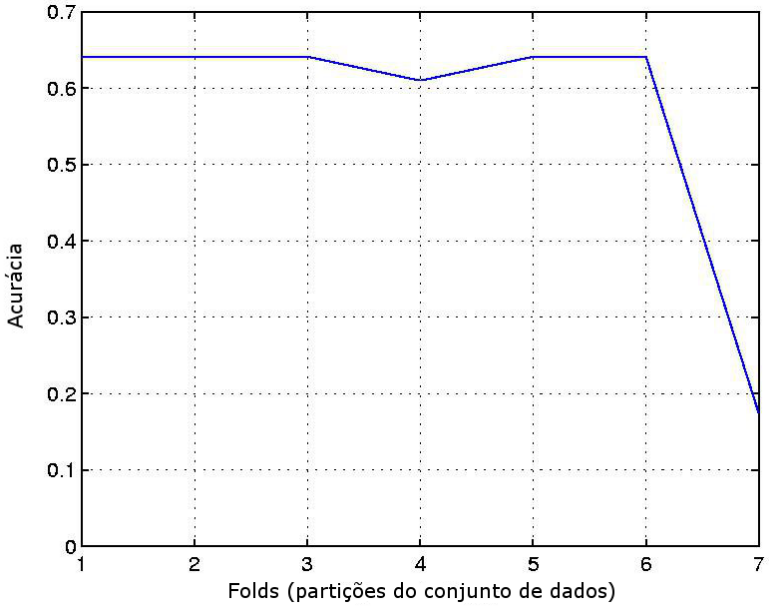

Figura 3.32: Acurácia no teste de validação cruzada para BioID escala 3.

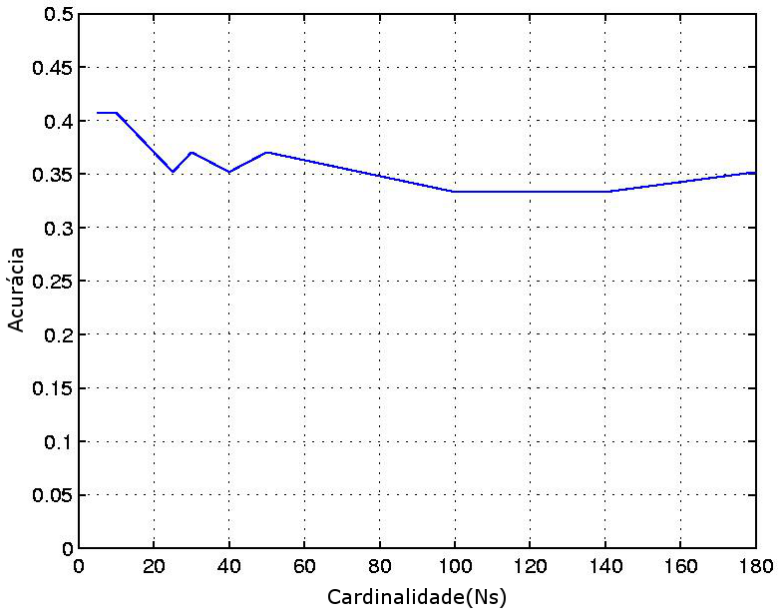

Figura 3.33: Acurácia no teste de variação de cardinalidade para BioID escala 3.

