Universidade Federal de Pernambuco
Centro de Informática

Graduação em Ciência da Computação

Erro Ponderado de Reconstrução com PCA para Detecção de Pedestres em Imagens

Lailson Bandeira de Moraes

Trabalho de Graduação

Recife
17 de dezembro de 2010

Universidade Federal de Pernambuco
Centro de Informática

Lailson Bandeira de Moraes

Erro Ponderado de Reconstrução com PCA para Detecção de Pedestres em Imagens

Trabalho apresentado ao Programa de Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: *George Darmiton da Cunha Cavalcanti*
Co-orientador: *Tsang Ing Ren*

Recife
17 de dezembro de 2010

Aos meus pais

Agradecimentos

Agradeço a Deus, que nos deu um universo tão vasto e interessante e, ao mesmo tempo, nos presenteou com inteligência e curiosidade para explorá-lo. Agradeço a Inez e Pedro, meus pais, pelo amor incodicional e pelo apoio que me fez chegar até aqui. Agradeço ao George, meu orientador, que pacientemente respondeu aos meus questionamentos e me inspira como pessoa e profissional. Agradeço a todos os amigos de curso, que fizeram dessa uma jornada muito mais divertida, em especial aos meus amigos Guilherme e Emanuel, parceiros de muitas empreitadas e pelos quais eu tenho grande admiração. Agradeço aos professores do Centro de Informática, que, além de conhecimento, me incutiram também a semente da excelência. Agradeço aos amigos da Infoway e da L. A. Informática, que apostaram em mim desde cedo e me ajudaram enormemente. Agradeço, enfim, a todos os meus amigos, inclusive aos que estão longe, que contribuem diariamente, de diferentes maneiras, para o meu sucesso.

Imagination will often carry us to worlds that never were.
But without it we go nowhere.
—CARL SAGAN (Cosmos)

Resumo

A detecção de pedestres é uma tarefa geralmente associada a sistemas de segurança e vigilância. O desenvolvimento de um sistema de detecção de pedestres é grande desafio, devido à grande quantidade de formas em que eles podem se apresentar. Neste trabalho, analisamos um modelo existente de detecção de pedestres baseado em erros de reconstrução com PCA que também usa imagens de borda para reduzir variação desnecessária. Investigamos como o método funciona e onde mudanças podem ser feitas para melhorar o desempenho original. As alterações propostas melhoram a acurácia do sistema através de pesos, que são encontrados de forma automatizada pelo uso de um algoritmo genético. Também observamos que alguns erros de reconstrução não são estritamente necessários e portanto podem ser eliminados para reduzir o tempo de classificação pela metade. Finalmente, testamos o classificador com diferentes métodos de borda para verificar como esta escolha afeta a performance do sistema.

Palavras-chave: detecção de pedestres, detecção de objetos, análise de componentes principais, visão computacional, inteligência artificial

Abstract

Pedestrian detection is a task usually associated with security and surveillance systems. The development of a pedestrian detection system poses a hard challenge, since they can assume many different forms. In this work, we present an analysis of an existing pedestrian detection model based on PCA reconstruction errors that also use edge images to reduce unecessary variation. We investigate how the method works and where changes can be made to improve its original performance. The proposed improvements enhance the system's accuracy by using weights, that are found in an automated way using a genetic algorithm. We also found that some reconstruction errors used by the original method are not strictly necessary and therefore can be eliminated to reduce the classifying time by half. Finally, we tested the classifier with different edge methods, to find out how this choice affects the system's performance.

Keywords: pedestrian detection, object detection, principal component analysis, computer vision, artificial intelligence

Sumário

1 Introdução **1**
1.1 Aplicações 1
1.2 Estado da Arte 2
1.3 Resumo 4

2 Análise de Componentes Principais (PCA) **5**
2.1 Definição 5
2.1.1 População e amostra da população 7
2.1.2 Exemplo 1 8
2.2 Discussão 8
2.2.1 Exemplo 2 10
2.3 Erro de Reconstrução 11
2.4 Algoritmo 13
2.5 Considerações Históricas e Outras Aplicações 13

3 Classificação por Reconstrução com PCA **15**
3.1 Sistema de Detecção 16
3.1.1 Reconstrução de Imagens com PCA 16
3.1.2 Imagens de Borda 17
3.1.3 Classificação por Reconstrução com PCA 17
3.2 Papel das Imagens de Não-Pedestres 20
3.3 Papel das Imagens de Borda 20
3.4 Erro de Reconstrução Ponderado 21

4 Experimentos **23**
4.1 Autoespectros 24
4.2 Imagens de Pedestres *versus* Imagens de Não-Pedestres 24
4.3 Imagens em Escala de Cinza *versus* Imagens de Borda 27
4.4 Erro Total Ponderado 30
4.5 Comparação Entre os Algoritmos de Borda 32

5 Conclusão **39**

Lista de Figuras

2.1 Aplicação do PCA em um conjunto de dados bidimensional 8

2.2 Aplicação do PCA em um problema de classificação 10

2.3 Autoesptecro 12

3.1 Imagens em escala de cinza de pedestres (acima) acompanhadas da imagem de borda correspondente (abaixo). Observe que apesar da cor e do fundo das imagens originais serem bastante distintos, as imagens de borda apresentam pouca diferença de uma para outra. As imagens de borda foram obtidas pela soma das magnitudes dos filtros Sobel horizontal e vertical (equação 4.1); um comportamento semelhante é observado com os outros filtros testados. 17

3.2 Quatro imagens usadas para treinamento do classificador, duas de pedestres e duas de não-pedestres, acompanhadas de suas imagens de borda e reconstruções. A primeira linha (a) mostra as imagens originais em escala de cinza com suas respectivas imagens de borda, obtidas com o filtro Sobel. A segunda linha (b) mostra as imagens reconstruídas usando 100 PCs positivos (P_{gp} para imagens em escala de cinza e P_{ep} para imagens de borda). A terceira linha (c) mostra as imagens reconstruídas usando 100 PCs negativos (P_{gn} para imagens em escala de cinza e P_{en} para imagens de borda). Observe que as imagens de pedestres são melhor reconstruídas com os componentes principais do conjunto positivo. 18

4.1 Autoespectros dos 600 primeiros componentes principais do conjunto de imagens em escala de cinza e de bordas de pedestres (gráficos superior e inferior esquerdo, respectivamente) e do conjunto de imagens em escala de cinza e de bordas de não-pedestres (gráficos superior e inferior direito, respectivamente). O eixo vertical representa os autovalores e o eixo horizontal o número de PCs (k). Os autovalores decaem muito rapidamente (o eixo y está em escala logarítmica), indicando que a maior parte da informação do conjunto está concentrada nos primeiros PCs. Observe que as imagens de não-pedestres possuem uma curva de queda ligeiramente mais suave, especialmente para os maiores valores de k; isto é reflexo da maior variabilidade entre as imagens do conjunto. Note também que os conjuntos de imagens de borda concentram ainda menos informações nos maiores PCs (o menor valor da escala vertical dos gráficos inferiores é 10^{-3}), resultado da eliminação de grande parte da variabilidade desnecessária deste conjunto. As imagens de borda foram obtidas com o filtro Sobel; um comportamento semelhante é observado para os demais algoritmos testados. 25

