UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE MATEMÁTICA
INSTITUTO TÉRCIO PACITTI DE APLICAÇÕES E PESQUISAS COMPUTACIONAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

**RAMIRO PEREIRA DE MAGALHÃES**

# ATRIBUIÇÃO DE PESOS A HAAR *WAVELETS* PARA A DETECÇÃO DE FACES

Rio de Janeiro
2014

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE MATEMÁTICA
INSTITUTO TÉRCIO PACITTI DE APLICAÇÕES E PESQUISAS COMPUTACIONAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

**RAMIRO PEREIRA DE MAGALHÃES**

# ATRIBUIÇÃO DE PESOS A HAAR *WAVELETS* PARA A DETECÇÃO DE FACES

Dissertação de Mestrado submetida ao Corpo Docente do Departamento de Ciência da Computação do Instituto de Matemática, e Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários para obtenção do título de Mestre em Informática.

Orientador: Josefino Cabral Melo Lima

Rio de Janeiro
2014

M188 Magalhães, Ramiro Pereira de

Atribuição de pesos a Haar *wavelets* para a detecção de faces / Ramiro Pereira de Magalhães. – 2014.
92 f.: il.

Dissertação (Mestrado em Informática) – Universidade Federal do Rio de Janeiro, Instituto de Matemática, Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais, Programa de Pós-Graduação em Informática, Rio de Janeiro, 2014.

Orientador: Josefino Cabral Melo Lima.

1. Detecção de Padrões. 2. Arcabouço de Viola e Jones. 3. AdaBoost. 4. Haar *wavelet*. 5. Análise de Componentes Principais. 6. Aprendizagem de Máquina. – Teses. I. Lima, Josefino Cabral Melo (Orient.). II. Universidade Federal do Rio de Janeiro, Instituto de Matemática, Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais, Programa de Pós-Graduação em Informática. III. Título

CDD

RAMIRO PEREIRA DE MAGALHÃES

**Atribuição de pesos a Haar *wavelets* para a detecção de faces**

Dissertação de Mestrado submetida ao Corpo Docente do Departamento de Ciência da Computação do Instituto de Matemática, e Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários para obtenção do título de Mestre em Informática.

Aprovado em: Rio de Janeiro, _____ de _____ de _____.

_____

Prof. Josefino Cabral Melo Lima (Orientador)

_____

Prof. Adriano Joaquim de Oliveira Cruz

_____

Prof. José Francisco Moreira Pessanha

_____

Profa. Priscila Machado Vieira Lima

_____

Profa. Valeria Menezes Bastos

_____

Prof. Woulter Caarls

*Ao bebê na barriga da minha esposa.*

# AGRADECIMENTOS

Agradeço aos meus pais, Leide Maria Pereira e Victor Hugo de Magalhães, pelo tanto que se dedicaram aos seus filhos. O quanto mais ouço, penso e me aproximo da paternidade, mais me impressiona o esforço que eles fizeram por mim. Foram meus pais, através de muitas palavras e ações autênticas, fundadas em suas próprias experiências de vida, que me ensinaram a continuar a fazer o melhor que puder, apesar das dificuldades.

Agradeço à minha esposa, Andrea Klein Mattioli, por estar sempre presente e compartilhar comigo tantas coisas pequenas e grandes que ocorrem em nossas vidas; por se dispôr a me escutar tagarelar sobre as vitórias, a lamentar as derrotas e por conversar sobre o que está entre esses extremos; por todo o apoio — na forma de um cafuné ou de uma xícara de café — dado durante as longas jornadas de estudo que se seguiam após as longas jornadas de trabalho; por toda a paciência em me esperar "só mais um pouquinho" para "resolver só mais esse probleminha" antes de sairmos para almoçar, dentre várias outras esperas à que ela se sujeitou enquanto eu me dedicava ao curso de mestrado; e por acreditar em mim e nos planos que compartilhamos.

Agradeço aos meus amigos, todos eles, pela companhia, piadas, exemplos e sinceridade! Em especial, agradeço à trupe Alexandre de Paiva Rio Camargo, Diego Werneck Arguelhes, Guilherme Salgado Braga, Mário Luis Carneiro Pinto de Magalhães e Yuri Kasahara, amigos de looooonga data, por toda a torcida e apoio, e por compartilharem suas experiências acadêmicas que me ajudaram a estabelecer uma clara expectativa sobre o que seria o mestrado. E agradeço ao Bruno Nicolau Nunes e ao Fabiano de Souza Sanches, amigos desde o estágio no SIGA, durante nosso curso de graduação na UFRJ, e hoje colegas de trabalho, pela companhia e alegria que me passam.

Agradeço ao meu orientador, Josefino Cabral Melo Lima, pela orientação e apoio durante o curso de mestrado, e por colocar os desafios onde, inicialmente, eu não podia alcançar. Sem isso, não teria aprendido tanto quanto sinto que aprendi.

# RESUMO

Magalhães, Ramiro Pereira de. **Atribuição de pesos a Haar *wavelets* para a detecção de faces**. 2014. 92 f. Dissertação (Mestrado em Informática) - PPGI, Instituto de Matemática, Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2014.

Viola e Jones publicaram em 2001 um arcabouço para detecção automática de faces que se equiparou, em termos de precisão, ao estado da arte, mas superou todos os demais em desempenho. O sucesso do trabalho desses autores chamou bastante atenção da comunidade científica, que contribuiu de muitas formas.

Nesse arcabouço, o detector de faces usa um extrator de características originalmente apresentado por Papageorgiou e colaboradores, chamado Haar *wavelet*. Uma Haar *wavelet* codifica as diferenças de intensidade locais de uma imagem em valores escalares, o que é feito através da simples soma ponderada das intensidades de pixels de regiões específicas. Pavani e colaboradores fizeram uma das contribuições importantes para o arcabouço de Viola e Jones: apresentaram métodos de atribuição de pesos às Haar *wavelets* que melhoraram a precisão do detector de faces.

Esta dissertação apresenta um método de atribuição de pesos a características baseadas em Haar *wavelets* complementar aos apresentados por Pavani et al.. O método consiste na aplicação da análise de componentes principais (ACP) no espaço vetorial de valores da característica produzido com as instâncias positivas durante o treinamento do detector de faces. A componente principal que projeta tais instâncias de forma mais compacta é usada como o vetor de pesos da Haar *wavelet*. Além disso, propõe-se uma nova função extratora de características que usa os parâmetros das distribuições estatísticas desse mesmo espaço vetorial. Ambas as propostas foram construídas sobre a hipótese de que o espaço vetorial de valores da característica oriundo das instâncias negativas se distribuem uniformemente. Os experimentos realizados com essas características superaram os resultados produzidos com outros detectores, mas a hipótese de uniformidade das distribuições de instâncias negativas não se confirmou.

**Palavras-chave:** Detecção de Padrões, Arcabouço de Viola e Jones, AdaBoost, Haar *wavelet*, Análise de Componentes Principais, Aprendizagem de Máquina.

# ABSTRACT

Magalhães, Ramiro Pereira de. **Atribuição de pesos a Haar *wavelets* para a detecção de faces**. 2014. 92 f. Dissertação (Mestrado em Informática) - PPGI, Instituto de Matemática, Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2014.

Viola and Jones published in 2001 an automatic face detection framework that was as precise as the state of the art, but performed faster. Their success drew attention of the scientific community, that responded with many extensions and contributions.

In such framework, the face detector uses a feature extractor named Haar wavelet that was originally proposed by Papageorgiou and colleagues. A Haar wavelet encodes the local intensity differences of an image in scalar values through the simple weighted sum of pixel intensities of specific regions. Pavani and colleagues made an important contribution to Viola and Jones' framework: they developed methods to assign better weights to the Haar wavelets, improving the face detector precision.

This thesis presents a new weight assignment method to Haar-like features that complements those shown by Pavani et al.. It consists in the application of principal component analysis in the feature's values vector space produced with the positive instances while training the face detector. The Haar wavelet weight vector is the principal component vector that projects the instances' feature values in the most compact way. Besides, a new feature extractor that uses parameters from the statistic distribution of the same vector space is presented. Both proposals were built upon the following hypothesis: the feature's values vector space originated from the negative instances is uniformly distributed. The experiments made with such features show a better result than those made with other detectors, but the uniformity hypothesis could not be confirmed.

**Keywords:** Pattern Detection; Viola-Jones Framework; AdaBoost; Haar Wavelet; Principal Component Analysis; Machine Learning.

# LISTA DE FIGURAS

# LISTA DE TABELAS

# SUMÁRIO

# 1 INTRODUÇÃO

O que as câmeras digitais modernas, o Departamento de Defesa americano e o Facebook© têm em comum? Todos executam em seus sistemas computacionais algum algoritmo de detecção de faces! As câmeras digitais detectam faces para tirarem uma fotografia automaticamente quando todas as pessoas presentes na foto estiverem sorrindo. O Departamento de Defesa americano emprega tal tecnologia para "apoiar profissionais de segurança, inteligência e polícia no cumprimento de seus deveres" [37]. Já o Facebook© deseja descobrir com quem e por onde andam seus usuários na vida real, melhorando sua capacidade de lhes fornecer anúncios e conteúdos que realmente lhes interesse.

Apesar das aplicações mencionadas acima serem diferentes, um detector de faces está presente em todas elas, afinal, antes duma máquina interpretar uma expressão facial, ou de reconhecer que uma face é de uma pessoa em particular, ela precisa ter a imagem da face para analisá-la. A função dum detector é encontrar numa imagem qualquer onde estão as faces, de modo que essa informação possa ser útil para consumo por humanos ou máquinas. É importante deixar claro que a detecção de faces em imagens já é uma aplicação específica de detecção de objetos em geral que tornou-se um problema de grande interesse para a comunidade científica devido à sua dificuldade, oriunda da vasta variedade de formas e cores que a face humana toma, quanto à sua aplicabilidade, exemplificada acima.

Como a detecção de objetos ou faces é, muitas vezes, apenas a parte inicial dum processo automático, ela deve ser bastante precisa e veloz. Por precisa, entende-se que, idealmente, somente instâncias do objeto de interesse devem constar entre os resultados de detecção. Por veloz, entende-se que os detectores devem ser rápidos o bastante para esquadrinhar vídeo em tempo real ou grandes bases de dados de imagens ou vídeo. Se um detector é impreciso, isso é, contém em seus resultados muitas instâncias de objetos que não são de interesse (chamados objetos de fundo),

os próximos processos automáticos tenderão a ficar sobrecarregados, deixando todo o sistema lento.

O "cérebro" do detector é o classificador: uma função responsável por classificar uma instância em objeto de interesse ou de fundo. De maneira geral, o classificador mede características das instâncias que lhes são fornecidas e usa tais medidas para produzir a classificação. As medidas das características são feitas por funções especiais, denominadas extratores de características. A escolha de extratores de características afeta diretamente a precisão e a velocidade do classificador, e é muito comum tal escolha ser feita com o suporte de diversas teorias, modelos e ferramentas matemáticas pertinentes às áreas de classificação de padrões e aprendizagem de máquina.

Talvez um dos métodos mais populares [66] para a produção dum detector veloz e preciso seja o proposto em 2001 por Viola e Jones [61]. Combinando uma poderosa técnica de aprendizagem de máquina chamada AdaBoost [17] com as características baseadas em Haar *wavelets* propostas por Papageorgiou e colaboradores [39], foram capazes de treinar classificadores de faces bastante precisos. O detector que comporta os classificadores opera como proposto por Rowley et al. [48], mas acrescido de algumas técnicas que garantiram sua rápida operação.

O aperfeiçoamento dos resultados obtidos por Viola e Jones foram resultado de esforços de múltiplos pesquisadores. Um desses trabalhos importantes é o de Pavani e colaboradores [41], que perceberam que os valores obtidos pelas características baseadas em Haar *wavelets* durante a inspeção de imagens podem ser interpretados como vetores dum espaço vetorial especial. Além disso, notaram que os vetores produzidos a partir de imagens do objeto de interesse concentram-se ao entorno de um valor central, enquanto que vetores produzidos com imagens de fundo tendem a se espalhar de outra forma, e ao entorno de outra medida. Assim, descreveram e testaram técnicas que ajudaram a distinguir melhor entre o que é uma imagem de face ou de fundo.

Esta dissertação estende o trabalho de Pavani e colaboradores propondo um

novo método de atribuição de parâmetros aos extratores de características partindo de observações das distribuições de vetores em seus respectivos espaços vetoriais, elaborado de modo a reduzir o tempo de treinamento dos classificadores. Além disso, apresenta um novo extrator de características que usa essas mesmas informações, mas para tentar melhorar sua capacidade de separar faces de fundos. Ao fazer isso, as propriedades do espaço vetorial das características são exploradas e explicitadas.

No capítulo 2 revisam-se os conhecimentos fundamentais para a compreensão adequada desta dissertação: o impulsionamento e seu principal algoritmo, o AdaBoost; o arcabouço para detecção de faces de Viola e Jones; as características baseadas em Haar *wavelets*; e a análise de componentes principais.

No capítulo 3 é feita a revisão do estado da arte dos conhecimentos considerados fundamentais para este trabalho e que foram estudados durante a definição de seu escopo com algum relevância para o tema dessa dissertação. Dá-se atenção especial ao trabalho de Pavani e colaboradores, de onde partiram as observações que levam à elaboração das hipóteses e contribuições originais desta dissertação.

O capítulo 4 discute as hipóteses levantadas, suas consequências, se verdadeiras, e os métodos usados para explorá-las e verificá-las. Antes disso, é apresentado o espaço de características retangulares simples, um espaço vetorial formado durante a extração de características das imagens.

O capítulo 5 descreve todo o preparo e execução dos experimentos realizados, as fontes das quais extraíram-se os dados utilizados nos experimentos, as bibliotecas e ferramentas de *software* reusados e construídos, e as técnicas empregadas para tratar diversos aspectos sutis, mas importantes, dos experimentos. Também é nesse capítulo em que as técnicas de avaliação e os resultados são apresentados e discutidos.

Por fim, o capítulo 6 conclui esta dissertação e apresenta alguns experimentos e possibilidades ainda passíveis de exploração.

# 2 REVISÃO DA LITERATURA

A detecção automática de objetos em imagens é, ainda hoje, uma tarefa complicada, e a detecção de faces é assunto de particular interesse para a comunidade científica, dada sua dificuldade e aplicabilidade. Tal dificuldade existe em grande parte devido a grande variedade de tipos de pele, olhos, pelos, cabelo, formato, texturas e marcas que a face humana pode apresentar. Ainda multiplicam essa dificuldade as expressões, acessórios, rotações, tamanhos, condições de iluminação, ruído, etc. que uma face pode apresentar numa imagem. A aplicação é ampla pois a detecção da face é o primeiro passo para seu processamento posterior em aplicações de fins diversos, tais como a busca e reconhecimento de pessoas, reconhecimento de gestos, reconhecimento de emoções, suporte à compreensão da fala por leitura labial, dentre outros.

Essa seção contém uma revisão do arcabouço para detecção de faces de Viola e Jones [61]. Além disso, essa seção também revisa a análise de componentes principais (ACP), uma técnica bastante importante para a contribuição feita nesta dissertação.

## 2.1 Impulsionamento

Impulsionamento é uma técnica de aprendizagem de máquina baseada na ideia de que é possível formar uma regra de classificação bastante precisa, comumente chamada de classificador forte, combinando regras imprecisas, chamadas de classificadores fracos ou básicos. No contexto da teoria de impulsionamento, classificadores também podem ser chamados de hipóteses, com o mesmo significado.

O algoritmo responsável por essa combinação recebe como parâmetros uma massa de dados rotulados e um objeto denominado aprendiz fraco. Basicamente, o trabalho desse algoritmo é apresentar toda ou parte da massa de dados ao aprendiz fraco, mas destacando certos exemplos de acordo com a dificuldade de classificá-los. Com isso, o aprendiz produz um classificador fraco que melhor classifica a infor-

mação dada durante o treinamento, considerando que é mais importante classificar bem os exemplos mais destacados do que os pouco destacados. Esse classificador fraco será entregue ao algoritmo para a reavaliação da importância de cada exemplo da massa de dados à luz dessa hipótese, e, em seguida, será incluído no classificador forte junto com um parâmetro que reflete sua qualidade. Isso se repete até que um critério de parada ocorra. Ao executar esses passos, diz-se que este algoritmo está impulsionando as hipóteses fracas produzidas pelo aprendiz fraco.

Se por um lado o algoritmo de impulsionamento é o mesmo para múltiplos casos, o aprendiz fraco é um componente dependente do problema. Sua escolha não pode ser completamente arbitrária pois, para funcionar satisfatoriamente, é importante que os classificadores fracos que produz classifiquem os dados com um erro ponderado pela importância de cada exemplo inferior a 50%. Fora esse aspecto, o aprendiz fraco é visto pelo algoritmo como uma caixa preta.

O principal algoritmo de impulsionamento é o AdaBoost [15] que será apresentado na seção 2.1.2.

### 2.1.1 Histórico sobre a teoria do impulsionamento

Essa subseção apresenta uma rápida revisão sobre o surgimento, no início da década de 80, da teoria que suporta o impulsionamento e os principais trabalhos que culminaram no surgimento do AdaBoost, em meados da década de 90. A intenção aqui é apenas a de apresentar um apanhado de fontes que contribuem para a boa compreensão do algoritmo e alguns de seus aspectos teóricos importantes.

A base para o estudo de impulsionamento é o modelo sobre aprendizado de máquinas proposto por Valiant [59] denominado Provavelmente Aproximadamente Correto (PAC, do inglês *Probably Approximately Correct*). Nesse modelo, um algoritmo aprendiz usa uma massa de dados de treinamento para escolher, com alta probabilidade de sucesso, uma função que classifique algo com **baixo** erro de generalização. Para fazer essa escolha, cada instância dos dados de treinamento possui um peso que representa a importância desse dado para o aprendiz. Aprendizagem PAC

é conhecida também como aprendizagem forte e, se por um lado ela é teoricamente factível, na prática é frequentemente inviável.

Trabalhando nesse modelo para torná-lo tratável em situações práticas, Kearns et al. [28] propuseram como alternativa a aprendizagem fraca, que consiste em fazer o algoritmo aprendiz analisar amostras para escolher, com alta probabilidade de sucesso, uma função que o classifique com erro de generalização **alto**, ou, mais especificamente, um pouco menor que 50%. Os autores também formularam o problema do impulsionamento de hipóteses (*hypothesis boosting*), que constitui em (1) verificar quais classes podem ser aprendidas por quais modelos de aprendizagem; (2) se é possível tornar um aprendiz fraco em um aprendiz forte e; (3) se aprendizagem fraca implica em aprendizagem forte e vice versa, ou seja, se tudo o pode ser aprendido num nível poder ser também noutro.

A resposta para essas três questões foi dada por Schapire, que em [50] prova que uma classe é fortemente apreensível se e somente se for fracamente apreensível. Com isso, apresentou o primeiro algoritmo de impulsionamento que comprovadamente produzia resultados em tempo polinomial. Em seguida, Freund [14] apresentou melhoras significativas sobre o algoritmo de Schapire.

Uma importante limitação prática dos algoritmos citados acima era a necessidade de se conhecer o comportamento estatístico do algoritmo de aprendizagem fraca, mais especificamente o viés que ele apresentaria ao classificar as amostras destinadas ao treinamento. Em [15], Freund e Schapire demonstraram que essa limitação pode ser vencida com um novo algoritmo que se adapta às amostras que recebe durante o treinamento. Surgia o Impulsionamento Adaptativo (*Adaptative Boosting*), ou AdaBoost, que conferiu aos pesquisadores o prêmio Gödel em 2003. Eles escreveram uma excelente introdução ao impulsionamento adaptativo [18].

Daquele momento em diante as comunidades de pesquisadores em aprendizado de máquinas, inteligência artificial, matemática e estatística apresentaram grande interesse na teoria e prática dessa técnica. Foram propostas múltiplas variantes do AdaBoost que tratavam de suas limitações, e melhoraram seu desempenho (capítulo

3). Um trabalho de destaque é o de Viola e Jones [61] onde classificadores fortes foram criados e combinados para detectar a presença de faces frontais em imagens. Os resultados se equiparavam ao estado da arte na época. Um grande número de pesquisadores se inspiraram na metodologia desses autores para resolver problemas similares.

### 2.1.2 AdaBoost

Esta subseção apresenta o AdaBoost e trata de algumas questões de projeto do aprendiz fraco, visando a aplicação desses algoritmos em problemas reais.

O objetivo de impulsionamento é, ao fim de $T$ iterações, gerar um classificador forte $H$ que aplica instâncias $x$ do domínio $X$ em elementos $y$ do conjunto imagem $Y$.

Todo impulsionamento recebe como entrada um conjunto de exemplos (também chamados de instâncias) dos objetos por classificar. Tais exemplos devem estar rotulados, isso é, identificados como pertencentes à classe ou não. Os exemplos rotulados serão representados por $(x_1, y_1), \ldots, (x_m, y_m)$. Ao longo desse trabalho, $m$ representa a quantidade de exemplos presentes no conjunto em questão; $i = 1, \ldots, m$; $x_i \in X$, o domínio, i.e. o conjunto de todos os objetos possíveis; e $y_i \in Y = \{-1, +1\}$, descreve as classificações possíveis onde $+1$ indica que o objeto pertence à classe, e $-1$ o oposto.

O aprendiz fraco é um algoritmo que se integra ao AdaBoost (ou a outro impulsor) para gerar os classificadores fracos que serão combinados em um classificador forte. O aprendiz fraco é tratado pelo impulsor como uma caixa-preta e será invocado uma vez em cada iteração para fornecer um classificador fraco $h_t(x)$, onde $t = 1, \ldots, T$ identifica a iteração em que se obteve o classificador. Todo classificador fraco produzido deve conformar com a premissa do aprendizado fraco, que afirma que um classificador fraco deve ser um pouco melhor que o palpite aleatório, isso é, deve classificar um exemplo corretamente com probabilidade superior a 50%.

O classificador forte $H : X \mapsto Y$ é o produto final do impulsionamento. Ele

nada mais é que uma coleção dos classificadores fracos associados a pesos que representam a importância de cada classificador, e são produzidos durante a execução do algoritmo.

Uma particularidade dos algoritmos de impulsionamento é que eles mantém pesos representados por $D_t(i)$ associados a cada instância $i$. Note que $D$ é atualizado em cada iteração $t$. $D_t$ é similar a uma distribuição de probabilidades pois $\sum_{\forall i} D_t(i) = 1$. $D_t(i)$ é um indicador da dificuldade de se classificar o exemplo $i$ na iteração $t$, e seu valor é obtido a partir da quantidade de erros cometidos pelo classificador fraco. O aprendiz fraco dará maior importância aos exemplos mais difíceis quando estiver produzindo um classificador fraco.