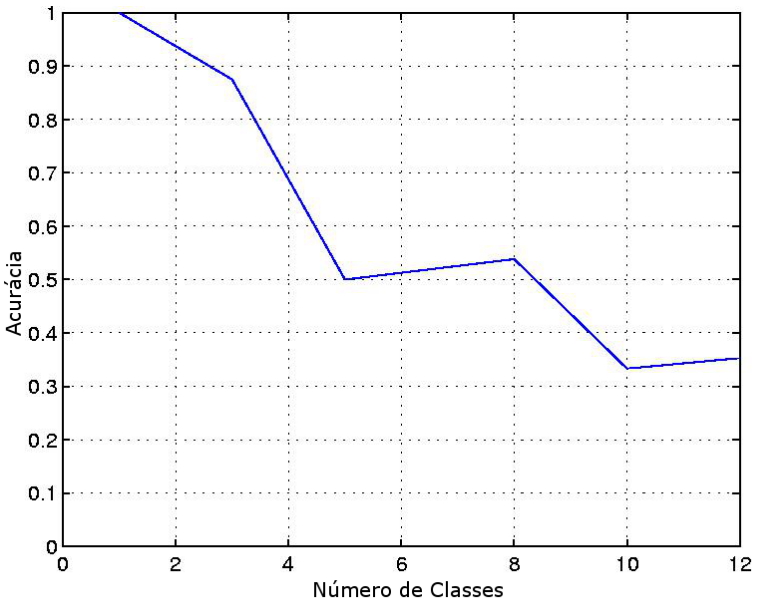

Figura 3.34: Acurácia no teste de variação do número de classes para BioID escala 3.

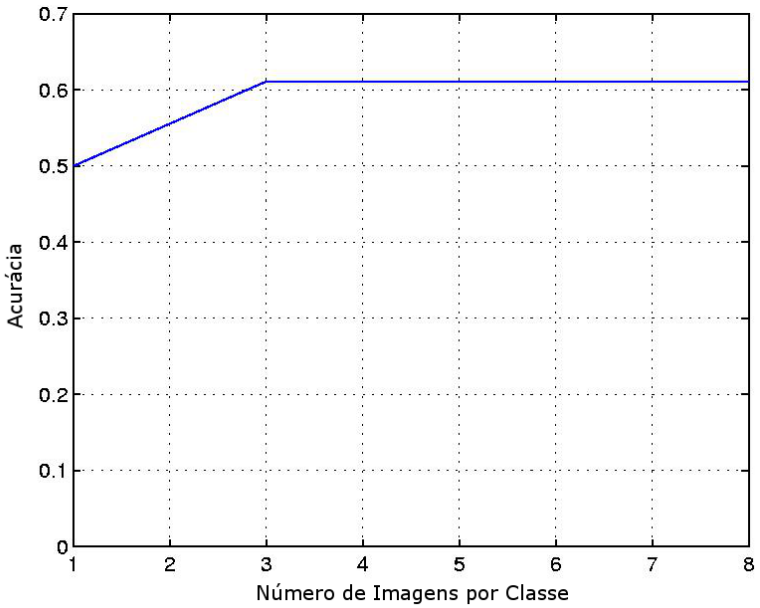

Figura 3.35: Acurácia no teste de variação do número de imagens por classe para BioID escala 3.

Capítulo 4

Conclusão

Neste capítulo, serão apresentados os seguintes tópicos:

- Conclusão.
- Propostas Futuras.

4.1 Conclusão

Podemos concluir que o algoritmo *DLBP* é uma solução promissora para o problema de reconhecimento de padrões faciais. Seus resultados foram superiores aos do algoritmo *CFA* em todos os testes considerados. Com relação ao escalamento, os resultados para imagens de 26×30 *pixels* foram, em sua maioria, inferiores aos das imagens de 43×50 *pixels*. Entretanto, mesmo para imagens de 26×30, foi possível obter valores altos de acurácia como, por exemplo, no teste de validação cruzada onde a média ficou entorno de 0,9 de acurácia ($DLBP$), valor próximo ao de 0,93 para imagens 43×50, ou no teste de variação do número de classes onde a média foi de 0,85 ($DLBP$) para imagens 43×50 e de 0,89 para imagens 26×30. Quanto ao tempo de processamento para as quantidades de imagens usadas nos testes, enquanto o algoritmo *CFA* gera seus filtros em poucos segundos, tanto para imagens de 43×50 *pixels* quanto para imagens de 26×30 *pixels*, o algoritmo de *DLBP* leva, para imagens de 43×50 *pixels*, mais de 23 horas. Um tempo muito maior, mas que se justifica pela supervisão imposta (minimização de $F(P)$ em cada passo). Entretanto, para imagens de 26×30 *pixels*, o tempo cai para menos de 20 minutos. Um valor ainda alto, quando comparado ao *CFA*, mas que demonstra a possibilidade de se alcançar tempos ainda mais baixos com um escalamento maior.