4.2 Comparação da distribuição dos erros de reconstrução positivos e negativos das amostras do conjunto de teste para $k = 100$. O primeiro gráfico mostra a distribuição dos erros de reconstrução parcial positivos; o segundo mostra a distribuição dos erros de reconstrução parcial negativos; e o terceiro mostra a distribuição dos erros de reconstrução total. Observe que as amostras estão completamente misturadas nos dois primeiros gráficos: é praticamente impossível separá-las usando apenas estes erros. Por outro lado, o terceiro gráfico mostra dois grupos bem definidos, o que viabiliza uma boa classificação. 26

4.3 Comparação das curvas ROC positivas e negativas. Da esquerda para a direita: a primeira curva (verde) se refere ao classificador que usa o erro de reconstrução total (eq. 3.4) e tem AUC de 0,9936; a segunda curva (vermelha) se refere ao classificador que usa apenas os PCs negativos (erro de reconstrução parcial negativo, eq. 3.8) e tem AUC de 0,6900; e a terceira curva (azul) se refere ao classificador que usa apenas os PCs positivos (erro de reconstrução parcial positivo, eq. 3.7) e tem AUC de 0,4476. Nos três classificadores, apenas os 100 primeiros componentes principais são usados, isto é, $k = 100$. Ambos os classificadores positivos e negativos obtêm um resultado ruim, longe do desempenho do classificador completo. O mais surpreendente é que o classificador negativo acaba gerando um resultado melhor que o classificador positivo. As imagens de borda foram obtidas com o filtro Sobel; um comportamento semelhante é observado para os demais algoritmos testados. 27

4.4 Comparação das curvas ROC do classificador que usa imagens em escala de cinza e do que usa imagens de bordas. Da esquerda para a direita: a curva verde se refere ao classificador que usa o erro de reconstrução total (eq. 3.4) e tem AUC de 0,9936; a curva vermelha se refere ao classificador que usa o erro parcial de bordas (eq. 3.10) e tem AUC de 0,9943; e a curva azul se refere ao classificador que usa o erro parcial de escala de cinza (eq. 3.9) e tem AUC de 0,9846. Nos três classificadores, $k = 100$. Observe que a escala dos eixos é diferente da usada na Figura 4.2; o canto superior esquerdo foi aproximado para mostrar melhor a pequena diferença entre as curvas. As imagens de borda foram obtidas com o filtro Sobel. 28

4.5 Comparação da distribuição dos erros de reconstrução de escala de cinza e de bordas das amostras do conjunto de testes para $k = 100$. O primeiro gráfico mostra os erros de reconstrução parcial de escala de cinza; segundo mostra a distribuição dos erros de reconstrução parcial de bordas; e o terceiro mostra a distribuição dos erros de reconstrução total. Observe que os três gráficos apresentam uma clara fronteira de separação entre as amostras positivas e negativas. 29

4.6 Comparação entre as AUCs do classificador ponderado e do classificador original para o erro total, em diferentes valores de k. A diferença é mais pronunciada para maiores valores de k e se torna pequena quando k diminiu. Ainda assim, a AUC do classificador ponderado é sempre maior, o que indica que o erro ponderado é um critério de classificação melhor que o erro não-ponderado. Observe também que a linha do classificador ponderado se mantém mais estável com a variação de k. 33

4.7 Distribuição dos erros de reconstrução total ponderadados das amostras de teste, para $k = 100$. Compare com o terceiro gráfico da Figura 4.3: aqui a fronteira entre os erros das amostras positivas e negativas é bem mais clara. 37

4.8 Imagens de bordas geradas por cada técnica. A linha (a) mostra as imagens em escala de cinza originais; a linha (b) mostra as imagens de borda obtidas com o filtro Sobel binarizado; a linha (c) o filtro de Roberts; a linha (d) o filtro de Roberts binarizado; a linha (e) o filtro de Prewitt; a linha (f) o filtro de Prewitt binarizado; a linha (g) o filtro de Canny; e a linha (h) o filtro de Canny binarizado. 38

Lista de Tabelas

4.1 AUCs dos classificadores para diferentes quantidades de PCs (k). O primeiro classificador usa o erro de reconstrução parcial de escala de cinza; o segundo usa o erro de reconstrução parcial de bordas; e o terceiro usa o erro de reconstrução total. Observe que para $k \geq 100$, o classificador parcial de bordas apresenta um desempenho melhor que classificador total, apesar deste último possuir mais informações. As imagens de borda foram obtidas com o filtro Sobel. 30

4.2 Tempo médio de detecção em milissegundos de uma imagem, para cada classificador e quantidade de PCs (k). As detecções foram executadas nas mesmas imagens, sob as mesmas condições, usando o filtro Sobel para obter as imagens de borda. A diferença entre os tempos de detecção dos dois primeiros classificadores corresponde basicamente à computação das bordas. 30

4.3 Pesos obtidos com relação às amostras treinamento. O algoritmo genético foi executado separadamente para cada um dos três classificadores por cinco vezes para cada valor de k e o conjunto que produziu o melhor resultado foi selecionado. Os pesos funcionam efetivamente como uma medida da importância de cada erro. Observe que para os classificadores parciais os valores geralmente indicam que o conjunto negativo possui maior importância, corroborando o resultado sugerido pela Figura 4.2. Já no classificador do erro total, os pesos atribuem maior importância às imagens de borda, que foram obtidas com o filtro Sobel. 32

4.4 AUCs dos classificadores ponderados para diferentes quantidades de PCs (k). Para facilitar a comparação, as AUCs dos classificadores originais também são mostradas. Observe que no classificador que usa o erro total, a versão ponderada sempre apresenta um melhor desempenho. Já nos classificadores parciais, a versão original superou a ponderada apenas quando $k = 25$. 34

4.5 Tempo médio em milissegundos para computação de uma imagem de borda, para cada algoritmo. 34

4.6 AUCs dos classificadores que usam o filtro Sobel binarizado (à direita). Para facilitar a comparação, as AUCs do dos classificadores que usam o filtro Sobel na versão de gradientes, mostradas anteriormente, também são listadas (à esquerda). 35

4.7 AUCs dos classificadores que usam o filtro de Roberts (na versão de gradientes, à esquerda, e na versão binarizada, à direita), para diversos valores de k. 35
4.8 AUCs dos classificadores que usam o filtro de Prewitt (na versão de gradientes, à esquerda, e na versão binarizada, à direita), para diversos valores de k. 36
4.9 AUCs dos classificadores que usam o detector de Canny (na versão de gradientes, à esquerda, e na versão binarizada, à direita), para diversos valores de k. 36

CAPÍTULO 1

Introdução

Técnicas para detecção de pedestres têm sido largamente utilizadas e desenvolvidas no campo da visão computacional. Elas possuem diversas aplicações, desde monitoramento de câmeras para segurança a sistemas avançados de auxílio a motoristas. Ao mesmo tempo, a classe de pedestres é uma das mais difíceis de detectar. Sua aparência varia enormemente de acordo com o tipo e estilo de roupa que o pedestre está usando, fazendo com que apenas algumas regiões locais sejam realmente características de toda a categoria. Além disto, pedestres não têm aparência fixa: as possibilidades de articulação do corpo humano fazem com que sua silhueta possa assumir diversas formas. Pedestres podem ainda usar uma enorme variedade de acessórios oclusivos, como mochilas, bolsas, pastas e sacolas, que alteram dramaticamente sua silhueta. Por fim, em algumas aplicações várias pressoas podem estar presentes na mesma região, obstruindo parcialmente umas às outras, o que dificulta ainda mais o trabalho de detecção.