A qualidade de um classificador fraco é medida pelo seu erro ponderado de classificação $\epsilon_t$, cujo cálculo nada mais é que o somatório de $D_t(i)$ para todo $i$ em que $h_t(x_i) \neq y_i$. É função do aprendiz fraco produzir classificadores que garantem que esse erro seja ligeiramente menor que 50%.

Outro peso importante é usado pelo classificador forte para relativizar a importância do classificador fraco, além de ser usado na atualização da distribuição de pesos. Representado por $\alpha_t \in R$, é obtido diretamente a partir de $\epsilon_t$.

O algoritmo 1 descreve o Real AdaBoost, conforme apresentado por Schapire [51]. Essa descrição, além de ser de leitura e interpretação mais simples, é mais geral que a primeira notação dada ao AdaBoost.

O primeiro passo é iniciar $D_1$. Na primeira iteração, a dificuldade de classificação de cada exemplo é a mesma.

Na $t$-ésima iteração, o aprendiz fraco produz um classificador fraco que minimize a chance de se obter aleatoriamente, de acordo com $D_t$, um exemplo que seja mal classificado. A notação $Pr_{i\sim D}[.]$ significa a probabilidade de se obter $i$ de acordo com a distribuição $D$. Há duas formas típicas do aprendiz fraco operar e calcular $\alpha_t$: através do impulsionamento por reamostragem; ou por reponderação.

Quando o impulsionamento é feito por reamostragem, o aprendiz fraco recebe uma amostra aleatória de tamanho $m'$ dos exemplos, de acordo com $D_t$. O tama-

**Algoritmo 1:** Uma das versões do AdaBoost.

**Dados**: $(x_1, y_1), \ldots, (x_m, y_m)$ onde $x_i \in X$ e $y_i \in \{-1, +1\}$
Inicie $D_1(i) = 1/m$ para $i = 1, \ldots, m$;
**para** $t = 1, \ldots, T$ **hacer**
  Forneça $D_t$ ao aprendiz fraco e dele obtenha $h_t : X \mapsto \{-1, +1\}$, tal que $h_t$ minimize $\epsilon_t = Pr_{i \sim D_t}[h_t(x_i) \neq y_i]$
  Seja $\alpha_t = \frac{1}{2} \ln(\frac{1-\epsilon_t}{\epsilon_t})$
  **para** *i = 1, ..., m* **hacer**

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

    onde $Z_t$ é um fator de normalização escolhido de modo que $D_{t+1}$ seja uma distribuição.
  **fin para**
**fin para**
**Retorne** $H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$.

nho de $m'$ pode ser maior, menor ou igual a $m$, dependendo das características do conjunto de dados de treinamento e do problema de classificação. Note que, independente do tamanho de $m'$, espera-se que certas instâncias apareçam mais de uma vez na amostra. Nesse método, essa é a maneira de fazer o aprendiz fraco dar maior importância às instâncias mais difíceis.

O aprendiz fraco busca dentre os classificadores fracos candidatos aquele que menos erra ao classificar os exemplos. Assim, considerando que há $K$ classificadores fracos, ao menos $K \times m'$ invocações aos classificadores fracos seriam necessárias para encontrar o que minimiza $\epsilon_t$.

Já quando é impulsionado por reponderação, o aprendiz fraco recebe $D_t$ e os $m$ exemplos. A busca do classificador fraco funcionará de forma similar, mas a avaliação do erro ponderado muda: ao invés de contar os casos em que a classificação foi mal feita, se somam os pesos $D_t(i)$ de cada exemplo mal classificado, isso é

$$\sum_{i:h_t(x_i) \neq y_i} D_t(i). \tag{2.1}$$

Ambos os métodos produzem o mesmo resultado final: tenderão a ser desfavorecidos nessa etapa de seleção os classificadores fracos que errarem a classe dos exemplos mais frequentes, no caso da reamostragem, ou mais importantes, no caso da reponderação. Seiffert e colaboradores [54] compararam esses métodos e descobriram que o impulsionamento por amostragem parece desempenhar melhor em geral.

De posse do erro ponderado $\epsilon_t$, o AdaBoost obtém $\alpha_t$ que serve tanto para a atualização de $D_t$ quanto para ponderar a classificação dada pelo classificador $h_t$. Observe que o valor de $\alpha_t$ será positivo quando $\epsilon_t$ for inferior a $1/2$, e negativo no caso contrário.

O passo seguinte é a atualização dos pesos. Supondo que $h_t$ cumpre a premissa do aprendizado fraco, $\alpha_t$ será um número positivo. Então, cada $D_t(i)$ será atualizado de acordo com o resultado de $h_t(x_i)$ quando comparado com $y_i$. Se $h_t$ classifica corretamente $x_i$, ou seja $h_t(x_i) = y_i$, tem-se que $D_{t+1}(i) < D_t(i)$, com o oposto ocorrendo caso a $h_t$ classifique mal a amostra $x_i$. Em outras palavras, os peso duma amostra diminui se o classificador fraco a classifica corretamente, mas aumenta caso contrário. Com efeito, as amostras de classificação difícil terão maior importância que as de classificação fácil. Em toda iteração, $D_t(i)$ é normalizado por um valor $Z_t$ escolhido de modo que $D_{t+1}$ manterá características de uma distribuição de probabilidades ($\sum_{\forall i} D_{t+1}(i) = 1$).

No último passo do algoritmo, tanto $\alpha_t$ quanto $h_t$ são adicionados ao classificador forte, então se inicia uma nova iteração. Para o AdaBoost, o classificador forte é simplesmente a saída do algoritmo e não é usado internamente.

Como é possível observar, a dificuldade de se classificar corretamente uma instância tem relação com os erros cometidos pelo aprendiz fraco ao tentar criar classificadores fracos. Quanto mais erros forem cometidos sobre certos exemplos, maior serão suas importâncias para o aprendiz que será forçado a produzir um classificador fraco que as classifique corretamente. Quando isso ocorrer, outros exemplos passarão a ser mais importantes e o ciclo se repete.

## 2.2 Detector de faces

Um detector automático de objetos determina a presença de certo tipo de objeto numa imagem qualquer. Caso este objeto esteja na imagem, o detector deve também determinar a região em que ele está [66].

Em 2001, Paul Viola e Michael J. Jones [61] propuseram um detector de faces bastante poderoso cuja precisão se equiparava ao estado da arte ([48] [53]), porém era muitas vezes mais rápido. Em 2004 os mesmos autores publicaram um artigo mais detalhado sobre o que fizeram [60].

Esse detector opera como uma **janela deslizante**, isso é, uma região retangular que pode ser posicionada sobre a imagem em qualquer posição, contanto que seus lados estejam paralelos aos lados da imagem, e também pode ter seus lados uniformemente redimensionados. Em [60], essa janela desliza horizontal e verticalmente pela imagem extraindo seu conteúdo sempre que muda de posição e entregando-o ao **classificador**, que é o efetivo responsável por declarar se o conteúdo da janela deslizante é ou não uma instância do objeto de interesse. Após assumir todas as posições possíveis na imagem, a janela é redimensionada e o processo se repete até que a janela deslizante alcance um tamanho máximo, ou se torne maior que alguma das dimensões da imagem completa. Esse processo foi inspirado pelo descrito por Rowley et al. [48].

A quantidade de exames que um detector realiza numa única imagem é muito grande. De fato, a união de todas as subjanelas formadas por este processo é um conjunto sobrecompleto (do inglês, *overcomplete*) da imagem original. Infelizmente, a probabilidade de certa subjanela conter uma face é, tipicamente, muito pequena, o que significa que despende-se muito tempo classificando subjanelas desinteressantes. Para tratar desse problema, e tornar o detector mais rápido, Viola e Jones decidiram que o classificador final deve ser estruturado como uma cadeia (ou cascata) de classificadores fortes intermediários (também chamados de nós ou estágios, neste contexto), que avaliam em sequência uma mesma subjanela. Basta um classificador da cadeia concluir que a subjanela não é um objeto de interesse para que ela

seja classificada como tal. Os estágios são treinados de maneira que os posicionados mais no início da cascata são menos complexos e poderosos que aqueles mais ao final, porém todos devem rejeitar o maior número possível de subjanelas sem faces, ao passo que rejeitam o mínimo de subjanelas com faces. Assim, os primeiros e menos complexos estágios eliminam rapidamente um grande número de janelas que obviamente não são faces, enquanto os estágios mais internos e mais complexos tratam das subjanelas de difícil classificação. Baker foi o propositor dessa "cascata rejeitora" [4], ilustrada na figura 2.1.

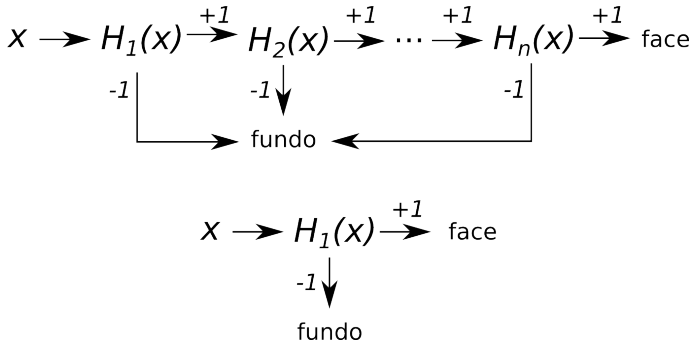


Figura 2.1: Cascata rejeitora composta de classificadores fortes $H_i(x)$ (acima) comparada com um classificador monolítico (abaixo). Em ambos os casos, basta que um classificador forte classifique uma amostra $x$ como $-1$ para para que todo o processo seja interrompido. Contudo, se todo classificador forte classificar $x$ como $+1$, $x$ será classificado como face.

Essa cadeia é resultado do seguinte processo: dois limiares são atribuídos aos nós: o máximo de falsos positivos e o mínimo de verdadeiros positivos. Outro limiar máximo de falsos positivos é atribuído à cadeia toda. As instâncias de treinamento são fornecidas ao AdaBoost que itera tanto quanto o necessário para alcançar os limiares do nó. Quando isso ocorrer, tal nó será inserido na cadeia que será testada para verificar se alcançou o seu limiar de falsos positivos. Se o limite foi alcançado, a cadeia está pronta; senão, uma quantidade pré-determinada das instâncias negativas mal classificadas pela cadeia durante o teste serão usadas para impulsionar o próximo nó. Note que o conjunto de instâncias positivas permanece o mesmo, enquanto o de instâncias negativas muda. Através desse processo os nós mais próximos do fim

da cadeia serão treinados com instâncias "mais difíceis", isso é, que não puderam ser corretamente classificadas pela cadeia pronta até então. Por isso os nós seguintes precisarão de mais classificadores fracos para atingir os limiares do nó, melhorando a precisão da cadeia como um todo. Isso é descrito em detalhes em [60].

No trabalho de Viola e Jones [60], cada estágio (nó) dessa cadeia é um classificador forte produzido por impulsionamento. O algoritmo de impulsionamento usado é similar ao já apresentado, porém os pesos iniciais das amostras e a regra de atualização dos pesos apresentam pequenas diferenças. Sejam $m^+$ e $m^-$ respectivamente a quantidade de instâncias positivas e negativas. Viola e Jones usaram a seguinte atribuição inicial de pesos:

$$D_1(i) = \begin{cases} 0.5/m^+ & \text{se } y_i = +1 \\ 0.5/m^- & \text{se } y_i = -1 \end{cases} . \tag{2.2}$$

Como $m^+ < m^-$, os pesos das instâncias positivas serão maiores que os das instâncias negativas. O objetivo disso é instruir o aprendiz fraco a buscar nas primeiras iterações um classificador fraco que cometa pouco ou nenhum erro de classificação das instâncias positivas, ainda que possa classificar mal várias instâncias negativas.

### 2.2.1 Processamento de imagem

Antes de classificar a imagem, o detector pode processá-la para reduzir as variações indesejadas. Viola e Jones propuseram a normalização por variância de cada subjanela para reduzir os efeitos de variação da luminosidade antes de fornecê-las ao classificador [61], mas eles não explicitaram tudo o que fizeram. Lienhart e Maydt [32] cobriram essa lacuna aplicando a seguinte equação:

$$l' = \frac{l - \mu_x}{2\sigma_x} \tag{2.3}$$

Na Equação 2.3, $l'$ é o valor do pixel que será usado pelo classificador, $l$ é a intensidade de um pixel da subjanela, $\mu_x$ é a média da intensidade dos pixels da subjanela $x$, e $\sigma_x$ a variância das intensidades de pixels da subjanela. A seção 2.3.1 explica como é possível realizar o cálculo rápido desses valores de cada subjanela.

### 2.2.2 Pós-processamento dos resultados

É claro que, por simplicidade de visualização e análise dos resultados de detecção, deseja-se que o detector produza apenas uma região para cada objeto. Porém, o que comumente se observa é a múltipla detecção de um mesmo objeto. Isso ocorre pois o classificador é insensível a pequenas variações de translação e escala, e a janela detectora, enquanto itera pela imagem assumindo tamanhos diversos, pode conter o objeto de interesse em posições ligeiramente diferentes. O resultado é visto na Figura 2.2.

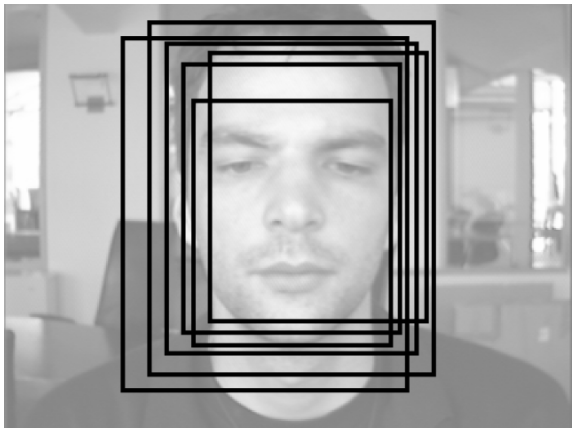

Figura 2.2: Cada retângulo preto representa uma detecção feita em regiões circunvizinhas à face. Note, além das diferenças de posição, as diferenças de escala. Imagem retirada da base BioId [24].

Para tratar dessa questão, Viola e Jones [60] sugerem a "integração" de detecções, um procedimento no qual detecções sobrepostas são, primeiro, colocadas num mesmo conjunto. Então, forma-se uma única detecção por conjunto com a média das posições dos vértices das detecções nele contidas. Os autores afirmam que, em alguns casos, essa abordagem também reduz a quantidade de falsos positivos.

## 2.3 Classificadores fracos

Até aqui, os classificadores ditos fortes foram bastante mencionados. Esta seção descreve os classificadores **fracos** que compõem os classificadores fortes.

Haar *wavelet* é uma função proposta por Alfred Haar [21] para transformar um sinal numa forma mais simples e representativa para certas análises. Uma tradução para o inglês de seu trabalho está disponível em [67]. A transformada de Haar simplesmente converte um sinal qualquer em valores discretos de acordo com certos limiares definidos na função da transformada. Efetivamente, isso converte um sinal qualquer em uma onda quadrada.

Papageorgiou e colaboradores [39] criaram um extrator de características que usa Haar *wavelets* para codificar diferenças locais de pixels em imagens. Essa **característica baseada em Haar *wavelet*** é um valor em $f(w) \in \mathbb{R}$ obtido da soma ponderada das intensidades de pixels contidos em $d$ regiões retangulares da Haar *wavelet*, onde cada região associa-se com um peso $v \in \mathbb{R}, v \neq 0$. Em geral, os pesos das características baseadas em Haar *wavelet* somam zero, e são proporcionais à quantidade de pixels contida em cada retângulo a que se referem. Considerando $w$ uma Haar *wavelet*, $r$ uma região retangular de $w$, e $l$ os pixels contidos em $r$, é possível escrever:

$$f(w) = \sum_{i=1}^{d} v_i(\sum_{l \in r_i} l). \tag{2.4}$$

Os **classificadores fracos** propostos por Viola e Jones em [61] usam tais características, e nada mais são que uma função $h(x, f(w), p, \theta) \mapsto \{-1, +1\}$, onde o valor $+1$ significa que o objeto pertence à classe de interesse, e $-1$ o oposto. Considerando $p \in \{-1, +1\}$ a polaridade (ou paridade), $\theta$ um limiar, e $x$ uma subjanela da imagem, estabeleceu-se:

$$h(x, f, p, \theta) = \begin{cases} +1 & \text{se } pf(x) < p\theta \\ -1 & \text{caso contrário} \end{cases} . \tag{2.5}$$

$p$ simplesmente afeta a orientação da comparação. Por vezes, utiliza-se 0 ao invés de $-1$. Detalhes sobre a atribuição de valores a $p$ e $\theta$ estão na seção 2.4.

Desde sua proposição, pesquisadores tentam melhorar as características baseadas em Haar *wavelets*. Viola e Jones [61] estenderam o conjunto originalmente

proposto por Papageorgiou e colaboradores, e apresentaram um método de calcular qualquer característica em tempo constante. Lienhart e Maydt [32] propuseram características rotacionadas em 45°, também calculadas em tempo constante. Li et al. [31], apresentaram uma forma mais geral de descrever as características, apresentando os retângulos disjuntos e sua importância no problema de detecção de faces rotacionadas. Viola e Jones [25] propuseram características "diagonais". A figura 2.3 mostra alguns exemplos de tais características. Zhang e Zhang [66] coletaram outras pesquisas sobre esse tópico.

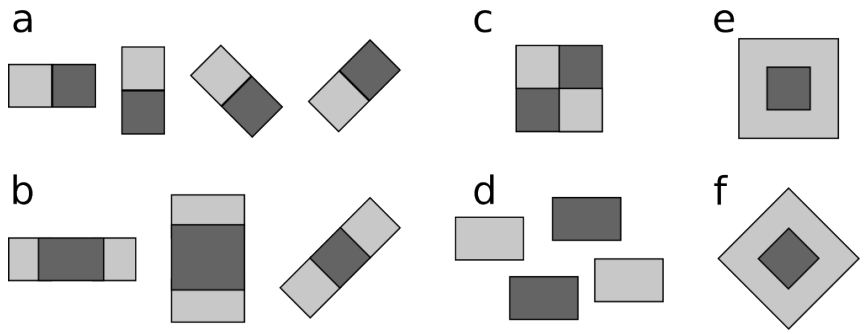


Figura 2.3: Exemplos de características baseadas em Haar *wavelets*: (a) borda; (b) linha; (c) característica quadridimensional proposta em [39]; (d) retângulos disjuntos propostos em [31]; (e) e (f) características centro-cercanias de [32]. As regiões escuras têm pesos diferentes das mais claras.

### 2.3.1 Cálculo rápido dos valores das características

Conforme mencionado na seção 2.2, a detecção de faces numa imagem envolve a extração de um grande número de subjanelas, seguido do exame de cada uma delas pelo classificador, que é uma cadeia de classificadores fortes (seções 2.1.2 e 2.2), por sua vez compostos de vários classificadores fracos cujas saídas dependem de cálculos realizados sobre cada pixel de cada subjanela. Está claro que o detector custará a produzir resultados a menos que o cálculo das características seja muito rápido. Com o uso de uma tabela de soma de áreas (do inglês *summed area table*) [10], é possível calculá-las em tempo constante.

Uma tabela de soma de áreas é uma matriz $SAT(\gamma+1, \tau+1)$ produzida a partir

de outra matriz $A(\gamma, \tau)$ tal que:

$$SAT(a+1, b+1) = \sum_{a' \leq a, b' \leq b} A(a', b') \text{ e } SAT(a, 0) = SAT(0, b) = 0. \quad (2.6)$$

Se $A$ for a imagem, com apenas quatro consultas à $SAT$, é possível calcular a soma de pixels $L$ de qualquer retângulo $r = (a, b, \gamma', \tau')$ contido na imagem:

$$L = SAT(a, b) + SAT(a + \gamma', b + \tau') - SAT(a, b + \tau') - SAT(a + \gamma', b) \quad (2.7)$$

A figura 2.4 facilita a compreensão da equação 2.7.

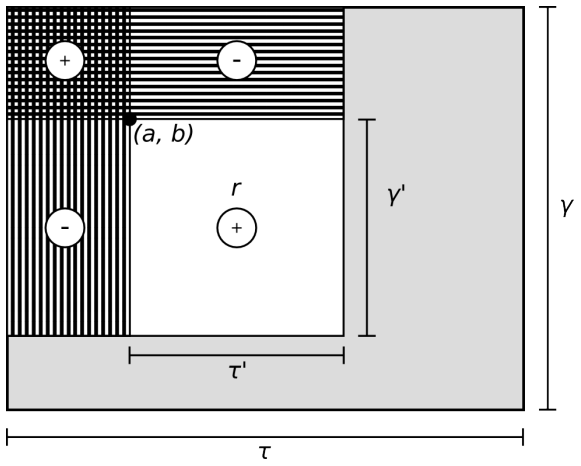


Figura 2.4: A soma das intensidades dos pixels contidos dentro de uma área retangular $r$ é feita conforme a equação 2.7.

Note que, conforme apresentado na seção 2.2.1, são necessárias também a média e a variância das intensidades dos pixels. Se a média é imediata com uma $SAT$, o cálculo da variância necessita do quadrado da soma dos pixels cuja rápida obtenção é possível através de uma tabela de quadrados das somas de áreas $SSAT$, definida da seguinte forma:

$$SSAT(a+1, b+1) = \sum_{a' \leq a, b' \leq b} A(a', b')^2 \text{ e } SSAT(a, 0) = SSAT(0, b) = 0. \quad (2.8)$$

Tais tabelas são construídas quando o detector recebe uma nova imagem, e são mantidas até o fim de seu esquadrinhamento. Esse técnica, proposta por Viola e Jones [61], acelera significativamente o treinamento e a operação do detector.

## 2.4 Aprendiz fraco

Conforme mencionado na seção 2.1.2, o aprendiz fraco é o algoritmo responsável por produzir um classificador fraco tal que (1) produza o menor erro ponderado $\epsilon = Pr_{i\sim D}[h(x_i) \neq y_i]$; e (2) $\epsilon < 0.5$. É fácil imaginar um algoritmo ingênuo para essa tarefa: basta testar todos os classificadores fracos disponíveis contra cada instância, verificando se a classificação resultante é a mesma que a do rótulo da instância, somando os pesos das instâncias mal classificadas. Aquele que somar o menor erro é selecionado. Apesar dos aprendizes fracos operarem de forma similar, essa abordagem não permite o aporte de nenhuma informação para o classificador fraco. Tal aporte não é sempre necessário, como é o caso do classificador proposto por Adhikari et al. [1], mas um grande número de classificadores bem-sucedidos precisam de certos parâmetros para serem úteis. O problema é que certos parâmetros do classificador fraco podem assumir um valor dentre infinitas possibilidades. Como se observa na seção 2.3, o classificador baseado em Haar *wavelets* usado por Viola e Jones e nesta dissertação, possui o parâmetro $\theta \in \mathbb{R}$ que se enquadra nessa situação.