Por exemplo, para um escalamento de 8 (imagens de 16×19 *pixels*), o tempo cai para menos de 2 minutos. Por outro lado, enquanto o treinamento com o algoritmo *DLBP* é lento, o reconhecimento é rápido, já que seus vetores de projeção, binários, simplificam as operações de projeção para obtenção das *features*. No caso do *DLBP*, as operações são somente somas de números reais.

4.2 Propostas Futuras

Como possíveis trabalhos a serem realizados temos:

- A possibilidade de testar outras funções custos.
- Um cálculo mais preciso do limiar de decisão do classificador.
- A redução de complexidade do algoritmo, calculando a matriz $\mathbf{S}$, na equação (3.1), para k vizinhos mais próximos de cada amostra ao invés de calcular para todas as amostras.

Referências Bibliográficas

[1] OPENCV, "Tutorial OpenCv", Disponível em: *http://www.tecgraf.puc-rio.br/ malf/opencv/index.htm.* Acesso em: Maio de 2010, Agosto 2007.

[2] CASTRO, A. A. M. D., PADRO, P. P. L. D., "Algoritmos para Reconhecimento de Padrões", *Rev. Ciênc. Exatas*, v. 5-8, pp. 129–145, 1999-2002.

[3] INFOWESTER, "Introdução à Biometria", Disponível em: *http://www.infowester.com/biometria.php.* Acesso em: Agosto de 2010.

[4] MARTINS, M., MEDEIROS, F., MONTEIRO, M., *et al.*, "Navegação Aérea Autônoma por Imagens", Disponível em: *http://www.sige.ita.br/VIII SIGE/GE/GE014.pdf.* Acesso em: Agosto de 2010.

[5] UWE, P. D. I., HANEBECK, U. D., SENSOR-ACTUATOR-SYSTEMS, D. H., *et al.*, "Template Matching Using Fast Normalized Cross Correlation", 2001.

[6] TSAI, D. M., LIN, C. T., "Fast Normalized Cross Correlation for Defect Detection", *Pattern Recognition Letters*, v. 24, n. 15, pp. 2625–2631, November 2003.

[7] YAN, S., TANG, X., YUAN, T., "Learning Semantic Patterns with Discriminant Localized Binary Projections". In: *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 168–174, New York, June 2006.

[8] ZADEH, TUFI, FILIP, *ET AL.* (eds.), *Non-negative Matrix Factorization Methods and their Applications*, Editing House of Romanian Academy, Bucharest, 2008.

[9] LIN, Y.-C., TAI, S.-C., "A Fast Linde-Buzo-Gray algorithm in Image Vector Quantization", *IEEE Transactions on: Analog and Digital Signal Processing*, v. 45, n. 3, pp. 432–435, March 1998.

[10] LOPES, E. C., *Detecção de Faces e Características Faciais*, Relatório Técnico 45, Pontifícia Universidade Católica do Rio Grande do Sul.

[11] VIOLA, P., JONES, M., "Robust Real-Time Face Detection", *International Journal of Computer Vision*, , n. 57, pp. 18, 2004.

[12] TAVARES, A. R., SALGADO, E. C. D., *Comparação de Algoritmos de Reconhecimento de Faces em Multidões*, Relatório técnico, Universidade Federal de Minas Gerais, 2009.

[13] SINFIC, "Reconhecimento Facial: um Pouco de História e Principais Abordagens", Disponível em: *http://www.sinfic.pt/SinficWeb/displayconteudo.do2?numero=24923*. Acesso em: Abril de 2010.