Neste trabalho, analisamos o funcionamento de um detector de pedestres em imagens que usa reconstrução com PCA como critério para classificação. Investigamos como e por que ele funciona e, a partir deste entendimento, propomos melhorias tanto para aumentar sua acurácia como também para diminuir o tempo de detecção, discutindo as trocas que devem ser feitas em cada caso. Avaliamos também o desempenho do classificador em diversos cenários. Todas as hipóteses são testadas empiricamente e os resultados obtidos são usados para a criação de novas hipóteses, que novamente são testadas através de experimentos.

Neste capítulo, mostramos algumas aplicações das técnicas de detecção de pedestres na Seção 1.1, descrevemos o estado da arte da área na Seção 1.2 e fazemos um resumo deste trabalho na Seção 1.3.

1.1 Aplicações

Devido à crescente preocupação com segurança, atualmente há uma grande demanda por sistemas eletrônicos de monitoramento. Os avanços tecnlógicos fazem com que equipamentos sofisticados sejam cada vez mais acessíveis para pessoas e organizações. Seja em áreas públicas ou em espaços privados, tais sistemas estão se tornando pervasivos na sociedade.

Além sensores diversos, sistemas eletrônicos de segurança geralmente contam com dispositivos de imagem, especialmente câmeras de vídeo de espectro visível. Com o barateamento destes dispositivos, hoje é comum encontrar sistemas compostos por várias câmeras de segurança, de modo a cobrir uma grande área de monitoramento.

No entanto, na maior parte dos sistemas a análise do material captado é feita de forma manual, por indivíduos que assistem aos vídeos continuamente, em uma bateria de monitores, à procura de atividades suspeitas ou de situações anormais. Além de estressante, esta análise é certamente ineficiente, já que a capacidade de concentração humana é limitada a um curto espaço de tempo e há muitos vídeos a serem observados simultaneamente [KD07].

Para lidar com este problema, alguns sistemas armazenam o material captado, de forma a usá-lo como ferramenta forense em caso de algum evento significativo. Mas até mesmo a análise posterior deste material pode ser difícil, principalmente quando não se sabe quando o evento ocorreu: vídeos de segurança tipicamente consistem em longos trechos de atividade normal intercaladas por rápidas passagens de eventos potencialmente suspeitos; encontrar estas passagens manualmente é uma atividade bastante laboriosa. Há, portanto, a necessidade de sistemas de análise automática ou pelo menos semi-automática, de forma a auxiliar profissionais na observação de vídeos de segurança. Em tais sistemas, uma atividade central é a deteccção de pedestres (também conhecida como detecção de humanos), que consiste basicamente em detectar a presença de pessoas em uma cena estática (imagem) ou em uma cena dinâmica (vídeo) e, opcionalmente, apontar sua localização [HTWM04].

Além do monitoramento de segurança, técnicas para detecção de pedestres podem ser aplicadas em diversos outros contextos, como em sistemas de contagem de pedestres para estimativa de fluxo ou em sistemas avançados de auxílio ao motorista (*advanced driver assistance systems*, ADAS, em inglês), para evitar atropelamentos ou diminuir sua severidade [GLSG10], por exemplo.

1.2 Estado da Arte

Várias abordagens foram introduzidas ao longo da última década para detecção de pedestres e um progresso significativo foi atingido em termos de robustez e eficiência neste campo. A maioria dos trabalhos usa o contorno ou a forma do pedestre como critério dominante para detecção em imagens devido à grande variabilidade que sua aparência pode apresentar. Em vídeos, o movimento tem sido largamente usado como um critério complementar.

Na literatura de detecção de objetos, esquemas de modelagem ou de extração de características de contornos podem ser grosseiramente classificados em duas categorias. A primeira modela formas globalmente, como características wavelet de Haar (*Haar wavelet features*, em inglês) [PP99], características retangulares [VJ01], histogramas de gradientes orientados (*histograms of oriented gradients*, HOG, em inglês) [DT05], modelos de Markov localmente deformáveis [WYH05] ou descritores de covariância [TPM07]. Tais abordagens são projetadas para tolerar algum grau de oclusão e articu-

lação das formas usando um grande número de amostras. Elas apresentam excelente desempenho em imagens mais ou menos alinhadas, com pedestres completamente visíveis. Em [GP99] e [ZD05], os contornos de pedestres são diretamente modelados como um conjunto de formas globais organizadas em uma árvore hierárquica. Devido à natureza da modelagem global, estas abordagens precisam de um grande número de imagens de treinamento positivas.

A segunda categoria modela formas usando características locais esparsas ou como uma coleção de partes visuais. Abordagens baseadas em características locais treinam detectores de partes do corpo com base em pontos de interesse esparsos e descritores [MSZ04] [LSS05], em redes de segmentos de contorno [FTVG06], segmentos *k*-adjacentes [FFJS08] ou edgelets [WYH05]. Abordagens baseadas em partes modelam contornos de objetos como uma configuração rígida ou deformável de partes visuais e têm se mostrado muito efetivas em casos de oclusões parciais. Em [MPP01], vários detectores são treinados separadamente para cada parte do corpo e combinados em um classificador de segundo-nível. Mikolajczyk et al. [MSZ04] usa características locais para detecção de partes e agrega estas detecções probabilisticamente. Wu e Nevatia [WYH05] introduziram *edgelet features* para detecção de pedestres e em seguida estenderam esta técnica para detecção e segmentação de objetos em geral projetando classificadores baseados em formas locais [WN07]. Shet et al. [SNRD07] propõe um método baseado em raciocínio lógico para juntar eficientemente detecções de partes.

Abordagens baseadas em partes deformáveis [FPZ03] [FMR08] [AT07] estabeleceram uma teoria probabilística elegante para modelar articulações de formas. Estas abordagens obtiveram bastante êxito em diversas tarefas de visão computacional, como categorização de objetos, reconhecimento de caracteres manuscritos e detecção de faces. Elas geralmente representam um objeto como uma coleção de partes e forçam certas restrições na sua configuração espacial, de acordo com as articulações do objeto. Amit e Trouve [AT07], por exemplo, propuseram um *framework* para reconhecimento de objetos baseado em partes (denominado *patchwork of parts*), que é capaz de detectar objetos e tratar oclusão precisamente. Este classificador foi aplicado com sucesso para reconhecimento de dígitos manuscritos e detecção de faces.

Abordagens globais requerem um único classificador, que precisa ser treinado apenas uma vez. São portanto mais simples que as abordagens baseadas em partes, onde vários classificadores devem ser treinados separadamente para cada parte, podendo incluir ainda um módulo de agregação, que também precisa de treinamento. Por outro lado, abordagens baseadas em partes (ou em características locais) lidam melhor com oclusões parciais e são mais flexíveis com relação a objetos articulados, quando comparadas com abordagens globais. Esquemas de modelagem de forma também podem ser combinados com técnicas baseadas em aparência para detecção e segmentação simultânea [KTZ05] [WS06] ou com movimento para detecção de pedestres [VJS03] [DTS06].

1.3 Resumo

O Capítulo 2 define formalmente a análise de componentes principais (PCA), um procedimento estatístico na qual o detector analisado por este trabalho é baseado. São mostradas duas derivações distintas da técnica juntamente com exemplos e discussões, de forma a facilitar seu entendimento. Este capítulo mostra também um algoritmo para computar o PCA de um conjunto de dados e faz considerações sobre a história da técnica e de suas aplicações. Por fim, nele ainda são definidos conceitos relacionados, como reconstrução e erro de reconstrução.