É notória a ausência de informação sobre os aprendizes fracos na literatura de detectores de objetos baseados no arcabouço de Viola e Jones, e até mesmo esses pesquisadores se omitiram de detalhar isso no trabalho publicado em 2001 [61], fazendo-o superficialmente somente na versão de 2004 [60]. Por causa disso, alguns trabalhos ([1] e [45]) declaram que esse é um problema nada trivial, e propõem abordagens alternativas.

Essa seção apresenta o aprendiz fraco utilizado neste trabalho, e, entende-se, em [60]. Trata-se de um algoritmo de construção de árvores decisórias de 1 nível (no inglês *decision stump*). A abordagem apresentada aqui é adaptada de [52].

Na iteração $t$, para produzir o classificador fraco $h_t$ com parâmetros $p$ e $\theta$,

o aprendiz fraco recebe do algoritmo de impulsionamento as instâncias rotuladas $(x_1, y_1), \ldots, (x_m, y_m)$ e seus pesos $D_t(i), i \in \{1, \ldots, m\}$. O algoritmo itera por cada característica baseada em Haar *wavelet* $f_k, k \in \{1, \ldots, K\}$ usando-a para formar um classificador fraco $h_k$ com parâmetros que produzam o menor erro ponderado $\epsilon_k$ possível. O melhor classificador $h_k$ é aquele com menor $\epsilon_k$, e o melhor $h_k$ será atribuído a $h_t$. Como o processo iterativo sobre as características $f_k$ é trivial, resta apresentar a atribuição de valores a $p$ e $\theta$ para um classificador fraco baseado numa característica específica.

Esse algoritmo se embasa no seguinte fato: apesar de haver infinitos valores para $\theta$, há um número finito de intervalos formados pelos valores $f_k(x_i)$ se eles estiverem em ordem crescente. A aplicação de tal critério de ordenação sobre a tupla $(f_k(x_i), D(i), y_i)$ permite a descoberta, em tempo linear, do intervalo $(f_k(x_{i-1}), f_k(x_i))$, onde qualquer valor de $\theta$ provoca o erro ponderado mínimo. Durante a passagem pela lista de tuplas, quatro somas são mantidas: $T^+$, o total de pesos de amostras positivas; $T^-$, o total de pesos de amostras negativas; $S^+$, a soma dos pesos das instâncias positivas até o $i$-ésimo valor sob exame; e $S^-$, a soma dos pesos das instâncias negativas até $i$-ésimo valor. Note que $T^+ - S^+$ representa a soma de pesos positivos existentes após o $i$-ésimo item da lista, e, analogamente, $T^- - S^-$ a soma de pesos de instâncias negativas. Assim, na $i$-ésima iteração, o erro do classificador fraco cujo $\theta \in (f_k(x_{i-1}), f_k(x_i))$ é:

$$\epsilon = min\{S^+ + (T^- - S^-), S^- + (T^+ - S^+)\}. \tag{2.9}$$

Além disso, tem-se:

$$p = \begin{cases} +1 & \text{se } T^+ - S^+ \geq T^- - S^- \\ -1 & \text{caso contrário} \end{cases} \tag{2.10}$$

Para entender a atribuição de valores a $\theta$ e $p$, note que $S^+ + (T^- - S^-)$, visto na equação 2.9, equivale à soma dos pesos das instâncias que seriam classificadas pela equação 2.5 como falsos positivos se $\theta \in (f_k(x_{i-1}), f_k(x_i))$ e se $p = -1$. Raciocínio

**Algoritmo 2:** Formação de $h_k$ com parâmetros $p$ e $\theta$ a partir de $f_k$.

**Dados**: Tuplas $(f_k(x_i), D_t(i), y_i), i \in \{1, \ldots, m\}$, em ordem crescente de valores de $f_k(x_i)$, onde $D_t(i)$ é o peso atribuído pelo algoritmo de impulsionamento à instância $x_i$ com rótulo $y_i$.

**Iniciação**

$T^- = \sum_{i:y_i=-1} D_t(i)$ e $T^+ = \sum_{i:y_i=+1} D_t(i)$

$S^+ = S^- = 0$

$\epsilon_{minimo} = min\{T^-, T^+\}$

$\theta = -\infty$

$p$ atribuído conforme a Equação 2.10

**para** $i = 1, \ldots, m$ **hacer**

  **Se** $y_i = +1$ **então**

    $S^+ \leftarrow S^+ + D_t(i)$

  **Fim se**

  **senão**

    $S^- \leftarrow S^- + D_t(i)$

  **Fim se**

  $\epsilon = min\{S^+ + (T^- - S^-), S^- + (T^+ - S^+)\}$

  **Se** $\epsilon < \epsilon_{minimo}$ **então**

    Escolha $\theta \in \begin{cases} [f_k(x_i), f_k(x_{i+1})) & \text{se } i < m. \\ [f_k(x_m), +\infty) & \text{se } i = m. \end{cases}$

    Recompute $p$ conforme a Equação 2.10.

    $\epsilon_{minimo} \leftarrow \epsilon$

  **Fim se**

**fin para**

**Retorne** classificador fraco $h_k(x, f_k, p, \theta)$.

análogo procede para $S^- + (T^+ - S^+)$, com $p = +1$. A Figura 2.5 facilita a visualização do que ocorre. O algoritmo 2 descreve a rotina do aprendiz fraco para formar um classificador fraco $h_k$ a partir de um classificador fraco $f_k$.

O aprendiz fraco completo tem complexidade $O(Km)$ [60]. Há propostas de melhorias para seu desempenho em conjunto com o AdaBoost [64].

Essa abordagem se assemelha à análise da característica de operação do receptor (do inglês, *receiver operating characteristic*, ROC) [13].

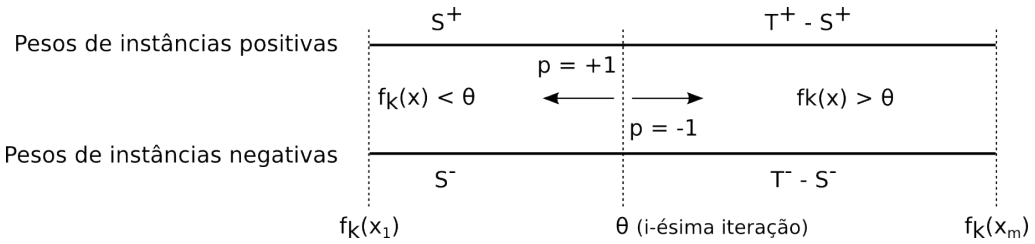


Figura 2.5: Visualização do que ocorre durante uma iteração $i$ do algoritmo 2.

## 2.5 Análise de componentes principais

Nesse trabalho, utiliza-se a análise de componentes principais (ACP) para a atribuição de novos pesos $v$ às Haar *wavelets*. A proposição de uso dessa técnica é feita no capítulo 4, e esta seção apenas a revisa rapidamente pois a literatura sobre o assunto é abundante.

A ACP foi introduzida por Karl Pearson [42]. Sua aplicação num conjunto de medidas multivariadas resulta em novas variáveis não correlatas que são a combinação linear das variáveis originais. Geometricamente, é possível interpretar o resultado da ACP como uma rotação do eixo de variáveis original em novos eixos (também chamados de componentes) ortogonais, cada um associado a uma medida de variação dos dados originais [12].

Um dos objetivos dessa transformação é selecionar, das medidas multivariadas originais, um conjunto menor de variáveis que represente as medidas originais de forma satisfatória. Em outras palavras, deseja-se eliminar variáveis redundantes, que trazem a mesma informação que outras dentro do contexto duma análise particular. Isso pode ser necessário, por exemplo, quando há tantas variáveis que o problema é computacionalmente intratável. Ainda que isso efetivamente destrua informação, o sucesso de ACP em múltiplas aplicações, tais como análises estatísticas, extração de características e compressão de dados [22], é um indicativo de que há circunstâncias em que tal perda é tolerável.

A intuição sobre a ACP pode vir através de um exemplo baseado na interpretação geométrica duma distribuição normal bivariada, como o apresentado na Figura

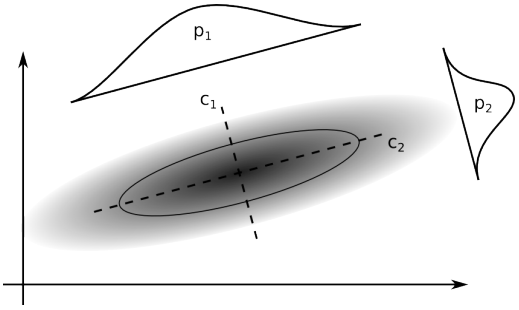


Figura 2.6: Componentes principais $c_1$ e $c_2$ de uma distribuição normal bivariada arbitrária. As distribuições $p_1$ e $p_2$ são oriundas da projeção dos pontos dessa distribuição normal na direção das componentes principais. A linha elíptica apresentada mostra uma secção equiprovável da distribuição.

2.6. Os dados dessa distribuição podem ser projetados na direção de infinitos vetores (unitários), resultando em distribuições monovariadas, cada qual com sua variância. Ao menos uma das projeções terá variância máxima e outra a variância mínima.

Ao observar as projeções das medidas bivariadas na figura 2.6, alguém pode concluir que a massa de dados poderia ser representada por uma das distribuições monovariadas resultantes da projeção dos dados na direção de uma das componentes. Nesse caso, a seleção da componente depende do problema em tratamento.

Em Hyvarinen et al. [22], a derivação de componentes principais por maximização da variância é feita em detalhes. Para este trabalho, basta saber que a solução para um problema de ACP é dada pelos autovetores e autovalores obtidos a partir da matriz de covariância da distribuição original.

# 3 ESTADO DA ARTE

Neste capítulo são revisados alguns trabalhos recentes relacionados com o arcabouço de Viola-Jones e com as propostas que serão detalhadas no capítulo 4. De fato, é sobre alguns dos trabalhos mencionados neste capítulo que as hipóteses serão formuladas.

## 3.1 Impulsionamento

Apesar dos algoritmos de impulsionamento serem muito importantes para esta dissertação, eles não são o assunto onde se concentram as principais discussões. Por essa razão, esta seção, dedicada à revisão de pesquisas recentes sobre impulsionamento, faz apenas um breve apanhado de alguns trabalhos recentes interessantes visando ilustrar o poder, a flexibilidade, a aplicabilidade e algumas limitações deste método de aprendizagem supervisionada.

Ramos [43] aplica e compara diversas técnicas de classificação para avaliar solicitações de indenização referentes a seguros agrícolas feitos pelo governo para proteger agricultores familiares. Seu objetivo é automatizar a detecção de potenciais solicitações indevidas, deixando para operadores humanos uma pequena parcela do trabalho. Após uma série de experimentos, o autor obteve bons resultados com diversos classificadores, porém o melhor resultado foi obtido com uma máquina de vetor suporte (do inglês *support vector machine*, SVM). O algoritmo de impulsionamento usado foi o AdaBoost.M1 [16].

Chaves [7] apresenta uma aplicação do AdaBoost no desenvolvimento de um classificador que operaria num sistema embarcado dotado de múltiplos sensores para detectar adulteração em etanol automotivo. O protótipo desse sistema opera usando um Arduino Mega e, além de conseguir operar sem problemas nesse dispositivo, alcançou uma taxa de classificação correta bastante alta.

Uma outra aplicação interessante de impulsionamento foi na área de física de

alta energia, onde se deseja identificar eventos de oscilação de neutrinos (i.e. eventos em que um tipo de neutrino degrada noutro) de outros eventos. Segundo Roe e colaboradores, testes com o AdaBoost e o $\epsilon$-Boost [19], ambos usando árvores decisórias como classificadores fracos, resultaram numa melhoria de 20% a 80% de desempenho, se comparado com o classificador usado anteriormente, baseado em redes neurais artificiais [46].

Khoshgoftaar e colaboradores procuraram entender melhor as limitações do impulsionamento. Em [29], compararam o desempenho do SMOTEBoost [8] e do RUSBoost [55] com dois algoritmos de *bagging* chamados de *Exactly Balanced Bagging* e *Roughly Balanced Bagging* em bases de treinamento ruidosas, isso é, cujas amostras possuem muitos erros; e desbalanceadas, isso é, quando há muito mais instâncias de uma classe do que de outra. De fato, os dois algoritmos de impulsionamento foram projetados especificamente para reduzir problemas de desbalanceamento de amostras de treinamento. Após uma grande quantidade de testes, os autores concluem que *bagging* tende a desempenhar melhor que impulsionamento, principalmente quando os dados apresentam muitos erros, mas quando a base de treinamento está livre de erros o impulsionamento parece funcionar melhor mesmo quando há desbalanceamento.

Diversas pesquisas adicionais e importantes que envolvem o uso de impulsionamento na detecção e rastreamento de objetos em imagens e vídeo foram sumarizadas por Zhang e Zhang [66], que tabularam convenientemente um sumário dos desafios enfrentados por muitos pesquisadores. Um trabalho similar e complementar, porém mais antigo, é o de Yilmaz [65].

## 3.2 Detectores de faces

Apesar do termo "detector" ser usado sem muitas formalidades, neste trabalho um detector tem funções específicas. Ele contém estruturas de dados e algoritmos responsáveis pelo processamento e esquadrinhamento da imagem; extração e processamento da subjanela antes de entregá-la ao classificador; coleta e processamento

dos resultados de classificação. Há pesquisadores dedicados a melhorar estas funções e esta seção apresenta alguns de seus trabalhos.

Um resultado interessante foi obtido por Ishii et al. [23], que trabalharam com detecção e rastreamento de faces. Usando uma câmera e uma placa dotada dum circuito FPGA (do inglês, *field-programmable gate array*) capaz de capturar 2000 quadros coloridos de 8 bits com resolução de $512 \times 512$ por segundo, e uma placa com GPU, os pesquisadores conseguiram detectar e rastrear uma face consistentemente numa taxa de 500fps.

O detector de faces operante nesse hardware segue o arcabouço de Viola e Jones, com algumas diferenças. Além de diversas operações executarem em paralelo, como o cálculo da imagem integral, ou o valor das características baseadas em Haar *wavelets*, os pesquisadores usaram (1) a variação dos pixels entre o quadro atual e o quadro prévio para priorizar a área de busca por uma face e acelerar a detecção; e (2) as informações de cor para excluir eventuais falsos positivos resultantes da classificação.

Também interessados na aceleração da detecção e rastreamento de faces, Ranjan e Malik mostram como é possível através dum novo conjunto de rotinas de processamento e esquadrinhamento da imagem, e de técnicas de paralelização, acelerar a detecção de faces em até 70% [44]. Os pesquisadores utilizaram duas bibliotecas da Intel: a *Integrated Performance Primitives* e a *Threading Building Blocks*.

Uma importante otimização envolve a separação do esforço de detecção em rastreamento de faces já detectadas e em detecção de novas faces. Estimando o fluxo ótico do quadro atual em relação ao anterior, é possível estimar a nova posição de uma face já detectada e confirmar sua presença utilizando o detector baseado no arcabouço de Viola e Jones na área estimada. Áreas onde houve movimento, mas não há faces se tornam candidatas para detecção de novas faces.

Uma das contribuições importantes dos autores sobre a paralelização da detecção de faces é também bastante simples: cada processador pode ficar responsável por esquadrinhar a imagem com uma janela de detecção de tamanho diferente. Como o processo de detecção com a janela de um tamanho é completamente independente

do processo de detecção com a janela de outro tamanho, a paralelização da forma proposta é imediata.

Wong e colaboradores [63], implantaram um detector baseado no arcabouço de Viola e Jones em um dispositivo de pequeno porte, de baixa capacidade de processamento e dotado de pouca memória. Para tanto, propuseram diversas modificações, tais como reduzir a imagem após esquadrinhá-la por completo, ao invés de aumentar o tamanho da janela de detecção; representar a imagem com apenas os 4 bits mais significativos; usar um novo esquema de cálculo de imagens integrais que também reduzia a quantidade de características disponíveis; e compactar a representação de cada classificador fraco em apenas 33 bits. Muitas dessas propostas implicam no sacrifício da taxa de detecção, que ficou ao entorno de 91%.

Numa linha de esforços diferente, Jun e Kim [26], dentre diversas contribuições, apresentaram uma técnica de acumulação de evidências visando a redução de falsos positivos. A ideia vem da observação de que detectores precisos comumente produzem regiões sobrepostas (como visto na seção 2.2.2), em diversas escalas, ao entorno duma face real, porém quando a detecção é falsa, há poucas sobreposições.

Jun e Kim sugeriram o cálculo de um indicador $C(x, y)$, onde $x$ e $y$ representam a posição horizontal e vertical dentro da imagem de cada região detectada, independente de escala, no qual se adiciona a saída de cada classificador fraco de cada estágio da cascata classificadora. Falsos positivos são eliminados comparando todo $C(x, y)$ com um limiar. Eles experimentaram essa técnica com um classificador fraco que **não** se baseia em Haar *wavelets*. Os autores não reportaram resultados do emprego isolado dessa abordagem, mas afirmam que a taxa de falsos positivos foi reduzida significativamente.

## 3.3 Classificadores fracos

O estudo de características baseadas em Haar *wavelets* não envolve somente a expansão do conjunto de características disponíveis (seção 2.3. Há, por exemplo, tentativas de entender quais são as mais importantes tornando possível a diminui-

ção do conjunto características [62]. Pesquisadores também investem em meios de selecioná-las mais rapidamente [64]. Investe-se também sobre o próprio AdaBoost para que ele selecione características usando conhecimentos do domínio de aplicação [5]. Há também outros tipos de características que podem ser usadas no lugar de Haar *wavelets*, como padrões binários locais (do inglês, *local binary patterns*) [26]. Nessa seção, revisaremos o trabalho feito nessa área.

Castrillón e colaboradores [6] elaboraram uma comparação entre detectores de faces completos e de partes da face: olhos, nariz, boca. Nenhum classificador de orelha foi considerado. No total, testaram o desempenho e a precisão de 35 classificadores baseados no trabalho de Viola e Jones [60] e de Lienhart e Maydt [32] retirados de 16 fontes diferentes contra as bases de dados da CMU e Yale. Para determinar os melhores classificadores, a área sobre suas curvas ROC (*receiver operating characteristic*) foram comparadas. Foram estabelecidos critérios para determinar o sucesso da detecção de cada objeto de interesse.

As bases de faces têm características distintas. A CMU está subdividida em 4 conjuntos nomeados test, new-test, low-res e rotated com imagens de faces frontais, em perfil, rotacionadas e em baixa resolução. A base Yale contém imagens frontais de indivíduos em maior resolução. Claramente, a base CMU é de classificação mais difícil que a de Yale.

Dois tipos de experimentos foram feitos. O primeiro testou independentemente os detectores de faces frontais, em perfil, busto, olho direito, olho esquerdo, par de olhos, nariz e boca. O artigo apresenta os detectores mais precisos para cada caso, e faz as seguintes observações interessantes:

1. os detectores de faces trazem mais verdadeiros positivos na base CMU que qualquer detector de partes da face. Parte disso se deve à dificuldade dos detectores de partes da face operarem em imagens de menor resolução;

2. a detecção de par de olhos traz menos resultados verdadeiros positivos e menos falsos positivos do que a detecção de cada olho individualmente. Isso é

particularmente perceptível quando a resolução da imagem é baixa;

3. detectores de boca e nariz são muito precisos quando testados na base de dados Yale.

O segundo tipo de experimento detectava os elementos internos uma vez que a face fora detectada. Os autores compararam três abordagens: (1) buscar livremente os elementos dentro da face; (2) buscar elementos dentro de regiões de interesse; (3) ampliar o tamanho do detector e usá-lo para buscar nas mesmas regiões de interesse. A abordagem (2) trouxe melhores resultados que a (1). A abordagem (3), comparada com (2) agregou mais verdadeiros positivos aos resultados, mas também aumentou um pouco a quantidade de falsos positivos.

Landesa-Vázquez e colaboradores [30] propuseram uma extensão do conjunto básico de características propostas por Viola e Jones. Inspirados em estudos da fisiologia da visão humana, observaram que (1) a polaridade do contraste entre áreas vizinhas do rosto humano é uma informação usada pelo cérebro para processar partes internas do rosto humano; (2) já os contornos do rosto aparentam estar codificados no cérebro de forma insensível à polaridade do contraste; e (3) no conjunto de características baseadas em Haar *wavelets* propostas por Viola e Jones há somente aquelas sensíveis à polaridade do contraste (como na equação 2.5).

Os autores propuseram um conjunto de classificadores fracos baseados numa característica apolar que modelaram considerando as questões acima. Em termos práticos, adicionaram uma função de classificação que considera o valor absoluto da característica baseada em Haar *wavelet*, isso é, dado $f_{\text{apolar}}(x) = |f(x)|$, tem-se:

$$h_{\text{apolar}}(x, f_{\text{apolar}}, p, \theta) = \begin{cases} 1 & \text{se } p f_{\text{apolar}}(x) < p\theta \\ 0 & \text{caso contrário} \end{cases} . \tag{3.1}$$

Com o conjunto original de características aplicado aos dois classificadores fracos polar e apolar, os autores treinaram um classificador forte de faces em cascata, seguindo os moldes de Viola e Jones. O resultado não apresentou nenhuma mudança sensível na precisão do detector, mas houve uma redução significativa na quantidade

de classificadores fracos inseridos no classificador forte, assim como uma redução na quantidade média de classificadores acionados ao se detectar faces. Os autores mostraram também que os classificadores polares e apolares são acionados como descrito pelos estudos fisiológicos humanos.

Dembski [11] realiza experimentos com classificadores propostos por Lienhart e Maydt, mas seu intento é verificar se é possível perceber padrões na contribuição das características para o classificador final, e daí concluir se há um conjunto de características mais útil que outros.

Dembski testa seus classificadores na base de dados do MIT ([32] contém testes na base CIF, enquanto [60] testaram em uma base MIT+CMU). Os resultados obtidos sugerem que as características em forma de linha horizontais ou rotacionadas resultam em melhor erro de generalização do que as características do tipo centro-cercanias e do tipo borda. Ele comparou também as características horizontais com as rotacionadas e notou que aquelas generalizam ligeiramente melhor que estas. Por fim, ele percebeu que as características com área maior que 50 pixels generalizam melhor que características com área acima de 25 pixels.