[14] TURK, M. A., PENTLAND, A. P., "Face Recognition Using Eigenfaces", *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, , 1991.

[15] BRASIL, C., "Cognitec", Disponível em: *http://www.cognitec.com.br/press.htm*. Acesso em: Agosto de 2010.

[16] LIU, C., WECHSLER, H., "Face Recognition Using Evolutionary Pursuit", *ECCV'98*, v. 2, pp. 596–612, June 1998.

[17] BACH, F. R., JORDAN, M. I., "Kernel Independent Component Analysis", *Journal of Machine Learning Research*, , n. 3, pp. 1–48, July 2002.

[18] SCHOLKOPF, B., SMOLA, A., MULLER, K. R., *Nonlinear Component Analysis as a Kernel Eigenvalue Problem*, Relatório Técnico 44, Max Planck Institut, December 1996.

[19] BRONSTEIN, A. M., BRONSTEIN, M. M., KIMMEL, R., *et al.*, *3D Face Recognition without Facial Surface Reconstruction*, Relatório técnico, Technion - Computer Science Department, May 2003.

[20] M.BRONSTEIN, A., M.BRONSTEIN, M., KIMMEL, R., "Expression-Invariant 3D Face Recognition", *Proc.Audio and Video-based Biometric Person Authentication*, pp. 62–69, 2003.

[21] GORODNICHY, D. O., "Video-Based Framework for Face Recognition in Video". In: *Proc. of Second Canadian Conference on Computer and Robot Vision*, pp. 330–338, Victoria, Canada, May 2005.

[22] ZHOU, S., KRUEGER, V., CHELLAPPA, R., "Probabilistic Recognition of Human Faces from Video", *Computer Vision and Image Understanding*, v. 91, pp. 214–245, 2003.

[23] WIKIPEDIA, "Characteristics and Features of Problems Solved by Greedy Algorithms", Disponível em: *http* : *//www.personal.kent.edu* Acesso em: Maio de 2010.

[24] BRAGA, A. C. D. S., *Curvas ROC: Aspectos Funcionais e Aplicações*. Tese Ph.D., Universidade do Minho, Dezembro 2000.

[25] SOUZA, C., "Análise do Poder Discriminativo Através de Curvas ROC", Disponível em: *http://crsouza.blogspot.com/2009/07/analise-de-poder-discriminativo-atraves.html* Acesso em: Maio de 2010.

[26] BIOMETRICS, G., "Equal Error Rate", http://www.griaulebiometrics.com/page/pt-br/book/understanding-biometrics/evaluation/accuracy/matching/interest/equal.

[27] XIE, C., SAVVIDES, M., KUMAR, B. V., "Redundant Class-Dependence Feature Analysis Based on Correlation Filters Using FRGC2.0 Data". In: *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA, 2005.

[28] PHILLIPS, P. J., FLYNN, P. J., SCRUGGS, T., *et al.*, "Overview of the Face Recognition Grand Challenge". In: *Proceedings International Conference on Computer Vision and Pattern Recognition*, pp. 947–954, 2005.

[29] BIOID, "BioID Face Database", Disponível em: *http://www.bioid.com/support/downloads/software/bioid-face-database.html.* Acesso em: Junho de 2010, 2008.

[30] DATABASE, T. F., "The Facial Recognition Technology FERET Database", Disponível em: *http://itl.nist.gov/iad/humanid/feret/feret_master.html.* Acesso em: Junho de 2010, 2001.

[31] KOHAVI, R., "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection". In: *International Joint Conference on Artificial Intelligence*, 1995.

[32] PILLAI, A. P. S. U., *Probability, Random Variables and Stochastic Process.* MacGraw-Hill, 2001.

[33] TURK, M., PENTLAND, A., "Eigenfaces for Recognition", *Journal of Cognitive Neuroscience*, v. 3, n. 1, pp. 71–86, 1991.