O Capítulo 3 descreve o classificador original e lança hipóteses sobre seu funcionamento, investigando por que ele funciona e como ele pode ser melhorado. Este capítulo avalia como os conjuntos de imagens usados no treinamento contribuem para a classificação e propõe um novo critério de classificação para melhorar a acurácia do sistema.

O Capítulo 4 descreve os experimentos executados, bem como o ambiente e o conjunto de dados em que eles foram realizados. Nele, discute-se também os resultados encontrados e suas implicações, que são usadas para criar novas hipóteses, novamente testadas com experimentos. Este capítulo avalia ainda o desempenho do classificador em diferentes cenários.

O Capítulo 5 sumariza as conclusões obtidas no trabalho e indica como esta pesquisa pode ser continuada, sugerindo trabalhos futuros.

CAPÍTULO 2

Análise de Componentes Principais (PCA)

A *análise de componentes principais* (*principal component analysis*, PCA, em inglês), também conhecida como transformada de Karhunen-Loève, é atualmente um dos principais métodos empregados para análise exploratória de dados. É aplicado em áreas tão diversas quanto neurociência, agricultura e visão computacional [Jol02] pela facilidade com que possibilita extrair informações relevantes de dados complexos.

A ideia central do PCA é reduzir a dimensionalidade de um conjunto de dados formado por um grande número de variáveis interrelacionadas, preservando o máximo possível a variância presente no conjunto. Para isto, os dados são transportados linearmente para um novo espaço, definido por vetores conhecidos como *componentes principais* (*principal components*, PCs, em inglês), de forma que as primeiras variáveis retenham a maior parte da variância presente em todo o conjunto original.

Em reconhecimento de padrões, esta propriedade torna o PCA especialmente interessante para extração de características e redução de dimensionalidade, que de fato é sua principal aplicação nesta área. Entretanto, o PCA possui outras propriedades comumente negligenciadas pela comunidade de reconhecimento de padrões, que certamente possibilitariam usos inovadores da técnica. Uma delas é o *erro quadrátrico médio mínimo* (*least mean square error*, em inglês), que mostra que o espaço definido pelo PCA resulta na melhor aproximação possível dos dados reduzidos em relação aos dados originais [TK09]. Assim, a reconstrução de uma observação tem erro mínimo, fato explorado pelo classificador descrito neste trabalho.

A Seção 2.1 descreve formalmente a análise de componentes principais e conceitos relacionados; a Seção 2.2 analisa os objetivos e o funcionamento da técnica, mostrando uma derivação alternativa; a Seção 2.3 define o erro de reconstrução e discute como escolher o número de componentes principais a serem usados; a Seção 2.4 mostra um algoritmo para encontrar os componentes principais; e a Seção 2.5 faz considerações sobre o histórico da técnica e sobre aplicações em outros campos.

2.1 Definição

Seja $\boldsymbol{X}$ uma *matriz de dados* $m \times n$, onde cada coluna $\boldsymbol{x}_i$ corresponde a uma *observação*, *exemplo*, *instância*, *amostra* ou *padrão*, $i = 1, \ldots, n$, e cada linha $\boldsymbol{d}_j$ corresponde a uma *variável*, *atributo*, *característica* ou *dimensão* d_j, $j = 1, \ldots, m$. O *valor esperado* E é equivalente ao *vetor média* $\boldsymbol{\mu}$, definido de forma que cada um dos seus elementos seja a média de uma linha de $\boldsymbol{X}$

$$E[\boldsymbol{X}] \equiv \boldsymbol{\mu} = [\mu_1, \ldots, \mu_m]^T. \tag{2.1}$$

A variância σ_i^2 de d_i e a covariância σ_{ij} de duas variáveis d_i e d_j são definidas como

$$\sigma_{ij} \equiv \mathrm{Cov}(d_i, d_j) = E[(\boldsymbol{d}_i - \mu_i)(\boldsymbol{d}_j - \mu_j)] = E[\boldsymbol{d}_i \boldsymbol{d}_j] - \mu_i \mu_j, \tag{2.2}$$

com $\sigma_{ij} = \sigma_{ji}$ e $\sigma_{ii} = \sigma_i^2$ quando $i = j$. Para m variáveis existem $m(m-1)/2$ covariâncias, que são geralmente representadas como uma matriz Σ denominada *matriz de covariância*, de dimensão $m \times m$, onde cada elemento na posição (i, j) corresponde à σ_{ij}

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1m} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{m1} & \sigma_{m2} & \cdots & \sigma_m^2 \end{bmatrix}. \tag{2.3}$$

Os termos diagonais correspondem às variâncias, os termos fora da diagonal correspondem às covariâncias e a matriz é simétrica. Alternativamente, Σ pode ser definida como

$$\Sigma \equiv \mathrm{Cov}(\boldsymbol{X}) = E[(\boldsymbol{X} - \boldsymbol{\mu})(\boldsymbol{X} - \boldsymbol{\mu})^T] = E[\boldsymbol{X}\boldsymbol{X}^T] - \boldsymbol{\mu}\boldsymbol{\mu}^T. \tag{2.4}$$

Se duas variáveis são linearmente relacionadas, sua convariâcia será positiva ou negativa dependendo da inclinação da curva. Se duas variáveis são linearmente independentes, sua covariância é zero. Entretanto, o contrário não é verdade: as variáveis podem ser dependentes (de forma não-linear) e ter covariância nula [Alp04].

Seja $\boldsymbol{p}$ um vetor-coluna $m \times 1$. A projeção $\hat{\boldsymbol{X}}$ de $\boldsymbol{X}$ na direção $\boldsymbol{p}$ é dada por

$$\hat{\boldsymbol{X}} = \boldsymbol{p}^T \boldsymbol{X}. \tag{2.5}$$

O objetivo da análise de componentes principais é encontrar um vetor $\boldsymbol{p}$ que maximize a variância de $\hat{\boldsymbol{X}}$. O componente principal $\boldsymbol{p}_1$ é definido de forma que os dados, após serem nele projetados, fiquem mais dipersos no novo espaço, tornando a diferença entre as amostras mais aparente. Para que a solução seja única e para fazer com que a direção seja um fator importante, requer-se que $||p_1|| = 1$. Portanto, $\boldsymbol{p}_1$ pode ser definido como

$$\boldsymbol{p}_1 = \underset{||p||=1}{\arg\max}\, \mathrm{Var}(\boldsymbol{p}^T X). \tag{2.6}$$

Desta forma, procura-se um vetor $\boldsymbol{p}_1$ que maximize $\mathrm{Var}(\hat{\boldsymbol{X}}_1)$, dado que $\boldsymbol{p}_1^T \boldsymbol{p}_1 = 1$. Verifica-se que esta condição é equivalente a

$$\Sigma \boldsymbol{p}_1 = \alpha \boldsymbol{p}_1, \tag{2.7}$$

o que é verdadeiro se $\boldsymbol{p}_1$ é um *autovetor* da matriz de covariância Σ e α é o *autovalor* correspondente [Alp04]. Para que a variância seja máxima, o autovetor com maior

autovalor deve ser escolhido. Assim, o componente principal é o autovetor da matriz de covariância com maior autovalor, $\lambda_1 = \alpha$.