Apesar dessas conclusões, as diferenças apresentadas são pequenas e podem estar dentro de uma margem de erro. Também não há detalhes sobre a composição dos classificadores fortes: está ausente, por exemplo, a quantidade de classificadores fracos que o compõem, assim como a participação de cada tipo de classificador. Apesar dos testes feitos, não há informação sobre o uso das características durante a classificação para, por exemplo, rejeitar uma amostra e qual a contribuição dos classificadores de vários tipos nesse sentido. Além disso, o trabalho está limitado ao contexto de classificação de faces. A detecção, tarefa que envolve o esquadrinhar de imagens em busca de faces, não foi testada. De fato, os resultados referem-se a um classificador monolítico, enquanto detectores de faces de Viola e Jones e de Lienhart e Maydt adotam uma topologia em cascata.

Baumann e colaboradores [5] também se inspiraram em [60] e propuseram uma modificação no AdaBoost para aproveitar a simetria da face humana durante o trei-

namento. Em seus experimentos, diminuíram em quase 40% o tempo de treinamento do classificador forte, além de promover um ligeiro aumento nas taxas de detecção (2.5% no melhor caso). Nenhuma informação sobre falsos positivos foi dada. Em seus testes, usaram a base de faces da AT&T e treinaram apenas classificadores monolíticos.

A ideia que apresentaram consiste em selecionar dois classificadores fracos baseados em Haar *wavelets* por iteração do AdaBoost em vez de apenas um, como ocorre tipicamente. O primeiro classificador é escolhido usando o procedimento normal proposto por Freund e Schapire [15]. Então, uma Haar *wavelet* simétrica à primeira é escolhida, e simetricamente posicionada na janela de detecção, ainda que sua localização final esteja sujeita a uma busca dentro de um pequeno limite nas direções verticais e horizontais. O erro ponderado do classificador simétrico é calculado como o erro ponderado de qualquer busca por classificador fraco no AdaBoost. O segundo classificador somente será incluído se seu erro for no máximo 2.5% maior que erro do primeiro classificador.

A aceleração do treinamento se deve a essa escolha simultânea de duas características, onde o espaço de busca da segunda é limitado a um único classificador dentro de uma pequena área. Os autores não deram detalhes sobre a frequência de rejeição do segundo classificador.

Em seguida, o algoritmo procede para a uma rotina que atualiza os pesos das instâncias de treinamento que considera independentemente os erros de classificação de ambos os classificadores fracos. Por fim, o peso dos classificadores também é calculado independentemente, mas eles são usados em conjunto pelo classificador forte. O artigo não deixa claro o que ocorre se o classificador simétrico for rejeitado, por isso supõe-se que as rotinas normais de atualização dos pesos das instâncias de treinamento e do peso do classificador fraco sejam usadas.

No caso da face humana, a linha de simetria é a vertical, mas é possível explorar também a simetria horizontal em outros problemas. De fato, os autores sugerem que há certos objetos de interesse que possuem várias linhas de simetria que poderiam

ser exploradas estendendo essa abordagem. Encerram destacando a aplicabilidade desse método a variantes do AdaBoost.

Vural et al. [62] apresentam 18 características baseadas em Haar *wavelets* com formas bastante curiosas. Tais características podiam ser rotacionadas em ângulos de 30 deg, o que totalizava um conjunto de 126 características diferentes. Usando apenas essas características em seus experimentos, os autores mostraram bons resultados de classificação e uma redução significativa do tempo de treinamento, consequência do uso de menos classificadores fracos durante o impulsionamento.

Uma feito interessante desse trabalho é a introdução de um método automático para montagem das características combinado com um procedimento de teste e mensuração do potencial classificatório de cada característica. Infelizmente, há pouca explicação sobre como isso é feito.

Em [9] combina-se segmentação a partir da análise do histograma de uma imagem com uma nova Haar *wavelet* composta de 5 retângulos em forma de cruz dedicado à detecção de olhos em imagens segmentadas. O valor da característica é calculado contando a quantidade de pixels dentro do segmento de interesse, ao invés de considerar a intensidade de cada pixel.

Pavani e colaboradores [41] argumentam que os pesos tipicamente atribuídos para cada parte retangular duma Haar *wavelet* resultam numa característica subótima. Para isso, primeiro definem um espaço vetorial de valores obtidos de cada um dos retângulos da Haar *wavelet* que compõem as características baseadas em Haar *wavelets* duma imagem. Esse vetor é multiplicado matricialmente pelos pesos tipicamente atribuídos a ele durante o cálculo da característica, e será sobre um plano paralelo ao vetor resultante dessa multiplicação que a característica será projetada. Os pesos tradicionalmente fixados para cada parte das Haar *wavelets* pouco ajudam na distinção entre o que é parte do objeto de interesse, e o que não é.

Para encontrar os pesos ótimos de cada característica, o autor sugere três métodos diferentes: busca por força bruta, algoritmos genéticos e análise de discriminante linear de Fisher (ADLF). Os métodos são aplicados para treinar o classificador fraco

e simultaneamente selecionar aquele com o menor erro ponderado.

Nos experimentos os autores usaram pouco mais de 200.000 características diferentes (são 160.000 em [60]) e treinaram uma cascata de classificadores de imagens de faces e de ressonância magnética de corações usando imagens de várias bases de dados disponíveis publicamente. O procedimento de formação de cascata é o mesmo que o apresentado por Viola e Jones. A cascata de classificadores tanto de faces como de corações foram treinados usando o AdaBoost tradicional (o grupo de controle) mais os três métodos de formação de classificadores fracos baseados em Haar *wavelets* propostos. O treinamento com o AdaBoost convencional tomou 2 dias, mas com o ADLF tomou 10 dias, com o algoritmo genético 20 dias e com a busca por força bruta 22 dias.

Os classificadores de faces resultantes foram comparados com as bases MIT + CMU. Os autores também compararam seus resultados com de outros pesquisadores importantes e concluíram que o classificador otimizado com algoritmo genético é o mais acurado. Do ponto de vista do desempenho, o classificador baseado em algoritmo genético é o mais eficiente, pois, em média, rejeita amostras negativas usando menos nós da cascata e usando menos classificadores fracos.

Um ponto de atenção no trabalho de Pavani e colaboradores são os tempos de treinamento dos classificadores fortes. Comparado com o método empregado por Viola e Jones [61], a otimização com ADLF tomou cerca de cinco vezes o tempo, enquanto as atribuições de pesos com algoritmos genéticos e busca forçada tomaram ao menos dez vezes o tempo.

Adhikari et al. [1] observam que Viola e Jones [60] não foram claros ao descrever como os parâmetros $p$ e $\theta$ de um classificador baseado em Haar *wavelets* devem ser atribuídos, portanto propuseram um método alternativo que os dispensa. De fato, a observação de Adhikari procede pois, apesar de todas as pesquisas feitas para esta dissertação, notou-se que este assunto não é mencionado explicitamente nos trabalhos relacionados com impulsionamento de detectores de faces.

Adhikari e colaboradores propõem um classificador que se embasa na teoria

decisória de Bayes. Em sua forma mais básica, tal teoria diz o seguinte: para um objeto com característica de valor $s$, e probabilidades *a posteriori* $P(-1|s)$ e $P(+1|s)$ de pertencer às classes $-1$ ou $+1$ respectivamente, é possível minimizar os erros de classificação decidindo que o objeto pertence à classe $-1$ se $P(-1|s) > P(+1|s)$, e $+1$ caso contrário. De forma mais geral, para tratar dicotomias, pode-se dizer que desejam-se funções discriminatórias $g_Y(\vec{s}), Y = \{-1, +1\}$ que minimizam o risco de classificação, i.e. o custo de se classificar mal [12]. O classificador fraco proposto por Adhikari pode ser descrito da seguinte forma:

$$h(x, f, g^+, g^-) = \begin{cases} -1 & \text{se } g^+(f(x)) < g^-(f(x)) \\ +1 & \text{caso contrário} \end{cases} . \tag{3.2}$$

Segundo os autores, os valores de $f(w)$, obtidos de características baseadas em Haar *wavelets*, quando aplicadas tanto a um conjunto de imagens de traseiras de carros (o objeto de interesse), quanto de um conjunto de imagens de fundos, apresentam uma distribuição gaussiana. Dessa maneira, propõem o uso da função discriminatória quadrática aplicada às distribuições normais [12, pg. 36].

Essa abordagem foi validada experimentalmente na detecção de traseiras de carros. Treinaram um classificador monolítico com 1000 instâncias positivas e 3000 instâncias negativas. Os testes foram feitos contra uma base com 500 instâncias positivas e 500 negativas. Os autores não especificaram os valores de $P_{-1}(s)$ e $P_{+1}(s)$ usados, nem informaram quantas características baseadas em Haar *wavelets* estavam disponíveis para o impulsionamento. O desempenho do classificador forte resultante foi comparado com outros dois classificadores que utilizavam métodos de atribuição de valores aos parâmetros $p$ e $\theta$ considerados típicos. As mesmas bases de instâncias foram usadas. O classificador proposto pelos autores apresentou uma curva ROC bem próxima da classificação perfeita, que é bastante superior aos demais classificadores.

Rasolzadeh et al. [45] também apresentam um classificador que se baseia na teoria decisória de Bayes, e, como Adhikari et al., modelam os valores de $f(w)$ como distribuições estatísticas. A diferença é que Rasolzadeh et al. não aplicaram funções

discriminantes quadráticas para comparar as probabilidades de um valor pertencer a uma instância negativa ou a uma instância positiva, ou seja, as probabilidades de um ponto pertencer a um conjunto ou outro são comparadas diretamente (equação 3.3).

$$h(x, f, P^+, P^-) = \begin{cases} -1 & \text{se } P^+(x) < P^-(x) \\ +1 & \text{caso contrário} \end{cases} . \tag{3.3}$$

Esses pesquisadores modelam as distribuições de valores de $f(w)$ com Gaussianas e histogramas, e argumentam que isso confere aos classificadores fracos baseado em Haar *wavelet* uma quantidade maior de limiares classificatórios, portanto aumentando suas chances de terem melhor precisão. De fato, o classificador fraco apresentado por Papageorgiou [39] e usado por Viola e Jones possui somente $\theta$ como limiar. Devido à forma de sua função de densidade, dois limiares são usados no classificador que modela $f(w)$ com Gaussianas, enquanto há uma quantidade indeterminada de limiares no classificador baseado em histogramas.

Rasolzadeh et al. obtiveram bons resultados nos testes de detecção de pedestres que fizeram com um classificador monolítico dotado de 100 classificadores fracos, superando o desempenho do classificador proposto por Viola e Jones. É interessante observar que o impulsionamento foi feito com apenas 10.000 Haar *wavelets*.

# 4 HIPÓTESES E PROPOSTAS

Em [41], Pavani e colaboradores apresentaram um método de treinamento de classificadores fortes combinado com otimização de Haar *wavelets* que melhora o desempenho de detectores de objetos baseados no arcabouço de Viola e Jones [61]. Os autores experimentaram três algoritmos de otimização de Haar *wavelets* diferentes, e constataram que o melhor resultado foi obtido com o algoritmo genético, porém isso tornou o treinamento bastante longo. Por outro lado, mesmo o método de treinamento mais rápido, que usa análise de discriminantes lineares de Fisher (ADLF), apresentou resultados superiores aos obtidos com a abordagem original de Viola e Jones.

A ADLF é uma técnica que separa duas classes de objetos de forma ótima se os valores das características de ambas as classes se distribuírem Gaussianamente e possuírem as mesmas covariâncias [12, p. 120], e, segundo Pavani et al. [41], os resultados da ADLF costumam ser bons mesmo quando isso não ocorre. Ainda assim, se por um lado os resultados com ADLF nos experimentos de Pavani e colaboradores chegaram a superar outros resultados importantes, por outro foram inferiores a outros métodos que nada assumem sobre as distribuições dos pontos dessas classes (seções 3.3 e 4.1). Os próprios autores apontam duas razões para isso: (1) a distribuição de pontos de faces não tem a mesma covariância que a dos pontos de fundo; e (2) uma das classes de pontos não se distribui Gaussianamente. A análise de uma pequena amostra de dados sobre essas distribuições sugere que ambos os casos ocorrem em muitas Haar *wavelets*. Mais especificamente, (1) ocorre quase sempre, porém (2) ocorre com periodicidade difícil de determinar.

Do que foi possível observar, poucas pesquisas consideram as informações contidas nesse trabalho de Pavani e colaboradores. Os dados contidos lá servem apenas como motivadores para o trabalho desses autores, e seus resultados, apesar de servirem como indicativos, não são conclusivos a respeito da formação dos ECRSs.

Outros trabalhos fizeram uso dessas distribuições, mas também não se aprofundaram a ponto de tentar descrevê-las ([40] e [56] são exemplos).

Esta dissertação apresenta alguns métodos complementares aos apresentados por Pavani e colaboradores para produção de detectores de faces com ajuste de pesos. Intenciona-se explorar as distribuições de pontos no ECRS de modo produzir um detector de objetos ou em menor tempo, ou com maior precisão. Para ir de encontro a esses objetivos, colocam-se hipóteses sobre o ECRS que permitem dar ao problema um tratamento especial, contornando as limitações mencionadas acima. São propostos métodos de otimização dos pesos das Haar *wavelets*, assim como classificadores fracos que procurarão explorar ao máximo as estatísticas dos ECRSs.

## 4.1 O espaço de características retangulares simples (ECRS)

Pavani e colaboradores introduziram o espaço de características retangulares simples (do inglês *single rectangle feature space*). Foi observando este espaço que eles propuseram métodos de otimização de pesos das Haar *wavelets* descritos na seção 3.3. Dada sua importância para esta dissertação, este espaço vetorial é aprofundado aqui.

Seja $S_w$ o ECRS de uma Haar *wavelet* $w$ dotada de $d$ retângulos. Seja $s \in S, s = (s_1, \ldots, s_d), s_i \in \mathbb{R}, 0 \leq s_i \leq 1, i = 1, \ldots, d$. Os valores $s_1, \ldots, s_d$ de cada vetor $s$ nada mais são que as médias das intensidades dos pixels de uma subjanela contidos em cada um dos $d$ retângulos de $w$, divididos pelo valor máximo que um pixel pode assumir (figura 4.1). Se há $m$ instâncias, há $m$ pontos em $S_w$.

Note que $S_w = S_w^+ \cup S_w^-$, onde $S_w^+$ contém apenas os vetores formados a partir de instâncias positivas, isso é, de subjanelas que contém o objeto de interesse (no caso desta dissertação, faces); e $S_w^-$ contém apenas vetores formados a partir de instâncias negativas, isso é, de subjanelas que não contém o objeto de interesse (também chamadas de fundo). Uma descoberta importante de Pavani e colaboradores foi que, dado um certo conjunto de instâncias, $S_w^+$ e $S_w^-$ parecem se distribuir no ECRS de formas muito distintas: aquele conjunto tende a ser compacto e se espalhar

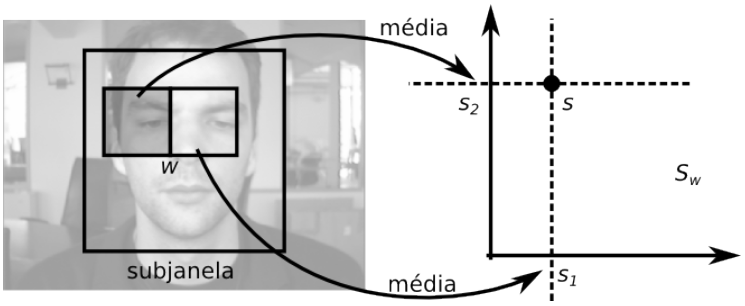


Figura 4.1: Obtenção de $s \in S_w$ com uma Haar *wavelet* com $d = 2$ a partir duma subjanela de uma imagem. Foto retirada de [24].

ao entorno de uma medida central, enquanto este se distribui de várias maneiras, dependendo da Haar *wavelet*. Os pesquisadores não apresentaram nenhuma análise conclusiva a respeito dessas observações, e se ativeram a ilustrar o que notaram com alguns exemplos, além de treinar o classificador conforme apresentado na seção 3.3. Outras pesquisas, tais como [45] e [56], relatam fatos similares, ainda que apresentem poucas evidências sobre eles. A figura 4.2 apresenta um $S_w$ bidimensional hipotético, destacando $S_w^+$ e $S_w^-$.

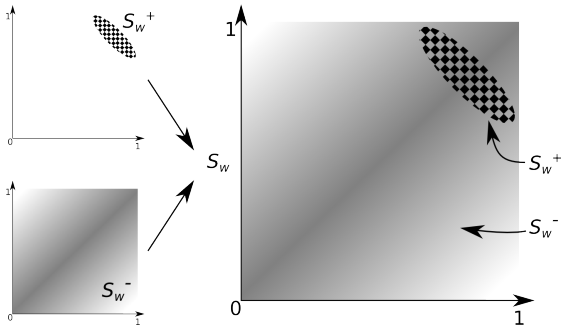


Figura 4.2: Um $S_w$ hipotético cuja Haar *wavelet* associada tem $d = 2$.

Lembrando que cada um dos $d$ retângulos de $w$ está associado a um peso $v_i$, é possível reescrever a equação 2.4 usando os conceitos apresentados nesta seção da seguinte forma:

$$f(w) = \sum_{i=1}^{d} v_i s_i. \tag{4.1}$$

Isso nada mais é que a combinação linear de um vetor de pesos $\vec{v}$ da Haar *wavelet* com o vetor $\vec{s}$ de médias das intensidades dos pixels contidas em cada retângulo. Geometricamente, é possível interpretar $f(w)$ como a projeção de $\vec{s}$ na direção de $\vec{v}$ ([57]). Essa simples observação permitiu a Pavani e colaboradores entenderem melhor o efeito dos pesos das características baseadas em Haar *wavelets* na capacidade discriminatória do classificador fraco. As projeções de $S_w^+$ e $S_w^-$ numa reta paralela a $\vec{v}$, chamadas $f(w, S_w^+)$ e $f(w, S_w^-)$, assumirão formas e parâmetros diversos, podendo misturar os pontos oriundos das projeções com maior ou menor frequência. É claro que, quanto menos parecidos forem os valores de $f(w, S_w^+)$ e $f(w, S_w^-)$, mais fácil será identificar se um ponto qualquer pertence ao conjunto $S_w^+$ ou $S_w^-$, ou seja, maior será o poder discriminatório de um classificador fraco baseado nesse Haar *wavelet*. Infelizmente, os valores tipicamente atribuídos a $\vec{v}$ (múltiplos inteiros de 1 de modo que a soma dos pesos seja 0) não garantem uma boa separação dos conjuntos $S_w^+$ e $S_w^-$. Feitas tais observações, Pavani e colaboradores propõem encontrar o vetor $\vec{v}$ de cada Haar *wavelet* que minimize a chance de pontos de $f(w, S_w^+)$ sobreporem pontos de $f(w, S_w^-)$ [41]. A figura 4.3 exemplifica uma situação em que os pesos típicos de uma Haar *wavelet* bidimensional tem poder discriminatório subótimo.

Um classificador fraco baseado em Haar *wavelets* classifica instâncias verificando se o valor de $f(w)$ está abaixo ou acima de um limiar $\theta$ (equação 2.5). Durante as rodadas de impulsionamento, o aprendiz fraco atribuirá aos parâmetros $p$ e $\theta$ de cada classificador valores que provoquem o menor erro ponderado (seções 2.1.2 e 2.4). Assim, é fácil observar que, dependendo dos pesos que os Haar *wavelets* têm, um ECRS será particionado de forma radicalmente diferente por um mesmo Haar *wavelet*, eventualmente cometendo menos erros. A figura 4.4 ilustra esse efeito.

A partir dessas observações é possível propor algumas técnicas para obtenção de um vetor de pesos ótimo. Dentre as possibilidades, conforme descrito em [41] e revisado na seção 3.3 desta dissertação, três técnicas (ADLF, algoritmos genéticos e busca forçada) já foram testadas com resultados positivos.

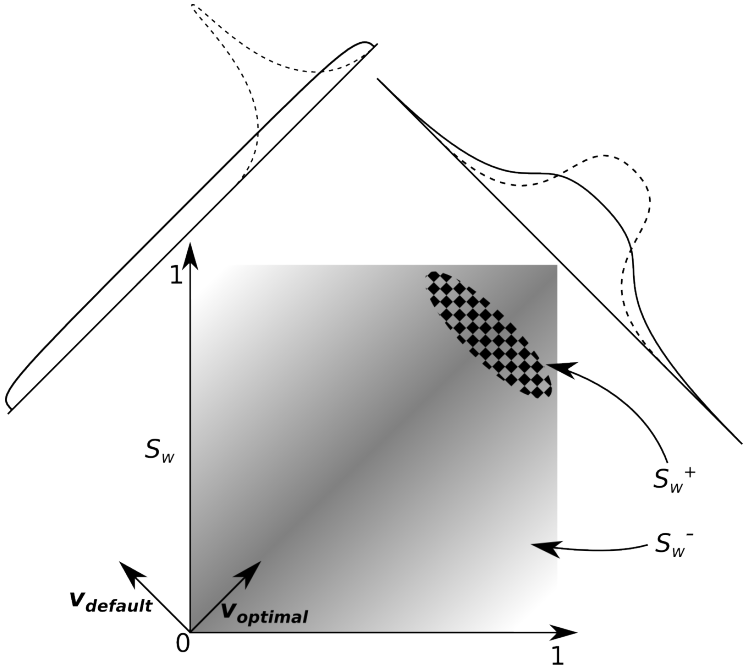


Figura 4.3: Um ECRS arbitrário $S_w$, similar ao apresentado na figura 4.2, quando projetado na direção de $\vec{v_{default}}$ é pouco discriminativo pois os pontos de $f(w, S_w^+)$ se misturam com grande frequência com os pontos de $f(w, S_w^-)$, porém quando projetado na direção do $\vec{v_{optimal}}$, ainda que haja mistura, há uma região em que a probabilidade da instância pertencer a $S_w$ é claramente superior.

## 4.2 Hipótese

Conhecer a forma com que os $S^+$ e $S^-$ se distribuem é importante para o estabelecimento de pesos que possam aumentar o poder discriminativo das Haar *wavelets*. Essas informações pode servir para estabelecer classificadores com maior poder discriminatório, porém, pelo que se pôde levantar, poucas pesquisas consideram isso ([40][41][56]). Nesta seção constam as hipóteses que servem de fundamentação teórica para as abordagens experimentais destinadas a explorar e entender melhor as propriedades desse espaço.