O segundo componente principal, $\boldsymbol{p}_2$, também deve maximizar a variância, ser de comprimento unitário e ainda ser ortogonal a $\boldsymbol{p}_1$. Este último requisito é necessário para que a projeção $\hat{\boldsymbol{X}}_2 = \boldsymbol{p}_2^T \boldsymbol{X}$ não apresente correlação com $\hat{\boldsymbol{X}}_1$. Para o segundo componente principal tem-se novamente que

$$\Sigma \boldsymbol{p}_2 = \alpha \boldsymbol{p}_2, \tag{2.8}$$

o que implica que $\boldsymbol{p}_2$ deve ser o autovetor de Σ com o segundo maior autovalor, $\lambda_2 = \alpha$. Similarmente, é possível mostrar que os componentes principais seguintes são dados pelos autovetores com autovalores em ordem decrescente [Alp04].

Como Σ é simétrica, para dois autovalores distintos, os autovetores são ortogonais, o que garante que as projeções $\hat{\boldsymbol{X}}_1$, $\hat{\boldsymbol{X}}_2$, ... não são correlacionadas. Se Σ é definida positiva ($\boldsymbol{x}^T \Sigma \boldsymbol{x} > 0$, para todo $\boldsymbol{x}$ não-nulo), então todos os seus autovalores são positivos. Se Σ é singular, então seu *rank* (também conhecido como dimensionalidade efetiva ou dimensionalidade intrínseca), é k tal que $k < m$ e λ_i, $i = k+1, \ldots, m$ são nulos (considerando que λ_i está ordenado descrescentemente). Os k autovetores com autovalores não-nulos formam as dimensões do espaço reduzido. O primeiro autovetor, $\boldsymbol{p}_1$, explica a maior parte da variância; o segundo, $\boldsymbol{p}_2$, explica a segunda maior variância; e assim por diante.

A *matriz de projeção* $\boldsymbol{P}$ é definida como a matriz de dimensão $m \times k$ formada pelos autovetores de Σ, onde cada coluna corresponde a um autovetor, dispostos em ordem decrescente, de acordo com os autovalores correspondentes. Após esta projeção, obtém-se um um espaço de k dimensões onde cada dimensão é um autovetor e a variância das amostras em cada dimensão é igual aos autovalores correspondentes de $\boldsymbol{P}$. A projeção $\hat{\boldsymbol{X}}$ de $\boldsymbol{X}$ neste novo espaço é dada por

$$\hat{\boldsymbol{X}} = \boldsymbol{P}^T \boldsymbol{X}, \tag{2.9}$$

considerando que os dados originais estão centrados na origem, ou seja, que $\boldsymbol{X}$ possui média nula (o que pode ser obitdo pela subtração de $\boldsymbol{X}$ por sua média $\boldsymbol{\mu}$).

2.1.1 População e amostra da população

As equações (2.1) e (2.2) são válidas quando $\boldsymbol{X}$ é formada por uma população, ou seja, quando todo o conjunto de observações é conhecido. No entanto, muitas vezes não é possível ou viável determinar a população e $\boldsymbol{X}$ contêm apenas um subconjunto de observações, uma *amostra da população*. Dada esta amostra, é possível estimar a média $\bar{\boldsymbol{\mu}}$ da amostra, tal que

$$\bar{\boldsymbol{\mu}} = \frac{1}{n} \sum_{k=1}^{n} \boldsymbol{x}_k. \tag{2.10}$$

A estimativa de Σ é a matriz de covariância $\boldsymbol{S}$ da amostra da população, composta pelos elementos s_{ij}, tal que

$$s_{ij} = \frac{\sum_{k=1}^{n} (\boldsymbol{x}_i - \bar{\mu}_i)(\boldsymbol{x}_j - \bar{\mu}_j)}{n-1}. \tag{2.11}$$

No restante deste trabalho, os pares $\boldsymbol{\mu}$ e $\bar{\boldsymbol{\mu}}$, e σ_{ij} e s_{ij} serão usados indistintamente, exceto quando explicitamente observado.

2.1.2 Exemplo 1

Para ilustrar o funcionamento do PCA, considere um exemplo simples. O gráfico (a) da Figura 2.1 mostra um conjunto de instâncias geradas de acordo com a função $d_1 = d_2 + \epsilon$, onde ϵ é um ruído adicionado artificialmente. Como definido pela função, o gráfico indica uma clara predominância da distribuição dos dados na direção diagonal e sugere uma correlação positiva entre as variáveis. O gráfico (b) da Figura 2.1 mostra a projeção das instâncias no espaço encontrado pelo PCA. Observe como as instâncias foram rotacionadas de forma que dimensão d'_1 apresente maior variância.

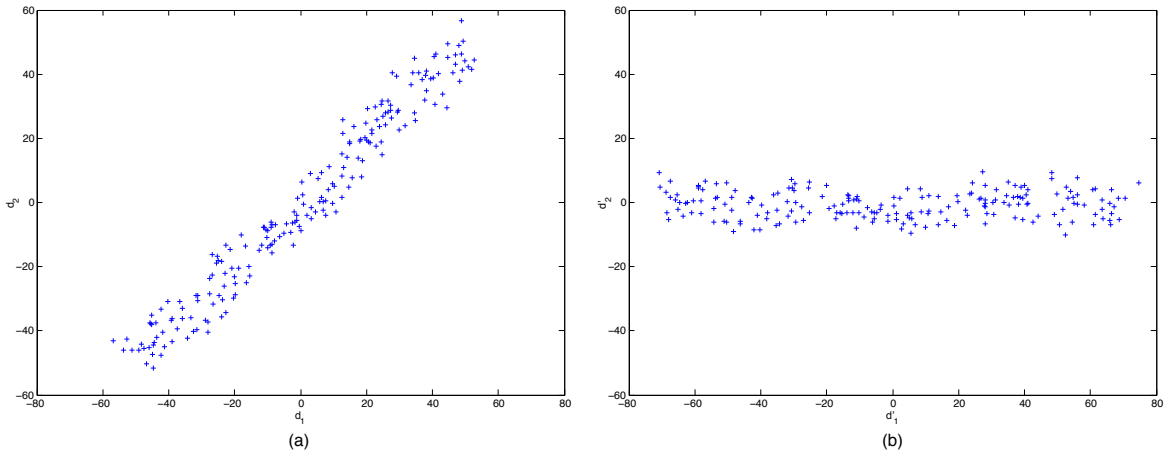

Figura 2.1 Exemplo da aplicação do PCA em um conjunto de dados com duas variáveis. O gráfico (a), à esquerda, mostra os dados originais e o gráfico (b), à direita, mostra os dados após serem projetados no novo espaço. Note que na projeção a dimensão d'_1 apresenta uma variância muito superior à variância da dimensão d'_2. Os autovalores revelam sua magnitude: neste exemplo, a relação λ_1/λ_2 é maior que 90, ou seja, a variância dos dados na dimensão d'_1 é pelo menos 90 vezes maior que a variância na dimensão d'_2; esta razão equivale efetivamente à *relação sinal-ruído* (*signal-to-noise ratio*, SNR, em inglês) da medição.

2.2 Discussão

O objetivo da análise de componentes principais é determinar *a forma mais significativa de re-expressar um conjunto de dados*. A expectativa é que esta nova forma seja capaz de filtrar interferências e revelar a estrutura fundamental dos dados, facilitando sua análise [Shl09].

Na álgebra linear, re-expressar os dados equivale a transportá-los para outro espaço vetorial por meio de uma *transformação linear* [1]. Isso é indicado pela equação (2.9), que usa a *matriz de transformação* $\boldsymbol{P}$ para transformar a matriz de dados original em um novo conjunto. Esta equação também pode ser interpretada como uma *mudança de base* do espaço-vetorial onde os dados se encontram originalmente para um novo espaço, que tem como base os vetores-coluna $\boldsymbol{p}_i$ de $\boldsymbol{P}$.

Para determinar a matriz de transformação, é necessário definir quais propriedades a projeção $\hat{\boldsymbol{X}}$ deve possuir, ou seja, é necessário definir o que *melhor forma de re-expressar os dados* significa exatamente. O PCA assume que a variância é o critério mais importante para descrever os dados e portanto que direções com maior variância possuem maior quantidade de informação. O PCA também assume que, para facilitar a análise dos dados, redundâncias entre as dimensões dos dados originais devem ser eliminadas, de forma que eles possam ser expressos de forma mais concisa e que a quantidade de variáveis possa ser diminuída se houver dependência entre as dimensões; isto equivale a reduzir a covariância entre as dimensões. Desta forma, o objetivo do PCA é *maximizar a variância* entre as amostras e *minimizar a co-variância* entre as dimensões [Shl09].

Para isto, deve-se encontrar uma matriz $\boldsymbol{P}$ tal que a projeção $\hat{\boldsymbol{X}} = \boldsymbol{P}^T\boldsymbol{X}$ (assumindo sem perda de generalidade que $\boldsymbol{X}$ está centralizada) possua uma matriz de covariância $\hat{\boldsymbol{S}}$, onde $\hat{\boldsymbol{S}}$ é uma matriz diagonal com diagonal máxima, ou seja, a covariância entre quaisquer duas dimensões d_i e d_j de $\hat{\boldsymbol{X}}$, $i \neq j$, é nula e variância de $\hat{\boldsymbol{X}}$ é máxima.

Seja $\boldsymbol{C}$ uma matriz de dimensão $m \times m$ na qual a i-ésima coluna corresponde ao autovetor normalizado $\boldsymbol{c}_i$ de uma matriz de covariância $\boldsymbol{S}$, então $\boldsymbol{C}^T\boldsymbol{C} = I$ e

$$\begin{aligned}
\boldsymbol{S} &= \boldsymbol{S}\boldsymbol{C}\boldsymbol{C}^T \\
&= \boldsymbol{S}(\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_m)\boldsymbol{C}^T = (\boldsymbol{S}\boldsymbol{c}_1, \boldsymbol{S}\boldsymbol{c}_2, \ldots, \boldsymbol{S}\boldsymbol{c}_m)\boldsymbol{C}^T \\
&= (\lambda_1\boldsymbol{c}_1, \lambda_2\boldsymbol{c}_2, \ldots, \lambda_m\boldsymbol{c}_m)\boldsymbol{C}^T = \lambda_1\boldsymbol{c}_1\boldsymbol{c}_1^T + \lambda_2\boldsymbol{c}_2\boldsymbol{c}_2^T + \cdots + \lambda_m\boldsymbol{c}_m\boldsymbol{c}_m^T \\
&= \boldsymbol{C}\boldsymbol{D}\boldsymbol{C}^T, \qquad (2.12)
\end{aligned}$$

considerando que $\boldsymbol{D}$ é uma matriz diagonal cujos elementos são os autovetores $\lambda_1, \ldots, \lambda_m$ de $\boldsymbol{S}$. Este processo é conhecido como *decomposição espectral* ou *autodecomposição* de $\boldsymbol{S}$ [Str03]. Como $\boldsymbol{C}$ é ortogonal e $\boldsymbol{C}^T\boldsymbol{C} = I$, é possível multiplicar a equação (2.12) à esquerda por $\boldsymbol{C}^T$ e à direta por $\boldsymbol{C}^T$ para obter

$$\boldsymbol{C}^T\boldsymbol{S}\boldsymbol{C} = \boldsymbol{D}. \qquad (2.13)$$

Sabe-se que se $\hat{\boldsymbol{X}} = \boldsymbol{P}^T\boldsymbol{X}$, então $\mathrm{Cov}(\hat{\boldsymbol{X}}) = \boldsymbol{P}^T\boldsymbol{S}\boldsymbol{P}$, que deve ser igual a uma matriz diagonal [Alp04]. Então, pela equação (2.13), $\boldsymbol{P} = \boldsymbol{C}$. Esta é uma derivação alternativa à mostrada na seção anterior, que formula o PCA como um problema de otimização.

[1] A transformação poderia ser não-linear, mas o PCA original é, por definição, linear. Isto simplifica o cálculo da matriz de transformação pois restringe o cojunto de bases em potencial. Existem variações de PCA baseadas em transformações não lineares, como kernel PCA [SS02] e processos gaussianos de variáveis latentes [RW06], entre outros.

2.2.1 Exemplo 2

O seguinte exemplo mostra como o PCA pode ser aplicado em um problema de classificação. Considere o cojunto de instâncias com duas variáveis mostrado no gráfico (a) da Figura 2.2. Cada amostra do conjunto está associada a uma classe (A ou B) e o objetivo do treinamento é generalizar o padrão exibido por estas instâncias, gerando um classificador capaz de classificar outros padrões fornecidos como pertencentes a uma das classes.

Observando os dados, percebe-se que é possível separar as amostras através de uma reta que passa entre os grupos formados por elementos da mesma classe. Com o PCA, no entanto, é possível separar os dados de forma mais simples, usando apenas uma dimensão, como mostrado no gráfico (b). Note que é possível construir um classificador que considera somente a dimensão d'_1, usando um apenas um limiar pré-estabelecido para determinar a que classe um dado padrão pertence. Em um exemplo simples como este, o ganho de uma dimensão pode não parecer tão vantajoso, mas em casos de instâncias com centenas ou milhares de dimensões—comuns em problemas de processamento de imagens—o PCA pode viabilizar a construção de classificadores eficientes [2].

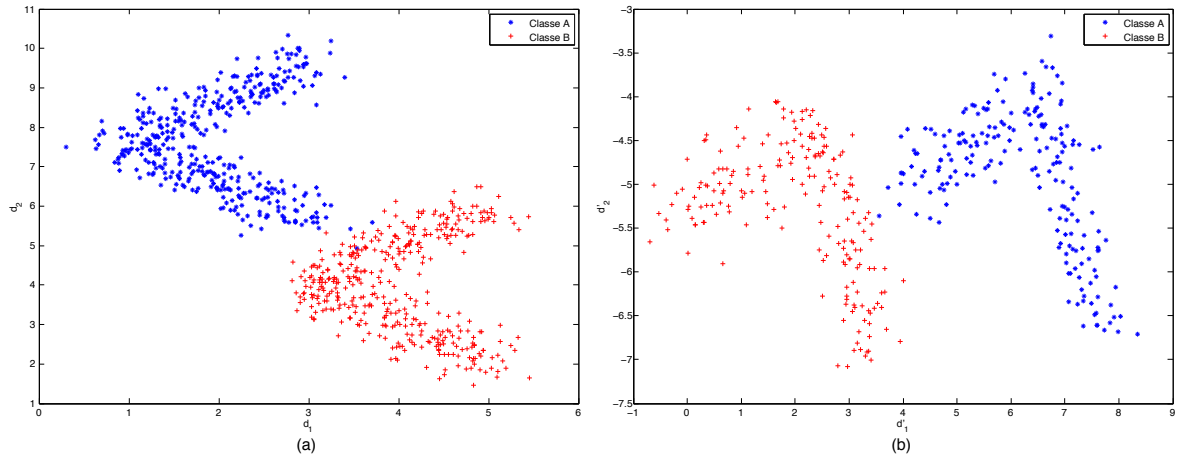

Figura 2.2 Exemplo da aplicação do PCA em um problema de classificação. O gráfico (a), à esquerda, mostra os dados originais: as instâncias (*), em azul, pertecem à classe A e as instâncias (+), em vermelho, pertencem à classe B. Note que para separar corretamente as classes é necessário usar informação das duas dimensões. O gráfico (b), à direita, mostra as instâncias projetadas no espaço gerado pelo PCA. Agora é possível separar as classes satisfatoriamente apenas usando a dimensão d'_1. A relação λ_1/λ_2 entre as variâncias no espaço de projeção é aproximadamente 9.