Levanta-se a seguinte hipótese sobre o ECRS: dado uma Haar *wavelet* $w$, o ECRS produzido com instâncias positivas $S_w^+$ se distribui Gaussianamente, enquanto o ECRS produzido com instâncias negativas $S_w^-$ se distribui uniformemente. For-

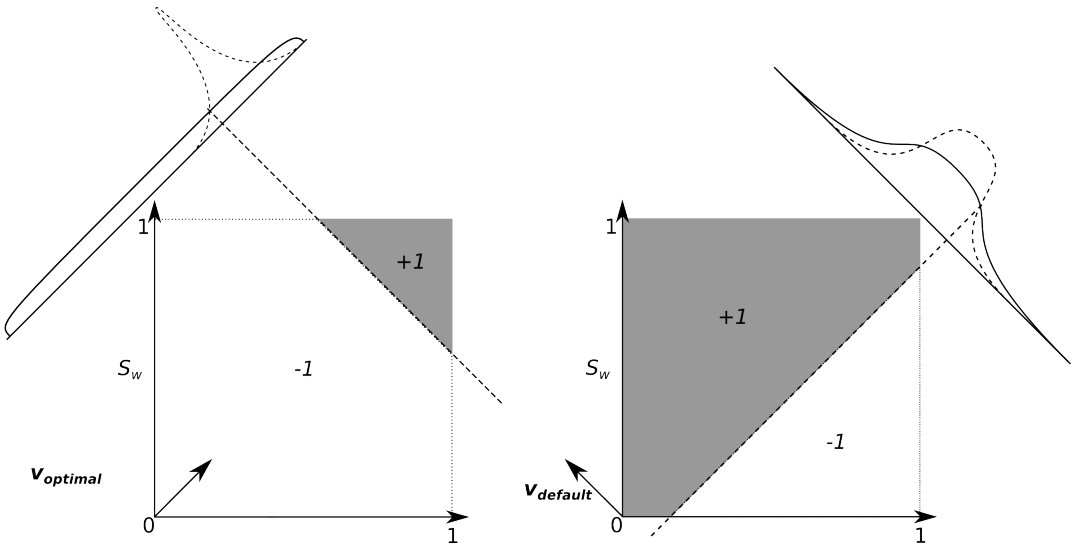


Figura 4.4: Continuando com o exemplo visto na figura 4.3, e assumindo $p = -1$, esta ilustração exemplifica como um classificador baseado em Haar *wavelet* poderia particionar $S_w$ com dois vetores de pesos diferentes.

malmente, $S^+ \sim \mathcal{N}(\mu^+, \Sigma^+)$ e $S^- \sim \mathcal{U}$. A motivação vem dos dados apresentados por Pavani e colaboradores, onde alguns conjuntos $S^-$ não parecem se distribuir conforme qualquer função de distribuição simples conhecida, além de se espalharem por todo o domínio. Nada disso é suficiente para estabelecer que a distribuição uniforme é o melhor modelo para $S^-$, mas na ausência de observações que contrariem a possibilidade de haver ECRSs suficientes que possam ser modelados como distribuições uniformes, essa hipótese serve como base para o estabelecimento de um experimento de verificação dessa possibilidade, sendo uma contribuição importante desta dissertação.

Outra motivação para essa hipótese é a possibilidade de se reduzir o tempo de treinamento de uma cascata classificadora completa. Os métodos de treinamento combinados com otimização de pesos das Haar *wavelets* propostos por Pavani e colaboradores, apesar de eficazes, tomam um longo tempo para concluírem. É evidente que a busca forçada em um domínio muito grande tomará muito tempo, e é comum os algoritmos genéticos demorarem para convergir. Mas, apesar da ADLF ser relativamente rápida, ela custa tempo adicional por razões mais sutis. Este método

depende da estimação da média e da variância dos conjuntos $S^+$ e $S^-$ antes de calcular o espalhamento intra-classe [12, p. 120]. Acontece que, conforme apresentado em 2.2, as amostras negativas são trocadas antes de se iniciar o processo de impulsionamento de um novo nó de uma cascata de classificadores, tornando necessário estimar a média e a variância de $S^-$ de todas as Haar *wavelets* antes da atribuição dos pesos com ADLF. Assumir que $S^- \sim \mathcal{U}$ implica na inexistência de informação sobre $S^-$ que possa ser usada para discernir entre instância negativa e positiva. Assim, restam somente as informações sobre $S^+$, que são imutáveis durante o treinamento afinal o conjunto de instâncias positivas não muda durante a formação da cascata de classificadores.

## 4.3 Abordagem experimental

Para produzir algum conhecimento novo sobre o ECRS, é necessário verificar as hipóteses apresentadas na seção 4.2, porém, há várias formas de fazer isso. Esta seção especifica os procedimentos e experimentos que serão aplicados no intuito de verificá-las.

Abordam-se as seguintes questões: (1) como, a partir das restrições da hipótese em questão, se atribuirão pesos às características baseadas em Haar *wavelets*? (2) que métodos serão utilizados para explorar as distribuições de cada classe no ECRS? e (3) como isso ajuda a coletar mais informações sobre o ECRS?

Conforme apresentado na seção 4.2, assume-se que elementos de $S_w^+$ se distribuirão gaussianamente, enquanto os de $S_w^-$ estão distribuídos uniformemente por todo o ECRS. Desse modo, para fins de classificação, não há nenhuma informação útil que se possa retirar da análise de $S_w^-$, restando trabalhar apenas com estatísticas e parâmetros de $S_w^+$.

Uma das formas de se atribuir valores ao vetor de pesos duma característica baseada em Haar *wavelet*, é através da análise de componentes principais de $S_w^+$. Como os pontos de $S_w^-$ estão presentes em todo o ECRS, deseja-se que $f(w, S_w^+)$ se concentre no menor espaço possível. O vetor paralelo à componente principal

de menor variância de uma distribuição gaussiana multivariada permite a projeção de pontos dessa forma (vide seção 2.5 e figura 2.6). Portanto, propõe-se que $\vec{v}$ seja o vetor unitário paralelo à componente principal de menor variância de $S_w^+$. Esse conjunto de pontos será criado para cada Haar *wavelet* a partir das instâncias positivas disponíveis para o treinamento.

Propõe-se também uma nova característica que usa os parâmetros da distribuição de $S_w^+$. Seja $\vec{\mu_w}$ a média de $S_w^+$. Define-se a nova característica como:

$$f'(w) = |\vec{v_w}(\vec{s} - \vec{\mu_w})|. \tag{4.2}$$

Ao combinar tal característica com o classificador de Viola e Jones e seus parâmetros $p$ e $\theta$, cria-se uma "faixa" classificatória no ECRS de $w$ perpendicular a $\vec{v_w}$, que passa por $\mu$ e tem largura $2\theta$. A figura 4.5.

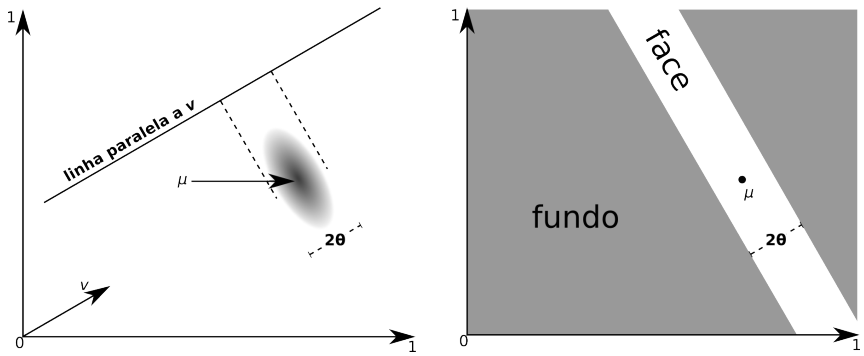


Figura 4.5: Descrição e efeitos no ECRS dum classificador fraco usando a característica descrita na equação 4.2.

A intuição por detrás desse classificador fraco é simples: quanto menor a área do ECRS destinada a classificar instâncias como positivas, menor será a quantidade de falsos positivos e falsos negativos. Isso faz sentido devido à forma com que $S_w^+$ e $S_w^-$ se distribuem: o primeiro se concentra ao entorno dum valor central, e o segundo está presente em todo o espaço. O classificador fraco de Viola e Jones faz apenas um "corte" no ECRS, enquanto o classificador proposto faz dois "cortes", limitando ainda mais a área do ECRS destinada à classificação positiva de instâncias.

Nesse classificador, $\vec{v}$ também será atribuído via ACP da mesma forma que a mencionada acima, portanto é razoável esperar que os valores das características extraídas de instâncias positivas sejam projetadas de forma bastante compacta ao entorno de $\vec{\mu_w}\vec{v}$. Considerando isso e o explicado acima sobre a largura da "faixa", espera-se que durante o impulsionamento o aprendiz fraco atribua valores bastante pequenos a $\theta$, reduzindo os erros de classificação dos aprendizes fracos. O algoritmo 3 descreve tal procedimento formalmente. Note que, tal algoritmo, salvo pela atribuição de $\vec{\mu_w}$, também é usado para a simples atribuição de pesos.

**Algoritmo 3:** Atribuição de $\vec{v_w}$ e $\vec{\mu_w}$ para a característica $w$

**Entrada**: $x_i \in X^+$, o conjunto de instâncias positivas.
**Entrada**: $w_k, k = 1, \ldots, K$, as características baseadas em Haar *wavelets*.
**Saída**: $\vec{v_k}$, o novo vetor de pesos para $w_k$.
**Saída**: $\vec{\mu_k}$, a média de $S_k^+$.
**Início**
  **Para cada** $w_k$ **faça**
    Use $w_k$ para produzir $S_k^+$ com cada $X^+$.
    Estime $\vec{\mu_k}$, a média de $S_k$.
    Estime $\Sigma_k$, a matriz de covariância de $S_k$.
    Encontre os autovalores e autovetores correspondentes de $\Sigma_k$.
    Atribua a $\vec{v_k}$ o autovetor de menor autovalor de $\Sigma_k$.
  **Fim para cada**
**Fim**

Em [30], Landesa-Vázquez e Alba-Castro propuseram um classificador que também usava o valor absoluto dos vetores de características na classificação fraca, apesar de terem se inspirado em aspectos fisiológicos da visão e não em observações estatísticas dum espaço vetorial. A semelhança do trabalho desses pesquisadores se encerra, portanto, neste único aspecto.

Essas propostas precisam ser testadas experimentalmente. Se o classificador forte resultante do emprego dessas abordagens obtiver um bom desempenho, então há características suficientes que sustentam as hipóteses sobre o espalhamento de $S_w^+$ e $S_w^-$ para a produção de um classificador dentro dos parâmetros experimentais

usados (capítulo 5). Isso é diferente de dizer que todas as Haar *wavelets* se distribuem dessa forma. De forma análoga, se o classificador forte desempenhar mal, não quer dizer que nenhuma característica se enquadra nas hipóteses, mas que há poucas características que apresentam tal comportamento.

É interessante comparar as propostas com outras, de modo a verificar seus desempenhos. Duas que apresentam certa semelhança com a apresentada aqui foram feitas por Adhikari e Rasolzadeh e seus respectivos colaboradores [1] [45]. Por causa disso, elas, junto com o arcabouço original de Viola e Jones, foram escolhidas para servirem como comparação com a proposta nesta dissertação.

# 5 EXPERIMENTOS

Este capítulo descreve em detalhes toda a preparação e configuração dos experimentos realizados para averiguar as hipóteses, conforme as abordagens descritas na seção 4.3, e os resultados obtidos.

Experimentos similares do autor e do orientador dessa dissertação foram publicados em [34].

## 5.1 Base de instâncias para treinamento

Num primeiro experimento simples, Viola e Jones treinaram um classificador monolítico usando 10000 amostras de faces e 15000 imagens de fundo [60]. O detector de pedestres de Rasolzadeh et al. [45] usa um classificador forte monolítico impulsionado com 8000 instâncias positivas e outras 8000 negativas. Já o treinamento de uma cascata de classificadores depende de muito mais dados. Viola e Jones [60] relatam o uso de amostras de imagens com 24 pixels, totalizando 4916 faces e cerca de 350 milhões de instâncias negativas. Já Pavani e colaboradores [41] produziram seus classificadores usando 5000 imagens de faces e cerca de $4 \times 10^9$ imagens de outros objetos, todas em forma quadrada de 20 pixels de lado. Todas essas imagens estavam em escala de cinza de 256 tons.

No trabalho de Pavani e colaboradores, as instâncias positivas foram obtidas de várias bases de dados e tratadas manualmente de modo a terem as características mencionadas acima. Já as instâncias negativas foram obtidas a partir do recorte de 27000 fotografias obtidas da Internet.

As imagens de faces usadas para esta dissertação terão as mesmas características das de [41]. Esta seção apresenta a preparação dos dados usados neste trabalho.

### 5.1.1 Instâncias positivas

Quatro bases de faces foram usadas para a criação do conjunto de instâncias positivas necessárias para os treinamentos: a BioId face database; a AR Face Database; a JAFFE Database; a MIT-CBCL Face Database #1; e a FEI Face Database (respectivamente [24], [35], [33], [36] e [27]). Um total de 7129 faces foram extraídas com programas projetados especificamente para tratar cada base. Note que esse conjunto difere do usado em [41] pois algumas das bases não estão mais disponíveis e precisaram ser substituídas por outras.

A base MIT-CBCL #1 [36] contém 2429 imagens de faces em escala de cinza de 256 tons com altura e largura de 19 pixels. Esta é a base de dados que melhor se adequa aos requisitos deste trabalho, visto que precisou apenas ser redimensionada para ter lados de 20 pixels. Isso foi feito com o auxílio de um programa que redimensionava cada imagem e interpolava linearmente os valores dos pixels.

Outra base utilizada foi a BioID [24], com 1521 imagens em escala de cinza de 256 tons e 384 pixels de largura por 284 de altura. Cada imagem está associada a um arquivo que descreve a posição dos olhos da única pessoa presente numa imagem. Segundo seus autores, esta não é uma base destinada ao treinamento, mas sim ao teste de detectores faciais. Porém, como durante esta pesquisa não se descobriu trabalhos de interesse que fornecessem resultados de testes de detectores de faces com esta base, optou-se por usá-la como fonte de instâncias de treinamento.

Como as faces presentes na base são majoritariamente de pessoas olhando frontalmente para a câmera, foi possível, sem muitas dificuldades, extrair dados úteis para o treinamento. Para tanto, um programa manipula automaticamente cada imagem a partir da posição dos olhos da pessoa. A imagem original é rotacionada para que ambos os olhos fiquem na horizontal, recortada proporcionalmente à distância dos olhos e redimensionada para que seus lados tenham 20 pixels.

A ilustração 5.1 ajuda a explicar melhor as operações feitas nas imagens da base BioID. Considere que o eixo horizontal da imagem crescendo da esquerda para a direita, e o eixo vertical crescendo de cima para baixo. Nesse plano, seja o centro do

olho direito $d = (x_d, y_d)$, o centro do olho esquerdo $e = (x_e, y_e)$, e $\hat{b}$ o ângulo entre a reta que contém $e$ e $d$ e a horizontal, crescendo no sentido anti-horário. A janela quadrada de lado $L$, cujo vértice superior esquerdo está em $(x_l, y_l)$ será usada para o recorte da face, mas antes é necessário rotacionar a imagem ao redor do olho direito de modo a alinhar com a horizontal ambos os olhos. A região delimitada pela janela é extraída e redimensionada para o tamanho adequado. Todo esse tratamento foi inspirado pelo descrito em [2] e [3].

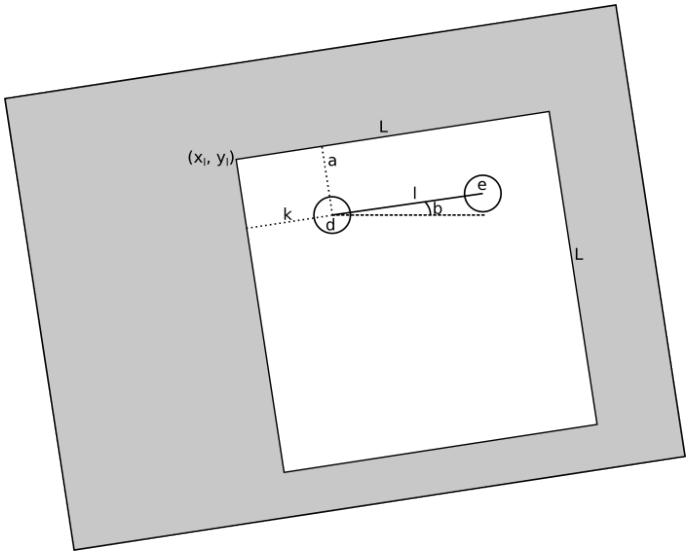


Figura 5.1: Formação da janela para extração automática de face de uma imagem da base de dados BioID. A área branca representa a região de onde se extrai a face, enquanto a área cinza representa o restante da imagem.

Os parâmetros relevantes para essas operações foram calculados a partir da avaliação manual de algumas imagens. Estabeleceu-se como a distância entre a aresta superior da janela e o plano horizontal dos olhos é $a = 0.25L$, e a distância entre os olhos é $l = 0.5154L$, portanto $k = L(1 - 0,5154)/2$ é a distância entre o centro dos olhos para as arestas verticais da janela.

Uma base de faces que se utilizou neste trabalho, mas não foi usada por Pavani e colaboradores é a FEI [27], cuja versão original contém fotografias coloridas de 640 pixels de largura e 480 de altura de 200 pessoas, sendo 100 de cada sexo, em 14

variações de poses e condições de iluminação diferentes num ambiente controlado. Outras versões são produto de diversos trabalhos já feitos sobre a base original. A versão usada contém imagens em escala de cinza com 250 pixels de largura e 300 de altura dos mesmos 200 indivíduos olhando diretamente para frente, mas em duas poses: relaxados ou sorrindo. As pessoas dessas imagens estão com a posição dos olhos normalizadas tanto vertical quanto horizontalmente. Mais detalhes em [3].

Extrair faces da base FEI é simples: basta recortar a imagem para que fique com um aspecto quadrado, ao invés de retangular e redimensioná-las. Optou-se por deixar a distância do olho ao topo da janela similar à deixada nas imagens extraídas da base BioID, equivalente a 1/4 da altura total da janela. Os pixels restantes da altura vertical são removidos da parte de baixo da imagem. Das arestas verticais, deu-se 25 pixels de distância.

A AR Face Database [35] contém fotografias coloridas com 120 pixels de largura e 165 de altura de faces de 126 pessoas tiradas frontalmente em dois dias diferentes. Cada pessoa posou para 13 fotografias com expressões faciais, acessórios e em condições de iluminação específicas em cada dia. Exceto pela necessidade de descoloração das imagens, o processo de extração é muito similar ao empregado para o banco de faces da FEI, visto que as imagens também foram normalizadas.

A base de dados JAFFE (*Japanese Female Facial Expression*) contém 191 fotografias frontais de faces de 10 mulheres japonesas expressando felicidade, tristeza, surpresa, raiva, desapontamento e medo). As imagens são quadrados de 256 pixels de lado em escala de cinza. Em todas as imagens os olhos estão sobre a mesma linha horizontal. Todas as faces desta base foram extraídas usando também um processo similar ao utilizado na FEI.

A figura 5.2 mostra algumas instâncias de faces extraídas dessas bases de dados usando os métodos descritos.

A união de todas as instâncias devidamente tratadas de todas essas bases forma uma base de dados com 7129 imagens para o impulsionamento do classificador. As atribuições de pesos às características baseadas em Haar *wavelets* foram feitas com

Figura 5.2: Amostra de faces usadas na preparação dos classificadores.

um subconjunto com 4938 instâncias desses mesmos dados, mas excluindo as imagens oriundas da JAFFE e das bases 5, 6, 9, 10, 11, 12 e 13 da AR Face Database. A criação dessas duas bases é consequência da observação dos resultados obtidos em experimentos prévios [34] em que se observou a ocorrência de *overfitting* em um dos detectores. Uma causa possível disso foi o uso do mesmo conjunto de dados para atribuição de pesos e impulsionamento.

### 5.1.2 Instâncias negativas

Um total de 114.865 imagens quadradas de lado de 20 pixels com 256 tons de cinza foram obtidas a partir de cerca de 2000 fotografias digitais coloridas. Criou-se um programa que iterava por essa coleção redimensionando, recortando e descolorindo frações das imagens. Nenhum recorte se sobrepunha a outro.

Muitas dessas fotografias continham faces de pessoas, portanto implementou-se um programa que permitia a exclusão de partes das imagens antes da extração de amostras para a base de dados. Note que outras partes do corpo humano, roupas e acessórios, constam no conjunto de instâncias negativas.

Também foram incluídas nessa base recortes de fotografias de reproduções artísticas de faces, contanto que tais reproduções não passassem de imitações simplistas (para os propósitos deste trabalho). Assim, entre as instâncias negativas, constam, por exemplo, imagens de pinturas ou de artesanato. Reproduções muito fiéis da face humana, principalmente de objetos tridimensionais cujas sombras trazem profundidade e complexidade à fotografia, não foram incluídas. A figura 5.3 apresenta algumas dessas instâncias de imagens de fundo.

Figura 5.3: Amostra de imagens de fundo usadas na preparação dos classificadores.

Dessa base, somente uma fração foi utilizada no impulsionamento. Para escolhê-las, em experimentos preliminares, formaram-se dois subconjuntos de instâncias. O primeiro foi formado aleatoriamente, e o segundo, manualmente, mas considerando o desvio-padrão das intensidades de pixels da instâncias. Em impulsionamentos preliminares, notou-se que o classificador produzido com a segunda abordagem apresentava melhor precisão, portanto ela se tornou a favorita para formação do subconjunto de instâncias negativas usadas na otimização e impulsionamento.

A seleção manual das instâncias envolveu o cálculo do desvio padrão das intensidades dos pixels e o estudo da distribuição do desvio padrão com um histograma

(figura 5.4). Um gerador uniforme de números aleatórios foi utilizado na primeira abordagem acima, portanto a distribuição do desvio padrão das intensidades de pixels das instâncias de tal subconjunto era muito parecida com o do conjunto completo. Isso quer dizer que muitas imagens com variedade tonal relativamente baixa estavam no primeiro subconjunto.