[2] No entanto, o PCA geralmente não é o melhor candidato para construção de classificadores, já que não leva em consideração as classes do problema, apenas a variação global dos dados. Neste caso específico, a *análise de discriminantes lineares* (*linear discriminant analysis*, LDA, em inglês), por exemplo, poderia encontrar uma projeção ainda melhor para separar as classes.

2.3 Erro de Reconstrução

Considere a amostra $\boldsymbol{x}$ com m variáveis e a matriz de projeção $\boldsymbol{P}$ que contém apenas os primeiros $\boldsymbol{p}_i$ autovetores obtidos pelo PCA, $i = 1, \ldots, k$, tal que $k < m$. A projeção $\hat{\boldsymbol{x}}$ de $\boldsymbol{x}$ em $\boldsymbol{P}$ é definida como

$$\hat{\boldsymbol{x}} = \boldsymbol{P}^T \boldsymbol{x}. \tag{2.14}$$

Este processo é denominado *redução de dimensionalidade* e é uma das aplicações mais comuns da análise de componentes principais. É possível obter uma *reconstrução* $\boldsymbol{x}'$ da amostra original $\boldsymbol{x}$ a partir de sua projeção $\hat{\boldsymbol{x}}$ efetuando a transformação de volta para o espaço original

$$\boldsymbol{x}' = \boldsymbol{P}\hat{\boldsymbol{x}}. \tag{2.15}$$

Como $\boldsymbol{P}$ não possui todos os componentes principais, há perda de informação no processo e $\boldsymbol{x}'$ é apenas uma aproximação de $\boldsymbol{x}$. Desta forma, $\boldsymbol{x}'$ tem um *erro de reconstrução* ϵ associado

$$\epsilon = ||\boldsymbol{x} - \boldsymbol{x}'|| = \sqrt{(x_1 - x_1')^2 + \cdots + (x_m - x_m')^2}, \tag{2.16}$$

considerando que x_i é a i-ésima variável de $\boldsymbol{x}$. Quanto maior for k, ou seja, quanto mais componentes principais houverem em $\boldsymbol{P}$, melhor é a aproximação e menor é o erro de reconstrução.

Considere agora o *erro quadrátrico médio* (*mean square error*, MSE, em inglês), defindo como a média do quadrado dos erros ϵ_i de todas as amostras $\boldsymbol{x}_i$ de uma matriz de dados $\boldsymbol{X}$, dado por

$$\text{MSE} = E[\epsilon^2] = E\left[||\boldsymbol{x} - \boldsymbol{x}'||^2\right]. \tag{2.17}$$

Verifica-se que o MSE corresponde à soma dos autovalores dos $(m - k)$ componentes principais eliminados de $\boldsymbol{P}$ [TK09]

$$\text{MSE} = \sum_{i=k+1}^{m} \lambda_i. \tag{2.18}$$

Portanto, a escolha dos autovetores correspondentes aos k maiores autovalores minimiza o erro quadrático médio. Além disto, prova-se também que este é o erro mínimo comparado com qualquer outra aproximação de $\boldsymbol{x}$ com dimensão k, isto é, dentre todos os espaços k-dimensionais, o espaço encontrado pelo PCA é o que minimiza o MSE [TK09].

Um questionamento comum é como determinar o valor de k, ou seja, como determinar quantos componentes principais serão escolhidos. Se as dimensões são altamente correlacionadas, haverá um pequeno número de autovetores com autovalores não-nulos e k será bem menor que m, o que possibilitará uma redução de dimensionalidade considerável. Este é o caso típico de muitas tarefas de processamento de imagens

e de fala, cujas variáveis próximas (no espaço ou no tempo) são bastante correlacionadas. Por outro lado, se as dimensões não são correlacionadas, k ficará próximo de m e o PCA não oferecerá grandes possibilidades de redução.

O mais comum, no entanto, é que as dimensões sejam moderadamente relacionadas. Nestes casos, deve-se fazer um balanço entre a quantidade de informação a ser preservada e a quantidade de dimensões que as instâncias resultantes devem ter. Uma opção é escolher os k autovalores que respondam por uma quantidade fixa da variância; 90%, por exemplo. Outra opção é selecionar os k primeiros autovalores de forma que a diferença entre λ_k e λ_{k+1} seja considerável. Uma terceira alternativa é plotar os autovalores em um gráfico, conhecido como gráfico de seixos (*scree graph*, em inglês) ou *autoespectro* (*eigen-spectrum*, em inglês), para analisá-los visualmente [Jia09]. A Figura 2.3 mostra um exemplo de autoespectro.