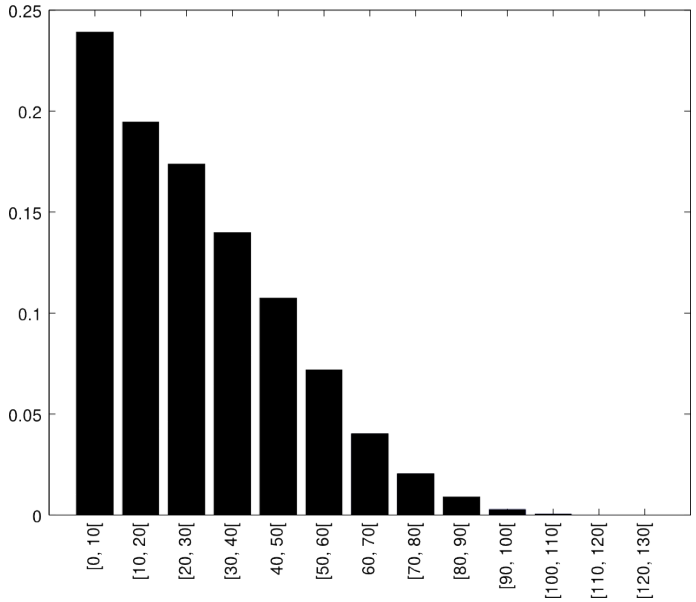


Figura 5.4: Histograma das 114.865 instâncias negativas. O eixo horizontal representa os intervalos de desvio padrão.

Supôs-se que instâncias com baixa variedade tonal podem ser uniformes demais para servirem de exemplos interessantes para a formação do classificador forte, pois tais instâncias podem ser "lisas", i.e. desprovidas de textura e complexidade. Podem ser, por exemplo, amostras de um céu claro e sem nuvens, de uma parede branca, ou imagens escuras demais para se distinguir qualquer objeto. Por outro lado, é importante que imagens desse gênero tenham alguma representação, já que elas existem e, muitas vezes, dominam uma fotografia.

A escolha manual foi feita da seguinte forma: ordenou-se o conjunto completo de instâncias por desvio padrão e dividiu-se essa lista em 4 listas de tamanho igual. As 2000 instâncias de maior desvio padrão de cada lista formaram o subconjunto men-

cionado na segunda abordagem. A distribuição dos desvios padrão das intensidades dos pixels de suas imagens está representada na figura 5.5.

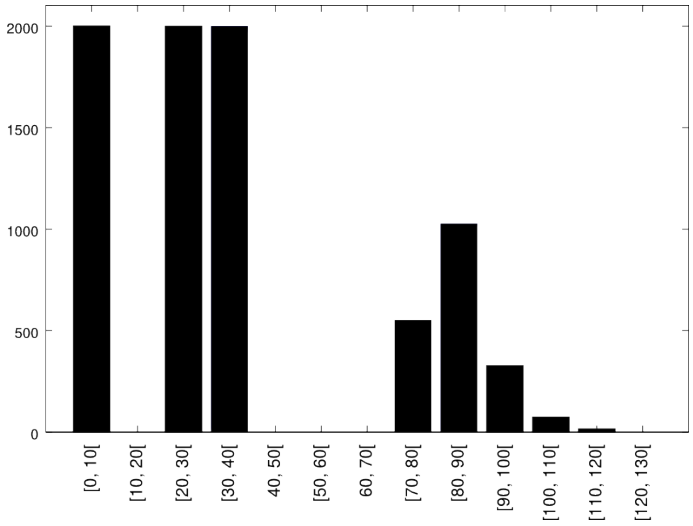


Figura 5.5: Histograma das 8000 instâncias negativas utilizadas no impulsionamento de classificadores.

## 5.2 Geração de Haar *wavelets*

Forma-se uma Haar *wavelet* combinando retângulos que caibam na janela de inspeção da imagem e atribuindo-lhes pesos. Mesmo numa janela de 20 pixels de lado, se não houver restrições para a combinação de tais retângulos, a quantidade de Haar *wavelets* que se pode formar é tão grande que torna impraticável a criação de detectores com todos eles, portanto é necessário limitar esse processo.

Os parâmetros de cada Haar *wavelet* utilizado neste trabalho foram produzidos com um programa que implementa as mesmas regras mencionadas em [41], acrescidas de outras regras novas que efetivamente limitavam a quantidade de Haar *wavelets* gerados para uma quantidade praticável. Tais regras são:

1. somente 2 a 4 retângulos podem ser combinados numa Haar *wavelet*;

2. o gabarito de cada Haar *wavelet* é um quadrado de 20 pixels de lado, portanto todo retângulo deve ficar contido nessa área;

3. não serão usadas Haar *wavelets* rotacionados, como os propostos por Lienhart e Maydt [32];

4. as distâncias $dx$ e $dy$ entre os retângulos, conforme descritas por Li et al. [31], são números inteiros múltiplos do tamanho do retângulo na respectiva direção;

5. todo retângulo duma Haar *wavelet* tem as mesmas altura e largura;

6. as dimensões mínimas de qualquer retângulo são de 3 pixels de altura e 3 de largura.

Pavani e colaboradores usaram um total de 207.807 Haar *wavelets*, e Viola e Jones [60] 160.000. Com apenas essas regras, gerou-se 1.641.107 Haar *wavelets* diferentes, o que é um número grande demais para ser treinado e impulsionado. O conjunto final foi reduzido para 218.544 através da manipulação manual, por tentativa e erro, de parâmetros do programa gerador.

Até onde o trabalho de pesquisa avançou, observou-se que os autores não descrevem seus conjuntos de Haar *wavelets*. No intuito de deixar mais explícito o conjunto usado, realizou-se uma análise de alguns de seus atributos. Dentre os 218.544 Haar *wavelets*, 28.927 são compostos por 2 retângulos, 71.423 por 3 retângulos e 118.194 por 4 retângulos. Há, portanto, um total de 744.899 retângulos. Suas larguras e alturas variam de 3 a 20 pixels, e a distribuição de alturas e larguras de retângulos é idêntica. Vide figura 5.6.

A ilustração seguinte mostra a distribuição dos mesmos retângulos pela janela quadrada. Ela está dividida em três segmentos verticais e três horizontais, demarcando as principais regiões da face humana quando fotografada frontalmente.

### 5.2.1 Atribuição de pesos

O método tradicional de atribuição de pesos a cada retângulo de uma Haar *wavelet* é simples: cada retângulo recebe um valor inteiro de modo que a soma dos pesos resulte zero [39] (Figura 5.8).

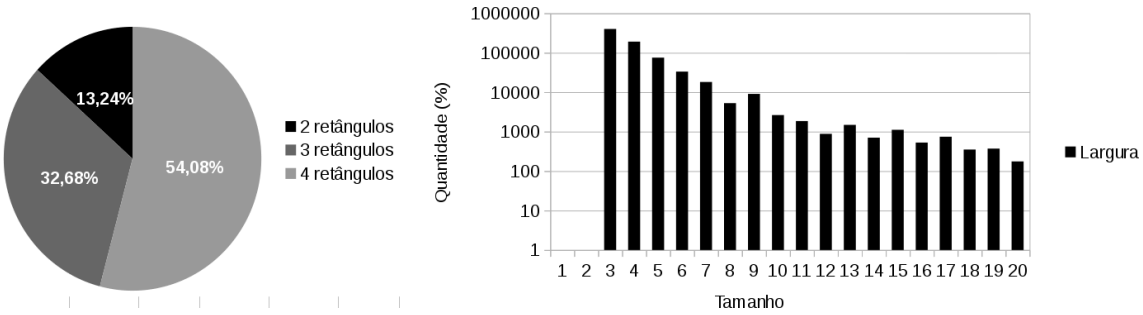


Figura 5.6: À esquerda, percentual de Haar *wavelets* de diferentes dimensões. À direita, histograma, em escala logarítmica, de retângulos de determinada largura de todas as Haar *wavelets*.

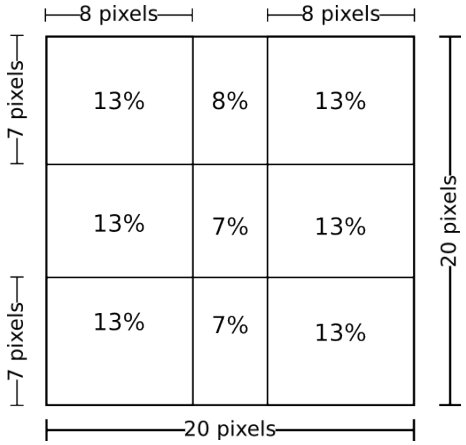


Figura 5.7: Distribuição dos centros dos retângulos na janela quadrada de 20 pixels de lado. Os percentuais contidos em cada partição contam a quantidade de centros de retângulos presentes nelas.

Conforme apresentado na seção 4.3, os pesos dos classificadores utilizados para testar a primeira hipótese serão atribuídos através de uma rotina baseada em análise de componentes principais. As imagens usadas para este fim estão discriminadas na seção 5.1.1.

Os classificadores baseados no trabalho de Viola e Jones, Adhikari e Rasolzadeh usam os pesos tradicionais.

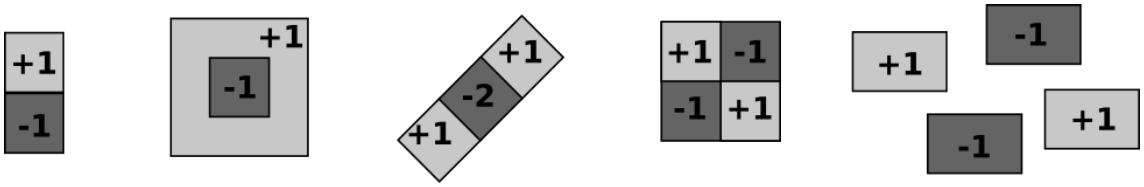


Figura 5.8: Exemplos de atribuição de pesos típicos a Haar *wavelets*.

## 5.3 Configuração do treinamento

Todos os classificadores fortes produzidos neste trabalho são monolíticos e foram treinados até possuírem 200 classificadores fracos. Apesar dos classificadores em cascata serem mais indicados para uso em aplicações reais, são dispensáveis para os objetivos desta dissertação, afinal nenhuma manipulação especial na produção ou emprego da cadeia classificadores foi feita. Isso também simplifica os esforços de implementação.

Para o impulsionamento utilizou-se o AdaBoost apresentado na seção 2.1.2, exceto pela atribuição inicial de pesos: optou-se por empregar a abordagem de Viola e Jones [60] onde os pesos iniciais das instâncias de treinamento são atribuídos como descrito na seção 2.2.

A estratégia de reponderação foi adotada para fornecimento de instâncias para o aprendiz fraco. Apesar da estratégia de reamostragem produzir classificadores fortes mais precisos (seção 2.1.2 e [54]), ela é de implementação e execução ligeiramente mais demorada. Como não é objetivo primordial deste trabalho estabelecer um novo estado da arte na classificação de faces, valorizou-se mais a simplicidade, ao invés de ganho de precisão.

## 5.4 Configuração dos testes

Os resultados foram colhidos com um detector de faces acoplado a cada um dos classificadores fortes produzidos. Tal detector opera como descrito por Viola e Jones [60], e revisto na seção 2.2. Cabe apenas mencionar alguns detalhes e parâmetros que afetam o comportamento do detector. Seja $z$ a razão entre o tamanho atual

da janela e o tamanho original, e seja $\Delta$ um parâmetro arbitrário. A janela se desloca $\lfloor \Delta z \rfloor$ pixels na direção vertical ou horizontal. Após esquadrinhar toda a imagem com um tamanho, cada lado da janela de detecção aumenta em 25% e o processo se repete, a menos que um de seus lados seja maior que um dos lados da imagem, fazendo o detector buscar faces na próxima imagem. Os valores iniciais dos parâmetros são: $\Delta = z = 1.5$. O tamanho inicial da janela de detecção é $20 \times 20$ pixels.

A busca de faces ocorre nas bases de dados MIT + CMU A, B e C [47] [58], que possuem 155 imagens em 256 tons de cinza, cada uma podendo conter faces de pessoas em diversas circunstâncias, olhando diretamente, com pequenos desvios, para a câmera, como na figura 5.9. Essa base é comumente utilizada na comparação do desempenho de detectores de faces frontais. Alguns de seus usuários são [6], [20], [30], [31], [41] e [60].

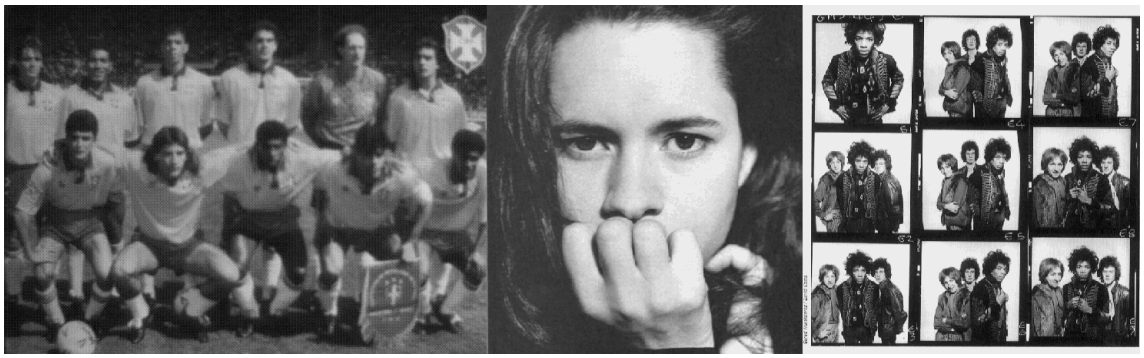

Figura 5.9: Exemplos de imagens contidas nas bases MIT+CMU A, B e C.

Além das imagens, a MIT + CMU contém arquivos com as coordenadas verticais e horizontais de partes importantes das 511 faces presentes nas imagens. Com esses dados é possível calcular as áreas retangulares onde se considera que as faces estão de fato, e a partir das quais é possível comparar os resultados do detector. Neste trabalho, os parâmetros e métodos de cálculo de tais regiões são os mesmos que os descritos para a extração de faces da base BioId (vide seção 5.1.1). Um exemplo do resultado deste método de marcação está na figura 5.10.

Figura 5.10: Uma imagem da base de dados MIT+CMU em que as regiões que contém as faces verdadeiras (calculadas como apresentado nesta seção) estão marcadas por um retângulo branco.

A verificação do resultado só é possível se algum critério rígido de aceitação de detecção de face em uma subjanela for estabelecido. Tal critério determina se uma subjanela classificada pelo detector como face está dentro de margens aceitáveis de erro, portanto se o resultado é um falso ou verdadeiro positivo. Em [6] uma detecção é considerada correta se a região detectada contém todas as anotações da face (olhos, nariz e boca) e o tamanho do detector é menor que quatro vezes a distância entre os olhos. Para Pavani et al. [41], uma detecção é considerada correta quando o tamanho da região detectada é $\pm 10\%$ da face anotada, e quando a distância do centro da região detectada ao centro da região anotada é no máximo $\pm 10\%$ do tamanho da anotação. Optou-se por usar a abordagem de Pavani e colaboradores.

Nessas circunstâncias o detector inspeciona um total de 19.024.094 subjanelas. Com o critérios de aceitação de detecção, as 511 regiões consideradas como faces são convertidas em 2.458 subjanelas verdadeiras positivas, portanto há 19.021.636

subjanelas sem faces. Nenhuma integração de subjanelas (como na seção 2.2.2) foi feita.

Comparam-se os desempenhos dos detectores com a análise da curva ROC [13], manipulando o limiar do classificador forte de $-\infty$ a $+\infty$. Análises adicionais serão feitas caso a caso, de acordo com a hipótese relacionada com o experimento.

## 5.5 *Software* reusado

As ferramentas e algoritmos apresentados nessa dissertação foram implementados em C++ e compilados com GCC. Toda manipulação de imagem foi apoiada pela OpenCV, e algumas rotinas de carga e leitura de propriedades dos arquivos de imagem foram escritas com OpenImageIO. As ferramentas que dependiam de alguma interface gráfica foram criadas em Qt 4. Facilmente paralelizaram-se diversos algoritmos, tais como o aprendiz fraco e a atribuição de pesos com ACP, com a Intel® Threading Building Blocks. A atribuição de pesos foi implementada com uma versão ligeiramente modificada das bibliotecas libpca e Armadillo [49]. Alguns módulos da Boost C++ Libraries foram usados para diversos fins. A plotagem de gráficos e algumas análises simples foram feitas com GNU Octave [38] e Inkscape.

## 5.6 Resultados

Ao todo, 5 classificadores diferentes foram produzidos: três referentes à primeira hipótese e dois referentes à segunda.

Os detectores referentes ao primeiro experimento são os seguintes:

- **detector A:** idêntico ao proposto por Viola e Jones [60];
- **detector B:** como o de Pavani [41], mas com classificadores fracos cujos pesos foram atribuídos com PCA (vide seção 4.3); e
- **detector C:** que usa a característica proposta neste trabalho que forma uma "faixa" no ECRS (vide a seção 4.3).

Os detectores referentes ao segundo experimento são:

- **detector D:** como proposto por Adhikari et al. [1], que usa funções discriminates quadráticas; e
- **detector E:** como proposto por Rasolzadeh et al. [45], cujos classificadores fracos usam histogramas.

O computador utilizado para execução de todos os programas possui um processador Intel® Core™ i7 com 8 núcleos 64 GiB de memória RAM. O sistema operacional é um Linux Debian 3.14.7-1 de 64 bits. Apesar da fartura de RAM, o maior consumo desse recurso foi inferior a 3 GiB, o que ocorreu durante o impulsionamento. Todos os programas apresentam algum grau de paralelização de suas rotinas principais.

A atribuição de pesos, parâmetros ou histogramas para todas as 218.544 características usadas nos detectores B, C e E com as 4938 instâncias positivas tomou pouco menos que 4 minutos.

O impulsionamento das 218.544 características baseadas em Haar *wavelets* com todas as 15.129 instâncias de imagens destinadas ao treinamento tomou entre 13 e 17 horas. Apesar do aprendiz fraco operar em paralelo na mesma máquina mencionada acima, ele não foi escrito buscando otimizar, por exemplo, o uso do cache do processador. Também não utiliza técnicas mencionadas em [64], que aceleram significativamente este processo.

Toma pouco mais que 2,5 minutos a execução dos testes com cada detector dotado do classificador forte com 200 classificadores fracos contra a base de imagens MIT + CMU A, B e C.

A curva ROC sobre o desempenho dos 5 detectores que usam um classificador forte dotado de 200 classificadores fracos está na figura 5.11. As áreas sob as curvas estão na tabela 5.1. O detector B apresenta a maior área sob a curva, seguido do detector C. O detector A obteve a terceira maior área, e os seguintes foram o D e E com quase o mesmo desempenho.

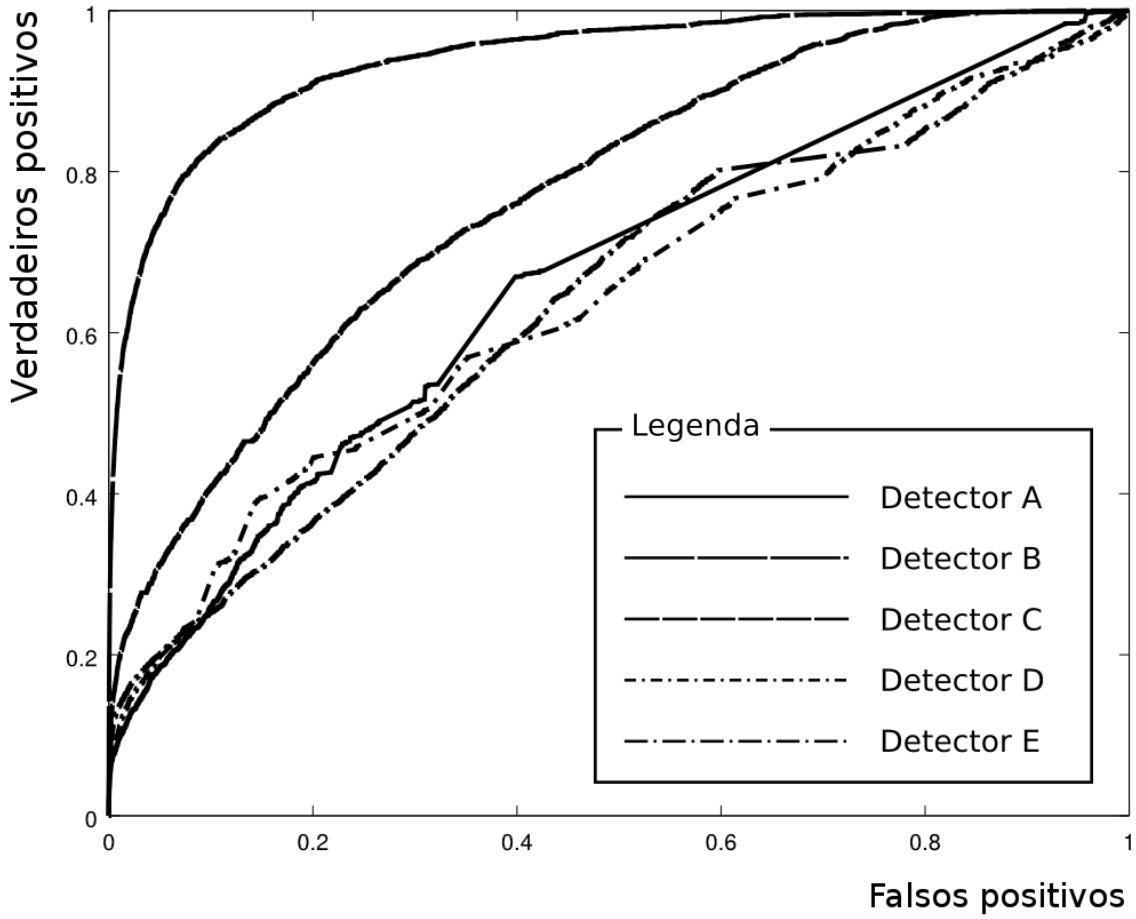


Figura 5.11: Curvas ROC dos 5 detectores.

O teste também foi feito com versões dos mesmos detectores com apenas os primeiros 20, 40, ..., 160, 180 classificadores fracos do classificador forte. As áreas sob a curva ROC de cada caso estão na tabela 5.1.

As melhores áreas sobre as curvas ROC dos detectores A, D e E apresentam valores muito próximos, porém A somente se equipara aos demais com 2,5 ou 5 vezes mais classificadores fracos. Isso importa pois o tempo de impulsionamento e de detecção é linear em relação à quantidade de classificadores fracos (seções 2.1.2 e 2.4). O cômputo dos valores das características do detector D, por causa da complexa equação que usa, é bem mais lento que dos demais. Considerando a precisão e o velocidade de operação, é fácil perceber que os detectores D e E superam facilmente

Tabela 5.1: Desempenho dos classificadores fortes com diferentes quantidades de classificadores fracos. As células destacadas contém o melhor resultado de cada detector.

| | Área sob a curva ROC (%) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Det. | 20 | 40 | 60 | 80 | 100 | 120 | 140 | 160 | 180 | 200 |
| A | 58.7 | 62.9 | 63.3 | 63.5 | 66.4 | 66.3 | 66.1 | 66.1 | 66.2 | 66.2 |
| B | 90.5 | 92.6 | 92.8 | 93.1 | 93.2 | 93.6 | 93.5 | 93.8 | 94.1 | 94.0 |
| C | 76.1 | 75.2 | 74.4 | 73.5 | 73.9 | 75.2 | 75.6 | 75.7 | 76.0 | 77.1 |
| D | 57.1 | 66.5 | 66.3 | 66.1 | 65.3 | 63.7 | 62.4 | 63.9 | 64.0 | 64.1 |
| E | 65.9 | 64.0 | 64.1 | 64.1 | 63.4 | 63.5 | 63.9 | 64.0 | 64.0 | 63.6 |

o detector A, mas o desempate entre esses dois depende da aplicação que lhes seria dada. A tabela 5.2 mostra os dados que suportam essas observações.

Tabela 5.2: Tempo de avaliação das imagens da base de testes.

| Detector | Qtd. de classificadores fracos | Tempo em minutos e segundos |
|---|---|---|
| A | 100 | 1:22 |
| A | 20 | 0:19 |
| B | 100 | 1:13 |
| B | 20 | 0:19 |
| C | 20 | 0:19 |
| D | 40 | 0:49 |
| D | 20 | 0:27 |
| E | 20 | 0:19 |

A figura 5.12 mostra a mesma imagem da base MIT + CMU apresentada na figura 5.10, porém com os resultados da detecção de faces obtidos com o detector B com limiar igual a 16. Este limiar não é o ótimo para este detector, mas serve para ilustrar um resultado possível que este detector monolítico pode alcançar.

### 5.6.1 Discussão

Antes de analisar e comparar em mais detalhes os desempenhos dos classificadores, é importante destacar as peculiaridades, diferenças e similaridades notórias entre os métodos usados aqui e nas pesquisas em que foram originalmente propostos.

Primeiro, é importante mencionar que no trabalho de Rasolzadeh et al. [45] os

histogramas dos classificadores fracos têm 128 intervalos de comprimentos iguais, mas não há menção sobre seus domínios. Nesta dissertação, para o detector E, arbitrou-se que tal domínio seria $[-\sqrt{2}, \sqrt{2}]$, que são os maiores valores que podem ser alcançados por qualquer elemento do vetor $\vec{s}$. A análise dos histogramas contidos nos classificadores fracos revela, infelizmente, que esse domínio é pequeno demais, pois muitos valores de características ficaram abaixo ou acima de seus limites. A figura 5.13 apresenta o perfil do primeiro classificador fraco contido no detector E, onde é possível notar isso. Apresenta também o perfil dos 3 primeiros histogramas contidos no detector E, mas com o primeiro e último intervalos excluídos para facilitar a visualização. É provável que seja possível obter desempenho superior desse detector se os domínios dos classificadores fracos forem ampliados, mas, mesmo assim, é interessante notar que o perfil de diversos desses histogramas se assemelham aos descritos em Rasolzadeh et al. e em Adhikari et al. [1].

Tanto Rasolzadeh et al. [45] quanto Adhikari et al. [1] não especificaram qual probabilidade *a priori* utilizaram para o conjunto de instâncias positivas e negativas. Nos experimentos relatados nesta dissertação, assumiu-se que essas probabilidades deviam ser proporcionais à quantidade de instâncias de cada classe utilizadas no treinamento.

Os classificadores de Adhikari et al. e Rasolzadeh et al. não foram aplicados na detecção de faces, mas na detecção de traseiras de carros e de pedestres respectivamente. Para o teste usaram bases próprias, cujas características não são conhecidas publicamente, mas informaram a quantidade de subjanelas de interesse e de fundo, cujas proporções são de 1 : 1 no trabalho de Adhikari et al., e de $O(1) : O(10^2)$ no trabalho de Rasolzadeh et al.. Nesta dissertação, a base de testes utilizada é conhecida e acessível publicamente, e a proporção de instâncias é de $O(1) : O(10^4)$, similar ao que ocorre no trabalho de Viola e Jones [60].

É importante comparar os resultados obtidos com algumas peculiaridades dos detectores. As curvas ROC de A, D e E apresentam seções retas longas, o que ocorre quando um classificador forte produz muitos valores iguais para diferentes amostras.

Já os detectores B e C apresentam seções retas tão curtas que, na imagem, não parece que existem. Uma causa para isso pode ser o pré-processamento das subjanelas, pois os detectores A, D e E as pré-processam como especificado por Viola e Jones [60] e complementado por Lienhart e Maydt [32], enquanto B e C estão conforme Pavani et al. [41]. Uma outra causa para isso ter ocorrido com o detector A pode ser atribuída à quantidade inferior de instâncias utilizadas em seu treinamento (15.129), afinal, em [60], o treinamento ocorreu com 20.000 instâncias. Para o detector E, uma provável explicação para este fenômeno foi o tamanho reduzido do domínio dos histogramas, conforme mencionado acima. Por fim, não é possível afirmar muito mais sobre o detector D pois não há resultados de seu desempenho em faces humanas: em [1], o trabalho no qual o detector D se embasa, o objeto de interesse eram imagens de traseiras de carros.

É importante relembrar que todos os classificadores foram impulsionados com o mesmo conjunto de dados. Isso sugere que, resguardadas as questões já mencionadas sobre os detectores D e E, os classificadores fortes dos detectores B e C têm maior capacidade de generalização, i.e., de desempenharem bem em situações reais após serem treinados com uma amostra dos dados, do que os usados em A, D e E.

Resta apenas discutir a conformidade das características contidas no classificador forte com as hipóteses propostas nesta dissertação. Para tal análise, focou-se nas características usadas pelos 20 primeiros classificadores fracos do detector B (figura 5.14). Restringiu-se o escopo deste estudo dessa forma pois os classificadores fortes usados em B e C foram impulsionados a partir do mesmo conjunto de características baseadas em Haar *wavelets*, mas somente o detector B apresentou alta precisão, portanto ele pode ter a maior aderência à hipótese.

Esse classificador forte usa 3 características com 2 retângulos, 11 com 3 retângulos e 6 com 4. Usando as instâncias negativas de treinamento, seus respectivos $S^-$ foram produzidos, suas matrizes de covariâncias calculadas e seus gráficos plotados. Nas figuras 5.15, 5.16 e 5.17 é possível ver o $S^-$ das características usadas pelos classificadores fracos 4, 10, 13, 16 e 20. As características 10 e 16, respectiva-

mente compostas de 4 e 3 retângulos, foram escolhidas pois apresentam as maiores variâncias entre as que possuem as mesmas quantidades de retângulos.

Todas essas características, em todas as suas dimensões, apresentam uma concentração de pontos ao longo do eixo $x = y$, e principalmente quando seus valores estão abaixo de $0,6$. Ainda que os pontos de $S^-$ efetivamente se espalhem por todo o ECRS, o comportamento mencionado é dominante. As outras características não presentes nessas ilustrações também apresentaram distribuição similar.

Essas observações parecem sugerir que os conjuntos $S^-$ contidos no classificador forte do detector B podem ser bem modelados com distribuições gaussianas, o que contradiz a hipótese levantada na seção 4.2.

Há diversas formas de avaliar objetivamente a gaussianidade de distribuições aleatórias, das quais se destacam a curtose e a negentropia, mas, por restrições de tempo, foi inviável estudá-las o suficiente para aplicá-las aqui. A curtose é o quarto momento de uma variável aleatória e mede o quão "pontiaguda" ou "achatada" ela é. É possível mostrar que distribuições gaussianas têm curtose zero [22]. A negentropia reflete o quão distante da entropia máxima está uma variável aleatória. Em teoria da informação, demonstra-se que a distribuição gaussiana tem a entropia máxima, portanto bastaria subtrair a entropia de uma variável aleatória da entropia da distribuição gaussiana para se obter a negentropia, que, quanto mais próxima de zero, mais similar à distribuição gaussiana será a variável aleatória [22].

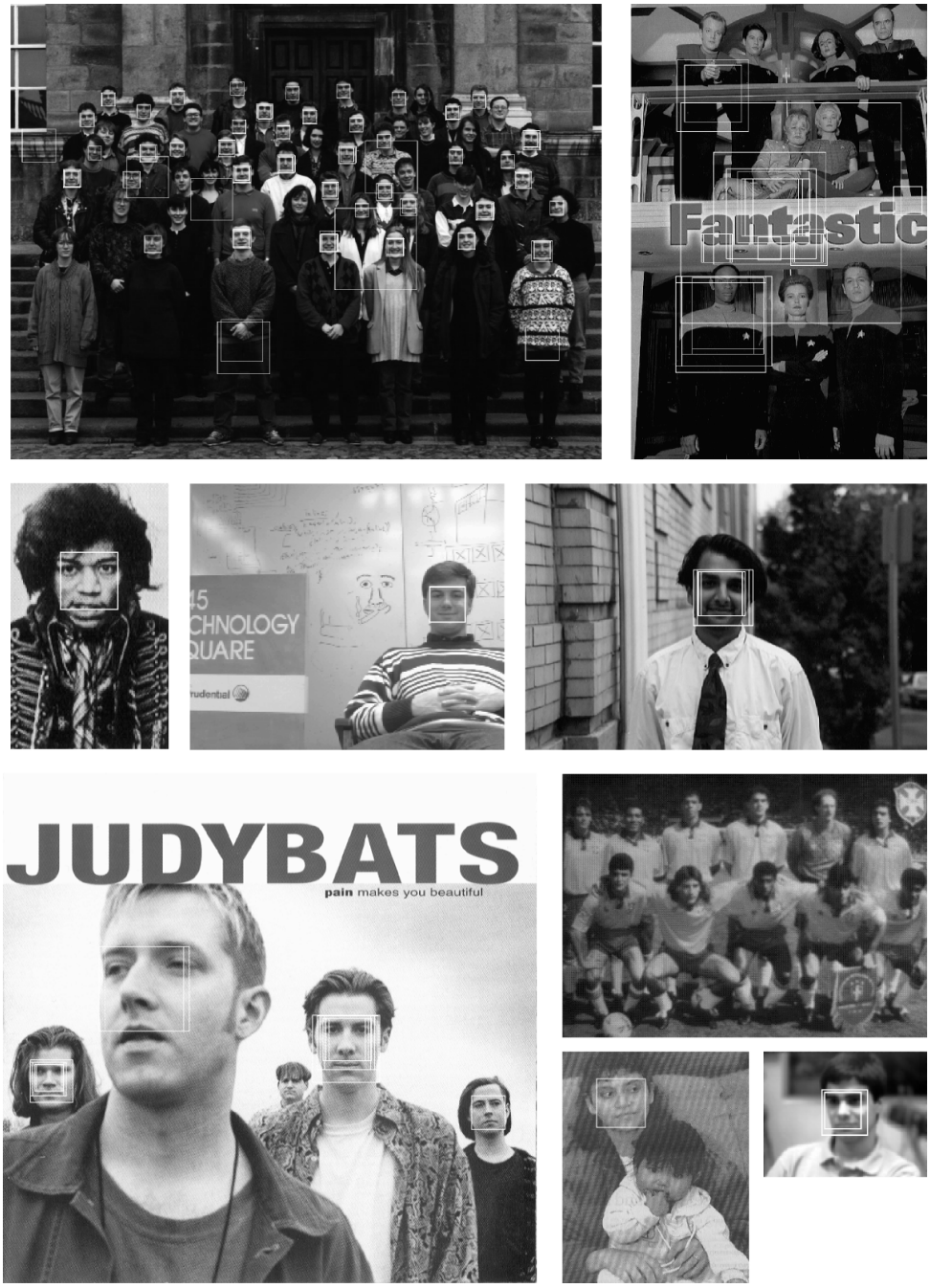


Figura 5.12: Detecções feitas com o detector B com limiar igual a 16 em imagens da base de MIT + CMU. Diversas ocorrências verdadeiras positivas e falsas positivas estão presentes nas imagens. Há imagens em que nenhuma face foi detectada, apesar de haver várias nelas. Por fim, há situações onde múltiplas detecções de uma única face ocorreram.

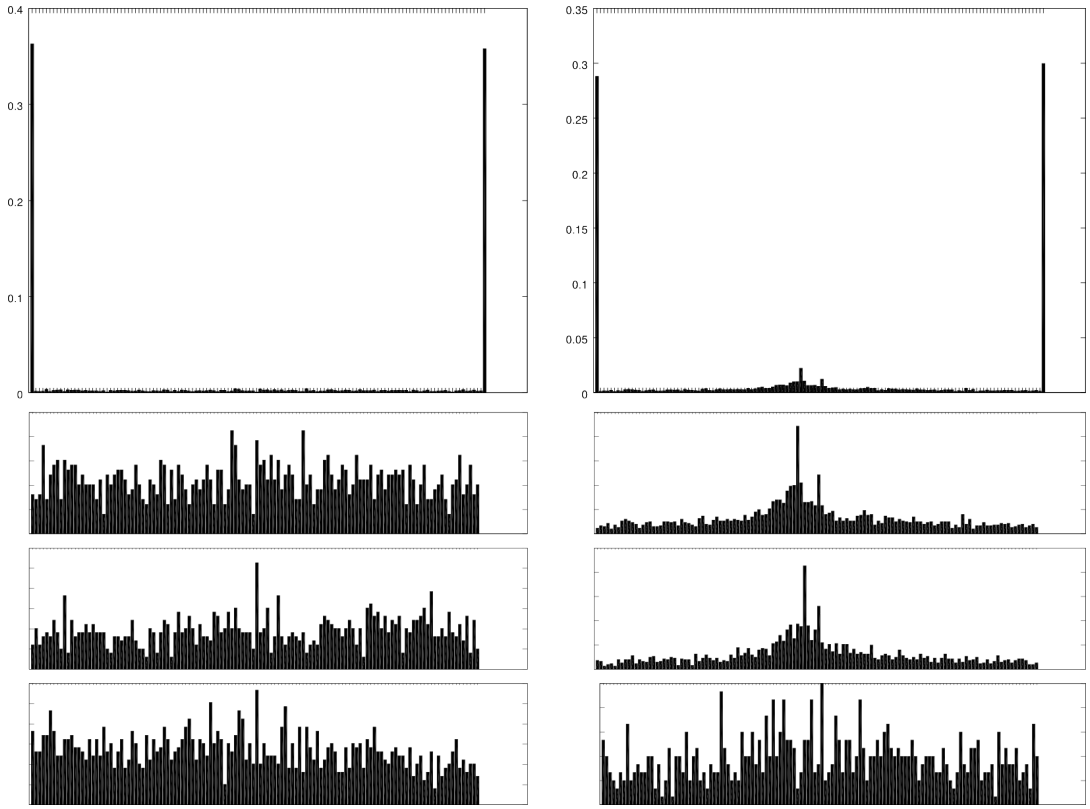


Figura 5.13: Histogramas dos valores de características dos classificadores fracos do detector E. À esquerda estão as distribuições das instâncias positivas, e à direita das negativas. Os histogramas na primeira linha pertencem ao primeiro classificador fraco adicionado ao classificador forte, e contém todos os intervalos do histograma. As linhas seguintes referem-se aos três primeiros classificadores fracos, mas sem o primeiro e o último intervalos para permitir a visualização de detalhes.

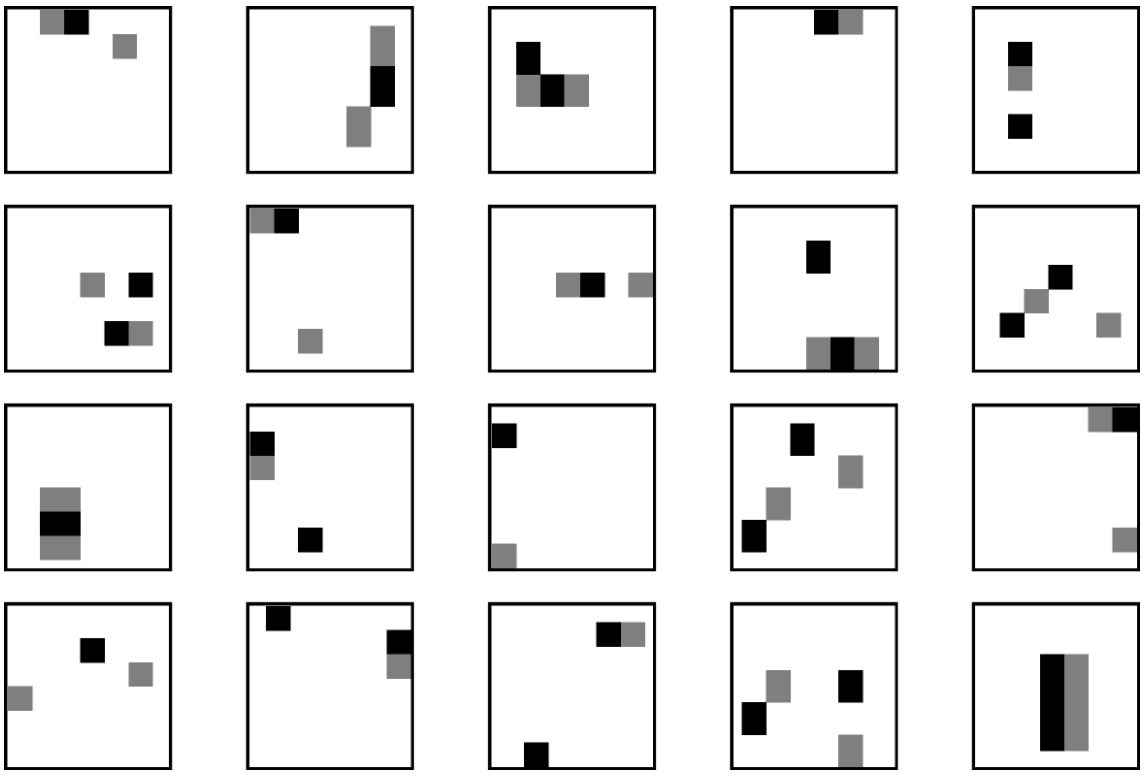

Figura 5.14: As 20 primeiras características do classificador forte do detector A, ordenadas de cima para baixo e da esquerda para direita.

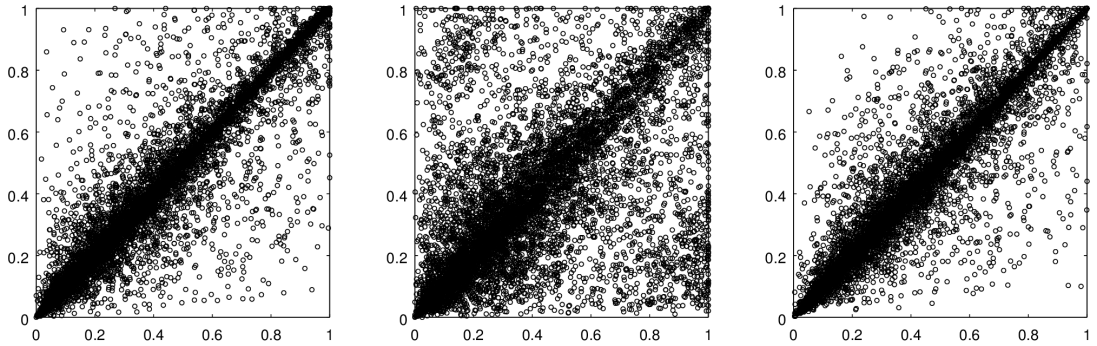


Figura 5.15: Da esquerda para a direita, $S^-$ das características com 2 retângulos usadas pelos classificadores fracos 4, 13 e 20 do detector B.

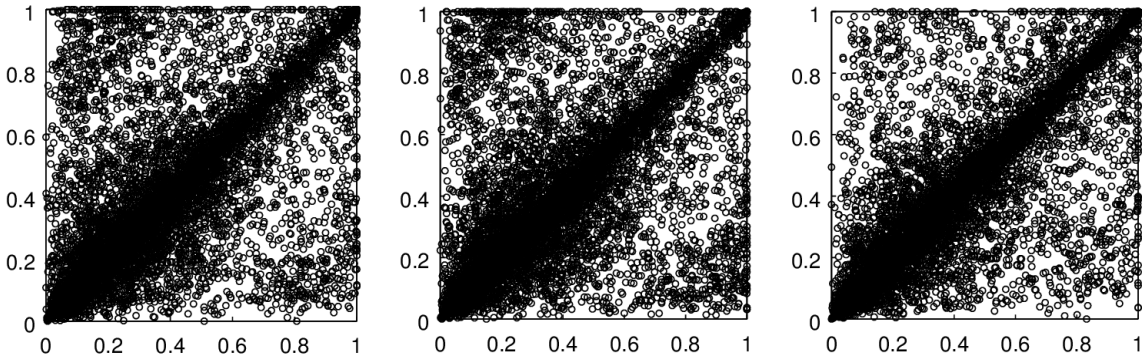


Figura 5.16: $S^-$ da característica dotada de 3 retângulos e usada no 16º classificador fraco do detector B. Para melhor visualização, as três dimensões desse ECRS estão apresentadas em 3 gráficos bidimensionais.

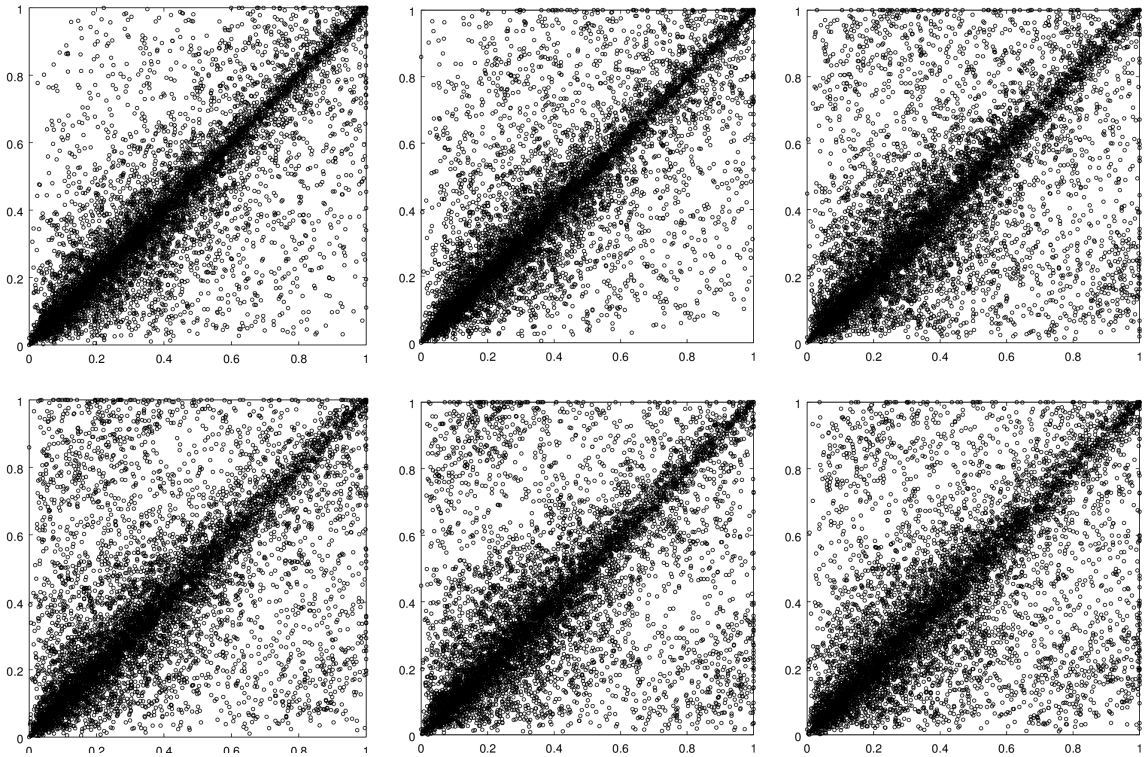


Figura 5.17: $S^-$ da característica dotada de 4 retângulos e usada no 10º classificador fraco do detector B. Para melhor visualização, as quatro dimensões desse ECRS estão apresentadas em 6 gráficos bidimensionais.

# 6 CONCLUSÃO

Nesta dissertação, cinco detectores de faces conformantes com o arcabouço de Viola e Jones [60] foram produzidos, testados e comparados. Dois desses detectores foram propostos aqui, estendendo o trabalho de Pavani e colaboradores [41], enquanto os demais serviram como experimentos de controle e se basearam no detector original de Viola e Jones e nos detectores de Adhikari [1], Rasolzadeh [45] e seus respectivos colaboradores.

Os classificadores fracos utilizados dos detectores propostos foram inspirados no trabalho de Pavani et al. que explora a distribuição de pontos no espaço vetorial formado pelas características baseadas em Haar *wavelets*, chamado de ECRS. A diferença é que, enquanto esses pesquisadores assumem que as distribuições nesse espaço vetorial de pontos oriundos de faces e de fundos são gaussianas, aqui assume-se que as distribuições de pontos de faces são gaussianas, mas a de fundos são uniformes. Com efeito, isso permite a adoção de uma abordagem de treinamento mais simples, dispensando certas rotinas de Pavani et al., e acelerando o treinamento.

Todos os detectores foram treinados com as mesmas instâncias negativas e positivas, e testados contra a mesma base de imagens. A partir da análise das curvas ROC produzidas durante os testes, evidenciou-se a precisão superior dos detectores propostos. Contudo, cabe observar que foi necessário arbitrar alguns parâmetros importantes para o detector de Rasolzadeh et al. visto que esses autores não os divulgaram. Também é importante notar que Rasolzadeh et al. testaram o detector que propuseram na detecção de traseiras de carro.

Muita atenção foi dada no fornecimento dos detalhes que permitem a repetição dos experimentos mencionados aqui por outros pesquisadores. Além disso, as bases de instâncias positivas usadas são acessíveis para o público em geral ou para uso em pesquisas científicas, e a base de instâncias negativas, criada pelo autor deste trabalho, também está disponível para uso por outros pesquisadores. O mesmo

ocorre para todo o código-fonte escrito.

Como os dois detectores propostos trabalham sobre os mesmos ECRSs, avaliou-se visualmente o comportamento dos pontos em seus respectivos espaços vetoriais originados pelos classificadores fracos selecionados para compôr o classificador forte. Tal análise concluiu que, apesar dos pesos terem sido atribuídos assumindo a uniformidade dos ECRSs, as características baseadas em Haar *wavelets* que participam do classificador forte formam de fato distribuições gaussianas.

Em conclusão, modelar a distribuição de pontos no ECRS como gaussianas parece ser a melhor abordagem quando o intuito é otimizar os pesos das características baseadas em Haar *wavelets* na tarefa de detecção de faces.

## 6.1 Trabalhos futuros

Ao longo desta dissertação, alguns trabalhos interessantes não foram feitos por limitações de tempo, por não terem relação direta ou por apenas tangenciarem o assunto principal. Esta seção apresenta alguns trabalhos relacionados com o tema desta dissertação e que poderiam ser feitos futuramente.

Uma extensão óbvia seria a construção duma cascata de classificadores fortes usando os classificadores fracos sugeridos. Ao fazer isso, seria interessante implementar e comparar os detectores propostos com aqueles apresentados por Pavani et al. [41] — comparação esta que também não foi feita neste trabalho.

Durante os experimentos, observou-se que o domínio do histograma do classificador fraco similar ao proposto por Rasolzadeh et al. [45] foi pequeno demais. Além disso, neste mesmo experimento, assumiu-se um valor para as probabilidades *a priori* que podem ter afetado seu desempenho negativa ou positivamente. Note que esses dois parâmetros não foram explicitados por seus autores. Por fim, Rasolzadeh et al. apontam que o ideal seria produzir histogramas com intervalos variáveis, considerando o nível de ruído das instâncias de treinamento. Esse é um trabalho interessante pois os algoritmos de impulsionamento podem perder desempenho se treinados com uma massa de dados ruidosa [29], e é possível que tal tratamento

contorne essa limitação.

Para o impulsionamento, utilizou-se um pouco mais que 15.000 instâncias rotuladas, mas Viola e Jones, para criar um classificador monolítico, utilizaram 20.000 instâncias. Um trabalho interessante seria aumentar a quantia de instâncias de faces e fundo, e comparar os resultados. Também seria interessante observar os efeitos sobre os classificadores produzidos com instâncias negativas selecionadas com critérios diferentes dos usados aqui.

Uma empreitada interessante seria implementar o impulsionamento em conjunto com o aprendiz fraco apresentado nesta dissertação em CUDA™ ou outra plataforma de processamento paralelo. Tal tarefa parece ser desafiadora pois, até onde se pesquisou, não há artigos públicos relatando-a. Porém, como cada classificador fraco pode ser produzido de forma independente, pode ser possível alcançar ganhos elevados de desempenho.

Outra extensão deste trabalho poderia ser a mensuração com negentropia ou curtose os ECRSs produzidos com cada classificador fraco. Isso daria uma visão mais objetiva sobre a gaussianidade das distribuições de pontos em seus espaços vetoriais.

Os programas usados neste trabalho foram escritos em C++ procurando fazer o melhor uso de técnicas de orientação a objetos, e dos recursos de gabaritagem dessa linguagem. Visava-se a implementação os algoritmos de impulsionamento e de aprendizagem fraca de tal maneira que fosse simples aplicá-lo em domínios diferentes da detecção de objetos em imagens. Apesar desse objetivo não ter sido alcançado, deu-se passos sólidos em sua direção. Completá-lo demanda apenas um pouco mais de esforço de projeto e codificação.

Para encorajar não somente a repetição desses experimentos, como também a execução por outros pesquisadores desses e de outros trabalhos de extensão ou relacionados, os códigos-fonte e dados resultantes dos esforços empreendidos estão disponíveis para acesso público.

# REFERÊNCIAS

[1] ADHIKARI, S. P.; YOO, H.-J.; KIM, H. Boosting-based on-road obstacle sensing using discriminative weak classifiers. *Sensors (Basel)*, v. 11, n. 4, p. 4372–84, 2011. ISSN 1424-8220.

[2] AMARAL, V. et al. *Normalização Espacial de Imagens Frontais de Face em Ambientes Controlados e Não-Controlados*. São Bernardo do Campo, São Paulo, Brasil, 2009. Disponível em: <http://fei.edu.br/ cet/facedatabase.html>.

[3] AMARAL, V.; THOMAZ, C. E. *Normalização Espacial de Imagens Frontais de Face*. São Bernardo do Campo, São Paulo, Brasil, 2008. Disponível em: <http://fei.edu.br/ cet/facedatabase.html>.

[4] BAKER, S.; NAYAR, S. K. Pattern rejection. In: *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1996.* [S.l.: s.n.], 1996. p. 544–549. ISSN 1063-6919.

[5] BAUMANN, F. et al. Symmetry enhanced adaboost. In: BEBIS, G. et al. (Ed.). *Advances in Visual Computing*. [S.l.]: Springer Berlin Heidelberg, 2010. v. 6453, cap. Lecture Notes in Computer Science, p. 286–295. ISBN 978-3-642-17288-5.

[6] CASTRILLÓN, M. et al. A comparison of face and facial feature detectors based on the viola–jones general object detection framework. *Machine Vision and Applications*, Springer-Verlag, v. 22, n. 3, p. 481–494, 2011. ISSN 0932-8092.

[7] CHAVES, B. B. *Estudo do algoritmo Adaboost em aprendizagem de máquina aplicado a sensores e sistemas embarcados*. Dissertação (Mestrado) — Escola Politécnica, Universidade de São Paulo, São Paulo, 2011.

[8] CHAWLA, N. et al. Smoteboost: Improving prediction of the minority class in boosting. In: LAVRAČ, N. et al. (Ed.). *Knowledge Discovery in Databases:*

*PKDD 2003*. [S.l.]: Springer Berlin Heidelberg, 2003. v. 2838, cap. Lecture Notes in Computer Science, p. 107–119. ISBN 978-3-540-20085-7.

[9] CHEN YEFEI; SU, J. Fast eye localization based on a new haar-like feature. In: *10th World Congress on Intelligent Control and Automation (WCICA), 2012*. [S.l.: s.n.], 2012. p. 4825–4830.

[10] CROW, F. C. Summed-area tables for texture mapping. In: *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*. Nova Iorque, NY, EUA: ACM, 1984. (SIGGRAPH '84), p. 207–212. ISBN 0-89791-138-5.

[11] DEMBSKI, J. Feature type and size selection for adaboost face detection algorithm. In: CHORAŚ, R. (Ed.). *Image Processing and Communications Challenges 2*. [S.l.]: Springer Berlin Heidelberg, 2010. v. 84, cap. Advances in Intelligent and Soft Computing, p. 143–149. ISBN 978-3-642-16294-7.

[12] DUDA, R. O.; HART, P. E.; STORK, D. G. *Pattern Classification (2nd Edition)*. [S.l.]: Wiley-Interscience, 2000. ISBN 0471056693.

[13] FAWCETT, T. An introduction to ROC analysis. *Pattern Recognition Letters*, v. 27, n. 8, p. 861 – 874, 2006. ISSN 0167-8655. ROC Analysis in Pattern Recognition.

[14] FREUND, Y. Boosting a weak learning algorithm by majority. *Information and Computation*, Academic Press, Inc., Duluth, MN, USA, v. 121, n. 2, p. 256–285, set 1995. ISSN 0890-5401.

[15] FREUND, Y.; SCHAPIRE, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. In: *Proceedings of the Second European Conference on Computational Learning Theory*. Londres, Reino Unido: Springer-Verlag, 1995. (EuroCOLT '95), p. 23–37. ISBN 3-540-59119-2.

[16] FREUND, Y.; SCHAPIRE, R. E. Experiments with a new boosting algorithm. In: *International Conference on Machine Learning*. [S.l.: s.n.], 1996. p. 148–156.

[17] FREUND, Y.; SCHAPIRE, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, v. 55, n. 1, p. 119–139, 1997. ISSN 0022-0000.

[18] FREUND, Y.; SCHAPIRE, R. E. A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, v. 14, p. 771–790, 1999.

[19] FRIEDMAN, J. H. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, v. 29, p. 1189–1232, 2000.

[20] GUO, Z. et al. Htf-boosting learning and face detection. In: *Pacific-Asia Workshop on Computational Intelligence and Industrial Application, 2008. PACIIA '08.* [S.l.: s.n.], 2008. v. 1, p. 376–380.

[21] HAAR, A. Zur theorie der orthogonalen funktionensysteme. *Mathematische Annalen*, Springer-Verlag, v. 69, n. 3, p. 331–371, 1910. ISSN 0025-5831.

[22] HYVÄRINEN, A.; KARHUNEN, J.; OJA, E. *Independent Component Analysis*. [S.l.]: Wiley, 2001. (Adaptive and Learning Systems for Signal Processing, Communications and Control Series). ISBN 9780471405405.

[23] ISHII, I.; ICHIDA, H.; TAKAKI, T. Gpu-based face tracking at 500 fps. In: *18th IEEE International Conference on Image Processing (ICIP), 2011.* [S.l.: s.n.], 2011. p. 557–560. ISSN 1522-4880.

[24] JESORSKY, O.; KIRCHBERG, K. J.; FRISCHHOLZ, R. W. Robust face detection using the hausdorff distance. In: *Proceedings of the Third International Conference on Audio and Video-Based Biometric Person Authentication*. [S.l.]: Springer, 2001. p. 90–95.

[25] JONES, M.; VIOLA, P. Fast multi-view face detection. *Mitsubishi Electric Research Lab TR-20003-96*, v. 3, p. 14, 2003.

[26] JUN, B.; KIM, D. Robust face detection using local gradient patterns and evidence accumulation. *Pattern Recognition*, v. 45, n. 9, p. 3304–3316, 2012. ISSN 0031-3203. Best Papers of Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA'2011).

[27] JUNIOR, L. L. de O.; THOMAZ, C. E. *Captura e Alinhamento de Imagens: Um Banco de Faces Brasileiro*. São Bernardo do Campo, São Paulo, Brazil, 2006. Disponível em: <http://fei.edu.br/ cet/iniciacaocientifica_leooliveira_2006.pdf>.

[28] KEARNS, M. et al. On the learnability of boolean formulae. In: *Proceedings of the nineteenth annual ACM symposium on Theory of computing*. New York, NY, USA: ACM, 1987. (STOC '87), p. 285–295. ISBN 0-89791-221-7.

[29] KHOSHGOFTAAR, T. M.; HULSE, J. V.; NAPOLITANO, A. Comparing boosting and bagging techniques with noisy and imbalanced data. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, IEEE, v. 41, n. 3, p. 552–568, may 2011. ISSN 1083-4427.

[30] LANDESA-VAZQUEZ, I.; ALBA-CASTRO, J. L. The role of polarity in haar-like features for face detection. In: *Proceedings of the 2010 20th International Conference on Pattern Recognition*. Washington, DC, EUA: IEEE Computer Society, 2010. (ICPR '10), p. 412–415. ISBN 978-0-7695-4109-9.

[31] LI, S. Z. et al. Statistical learning of multi-view face detection. In: *In Proceedings of the 7th European Conference on Computer Vision*. [S.l.: s.n.], 2002. p. 67–81.

[32] LIENHART, R.; MAYDT, J. An extended set of haar-like features for rapid object detection. In: *IEEE ICIP 2002*. [S.l.: s.n.], 2002. p. 900–903.

[33] LYONS MIYUKI KAMACHI, J. G. M. J. *Japanese Female Facial Expressions (JAFFE), Database of digital images*. 1997. Acesso em: julho de 2014. Disponível em: <http://kasrl.org/jaffe.html>.

[34] MAGALHÃES, R. P. de; LIMA, C. A new method for haar-like features weight adjustment using principal component analysis for face detection. In: *ICONS 2014, The Ninth International Conference on Systems*. [S.l.: s.n.], 2014. p. 55–62.

[35] MARTINEZ, A. M.; BENAVENTE, R. *The AR Face Database*. [S.l.], 1998. Disponível em: <http://www2.ece.ohio-state.edu/ aleix/ARdatabase.html>.

[36] MIT Center for Biological and Computation Learning. *CBCL Database # 1*. 2000. Acesso em: junho de 2013. Disponível em: <http://www.ai.mit.edu/projects/cbcl>.

[37] National Institute of Standards and Technology. *Face Recognition Technology (FERET)*. Acesso em: abril de 2014. Disponível em: <http://www.nist.gov/itl/iad/ig/feret.cfm>.

[38] Octave community. *GNU Octave 3.8.1*. 2014. Acesso em: setembro de 2013. Disponível em: <www.gnu.org/software/octave/>.

[39] PAPAGEORGIOU, C.; OREN, M.; POGGIO, T. A general framework for object detection. In: *ICCV*. Washington, DC, EUA: IEEE Computer Society, 1998. (ICCV '98), p. 555–562. ISBN 81-7319-221-9.

[40] PAVANI, S.-K.; GOMEZ, D. D.; FRANGI, A. F. Gaussian weak classifiers based on haar-like features with four rectangles for real-time face detection. In: JIANG, X.; PETKOV, N. (Ed.). *Computer Analysis of Images and Patterns*. [S.l.]: Springer Berlin Heidelberg, 2009, (Lecture Notes in Computer Science, v. 5702). p. 91–98. ISBN 978-3-642-03766-5.

[41] PAVANI, S.-K.; GOMEZ, D. D.; FRANGI, A. F. Haar-like features with optimally weighted rectangles for rapid object detection. *Pattern Recognition*, Elsevier Science Inc., Nova Yorque, NY, EUA, v. 43, n. 1, p. 160–172, jan 2010. ISSN 0031-3203.

[42] PEARSON, K. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, v. 2, n. 6, p. 559–572, 1901.

[43] RAMOS, R. M. *Identificação de comunicado de ocorrência de perdas em seguro agrícola utilizando algoritmos de inteligência artificial.* Dissertação (Mestrado) — Universidade de Brasília, 2008.

[44] RANJAN, A.; MALIK, S. Parallelizing a face detection and tracking system for multi-core processors. In: *Computer and Robot Vision (CRV), 2012 Ninth Conference on.* [S.l.: s.n.], 2012. p. 290–297.

[45] RASOLZADEH, B.; PETERSSON, L.; PETTERSSON, N. Response binning: Improved weak classifiers for boosting. In: *IEEE Intelligent Vehicles Symposium, 2006.* [S.l.: s.n.], 2006. p. 344–349.

[46] ROE, B. P. et al. Boosted decision trees as an alternative to artificial neural networks for particle identification. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, v. 543, n. 2-3, p. 577–584, 2005. ISSN 0168-9002.

[47] ROWLEY, H.; BALUJA, S.; KANADE, T. Neural network-based face detection. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1996.* [S.l.: s.n.], 1996. p. 203–208. ISSN 1063-6919.

[48] ROWLEY, H. A. et al. Neural network-based face detection. *IEEE Transactions On Pattern Analysis and Machine intelligence*, v. 20, p. 23–38, 1998.

[49] SANDERSON, C. *Armadillo: An Open Source C++ Linear Algebra Library for Fast Prototyping and Computationally Intensive Experiments*. [S.l.], 2010. Disponível em: <http://arma.sourceforge.net/armadillo_nicta_2010.pdf>.

[50] SCHAPIRE, R. E. The strength of weak learnability. *Machine Learning*, Kluwer Academic Publishers, Hingham, MA, USA, v. 5, n. 2, p. 197–227, jul 1990. ISSN 0885-6125.

[51] SCHAPIRE, R. E. The boosting approach to machine learning: An overview. In: *Nonlinear estimation and classification*. [S.l.]: Springer, 2003. p. 149–171.

[52] SCHAPIRE, R. E.; FREUND, Y. *Boosting: Foundations and Algorithms*. [S.l.]: The MIT Press, 2012. ISBN 0262017180, 9780262017183.

[53] SCHNEIDERMAN, H. W. *A statistical approach to 3d object detection applied to faces and cars*. Tese (Doutorado) — Carnegie Mellon University, Pittsburgh, PA, USA, 2000. AAI9986625.

[54] SEIFFERT, C. et al. Resampling or reweighting: A comparison of boosting implementations. In: *Tools with Artificial Intelligence, 2008. ICTAI '08. 20th IEEE International Conference on*. [S.l.: s.n.], 2008. v. 1, p. 445–451. ISSN 1082-3409.

[55] SEIFFERT, C. et al. Rusboost: A hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, v. 40, n. 1, p. 185–197, Jan 2010. ISSN 1083-4427.

[56] SHEN, J. et al. A novel distribution-based feature for rapid object detection. *Neurocomputing*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 74, n. 17, p. 2767–2779, oct 2011. ISSN 0925-2312.

[57] STRANG, G. *Introduction to Linear Algebra*. Wellesley, MA: Wellesley-Cambridge Press, 2009. ISBN 0980232716.

[58] SUNG, K.-K.; POGGIO, T. Example-based learning for view-based human face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 20, n. 1, p. 39–51, 1998. ISSN 0162-8828.

[59] VALIANT, L. G. A theory of the learnable. *Communications of the ACM*, ACM, New York, NY, USA, v. 27, n. 11, p. 1134–1142, nov 1984. ISSN 0001-0782.

[60] VIOLA, P.; JONES, M. J. Robust real-time face detection. *International Journal of Computer Vision*, Kluwer Academic Publishers, Hingham, MA, USA, v. 57, n. 2, p. 137–154, may 2004. ISSN 0920-5691.

[61] VIOLA, P. A.; JONES, M. J. Rapid object detection using a boosted cascade of simple features. In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001. CVPR 2001.* [S.l.: s.n.], 2001. p. 511–518.

[62] VURAL, S. et al. Multi-view fast object detection by using extended haar filters in uncontrolled environments. *Pattern Recognition Letters*, v. 33, n. 2, p. 126–133, 2012. ISSN 0167-8655.

[63] WONG, W.-S.; CHEN, C.-R.; CHIU, C.-T. A 100mhz hardware-efficient boost cascaded face detection design. In: *16th IEEE International Conference on Image Processing (ICIP), 2009.* [S.l.: s.n.], 2009. p. 3237–3240. ISSN 1522-4880.

[64] WU, J. et al. Fast asymmetric learning for cascade face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 30, n. 3, p. 369–382, 2008. ISSN 0162-8828.

[65] YILMAZ, A.; JAVED, O.; SHAH, M. Object tracking: A survey. *ACM Computing Surveys (CSUR)*, ACM, New York, NY, USA, v. 38, n. 4, p. 13, dezembro 2006. ISSN 0360-0300.

[66] ZHANG, C.; ZHANG, Z. *A Survey of Recent Advances in Face detection*. [S.l.], 2010. Disponível em: <http://research.microsoft.com/apps/pubs/default.aspx?id=132077>.

[67] ZIMMERMANN, G. *On the Theory of Orthogonal Function Systems*. Acesso em: novembro de 2012. Disponível em: <https://www.uni-hohenheim.de/ gzim/Publications/haar.pdf>.