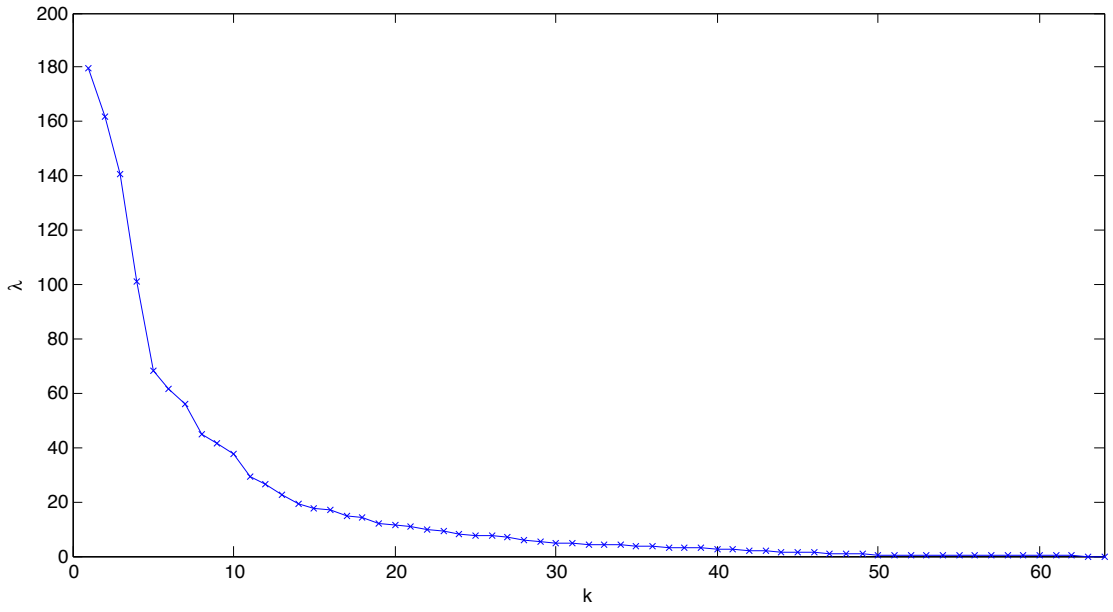

Figura 2.3 Autoespectro gerado a partir da base de dados *Optical Recognition of Handwritten Digits* (optdigits), do *UCI Machine Learning Repository* [FA10]. A base é formada por 3.823 instâncias com 64 variáveis. Os 40 primeiros autovalores respondem por mais de 98% da variância total no espaço de projeção. É possível, portanto, eliminar pelo menos 24 dimensões das instâncias, obtendo um erro menor que 2%. No geral, esta é a forma apresentada pela maioria dos autoespectros: uma curva que decai rapidamente e possui uma cauda bastante alongada. Em alguns problemas de processamento de imagens, que envolvem instâncias com um grande número de dimensões (2.000, por exemplo), esta curva é tão acentuada que o eixo dos autovalores precisa ser disposto em escala logarítmica para permitir que se visualize propriamente como eles se comportam.

2.4 Algoritmo

Na prática, o PCA de um conjunto de dados pode ser calculado da seguinte forma:

1. Organize os dados em uma matriz $\boldsymbol{X}$, de dimensão $m \times n$, onde m é o número de variáveis, dimensões ou medidas e n é o número de amostras, ou seja, cada amostra corresponde a uma coluna da matriz.
2. Calcule a média de cada variável, gerando o vetor m-dimensional $\boldsymbol{\mu}$.
3. Subtraia a média $\boldsymbol{\mu}$ de cada amostra da matriz $\boldsymbol{X}$.
4. Calcule a matriz de covariância $\boldsymbol{S}$ de $\boldsymbol{X}$.
5. Aplique a autodecomposição a $\boldsymbol{S}$, encontrando os conjuntos de autovetores $\boldsymbol{P}$ e de autovalores $\boldsymbol{V}$.
6. Reorganize a matriz $\boldsymbol{P}$ de forma que o autovetor com maior autovalor corresponda à primeira coluna, o autovetor com menor valor corresponda à segunda coluna, e assim por diante.
7. A projeção $\hat{\boldsymbol{X}}$ de $\boldsymbol{X}$ é dada por $\hat{\boldsymbol{X}} = \boldsymbol{P}^T \boldsymbol{X}$; cada coluna da matriz $\hat{\boldsymbol{X}}$ corresponde à amostra original projetada no novo espaço.

2.5 Considerações Históricas e Outras Aplicações

Embora atualmente seja empregada nas mais diversas áreas do conhecimento, a análise de componentes principais teve origem no campo da estatística. As primeiras descrições da técnica hoje conhecida como PCA foram dadas por Pearson [Pea01], em 1901, e por Hotelling [Hot33], em 1933. Embora os dois autores usem abordagens e interpretações distintas, chegam ao mesmo resultado.

O objetivo de Pearson eram encontrar linhas e planos que melhor se ajustassem a um conjunto de pontos em um espaço p-dimensional. A resolução deste problema, expresso por ele como uma otimização geométrica, acabou levando aos PCs, embora este nome não seja citado no seu artigo. Foi Hotelling que expressou o problema da forma como ele é mais conhecido atualmente e introduziu o termo *componentes*. Sua motivação era determinar se havia um "conjunto menor de variáveis fundamentais independentes que determinam os valores" de um conjunto original de p variáveis. Ele escolheu os componentes de forma a maximizar suas contribuições sucessivas ao total das variâncias e chamou-os de *componentes principais*. A análise que encontra estes componentes foi então denominada *método dos componentes principais* [Jol02].

Nos 25 anos seguintes à publicação do artigo de Hotelling, há poucos registros de estudos ou mesmo aplicações do PCA. Entretanto, desde então ocorreu uma explosão de novos usos e trabalhos teóricos sobre a técnica. Esta expansão reflete o crescimento da literatura estatística, mas sua principal causa certamente está ligada ao surgimento e uso em larga escala de computadores eletrônicos. Embora em seu artigo Pearson tenha sido otimista sobre a computação manual do PCA, não é de fato viável fazer os cálculos à mão para um número grande de variáveis, justamente os casos em que a técnica é mais útil.

No começo da década de 1990, o PCA ganhou popularidade no campo de reconhecimento de padrões a partir dos trabalhos de Sirovich e Kirby [SK87], em 1987, e de Turk and Pentland [TP91], em 1991, que aplicaram a técnica para reconhecimento de faces. Turk and Pentland denominaram os componentes principais obtidos a partir de imagens de rostos de *eigenfaces*. Popular até os dias atuais, o trabalho figura entre os mais citados nas áreas de visão computacional e reconhecimento de padrões.

Apesar de sua aparente simplicidade, o PCA atualmente é um campo de pesquisa ativo, sendo uma técnica largamente usada tanto no meio acadêmico como no meio comercial. Isto é claramente ilustrado pela quantidade de trabalhos relacionados ao PCA que foram publicados nesta década: a biblioteca digital do Instituto de Engenheiros Eletricistas e Eletrônicos (*Institute of Electrical and Electronics Engineers*, IEEE, em inglês) registra mais de 8.000 publicações que contêm o termo "análise de componentes principais" apenas entre 2000 e 2010.

CAPÍTULO 3

Classificação por Reconstrução com PCA

Os primeiros registros do uso de PCA em tarefas de visão computacional datam da década de 1980, com os trabalhos de Sirovich e Kirby [SK87] e Turk e Pentland [TP91], ambos para reconhecimento de faces em imagens. A motivação básica é que o PCA captura as características fundamentais de cada face fazendo com que projeções de imagens que contenham a face de um determinado indivíduo ocupem uma região específica no espaço gerado pelos componentes principais (PCs). Assim, após o processo de treinamento, é possível reconhecer um indivíduo analisando a posição em que a projeção de sua face se encontra no espaço obtido pelo PCA. Desde então, esta tem sido a principal forma de aplicação do PCA em visão computacional, sendo a base de funcionamento de vários sistemas de reconhecimento até hoje.

Entretanto, a análise de componentes principais também pode ser usada de outras maneiras. Em 2009, Malagón-Borja e Fuentes [MBF09] propuseram um classificador que se baseia no erro de reconstrução do PCA para detecção de pedestres em imagens. A ideia principal é que o PCA pode comprimir de forma ótima apenas imagens semelhantes às que foram usadas para computá-lo e que imagens de outro tipo não serão tão bem comprimidas. Então, o PCA é calculado separadamente para um conjunto de imagens que contêm apenas pedestres (conjunto positivo) e para um conjunto de imagens que não contêm pedestres (conjunto negativo). Para classificar uma nova imagem, o sistema compara a reconstrução feita a partir do cojunto positivo com a reconstrução do conjunto negativo: imagens que contêm pedestres devem apresentar melhor reconstrução com o cojunto positivo, enquanto imagens que não contêm pedestres devem apresentar pior reconstrução com o conjunto positivo.

Para melhorar a performance do conjunto, o sistema também usa imagens de borda juntamente com as imagens em escala de cinza. Elas diminuem a variabilidade desnecessária e destacam a silhueta dos pedestres, que serve como informação adicional para o classificador. No sistema original, apenas o filtro Sobel é usado para computar as imagens de borda. Os autores, no entanto, não justificam a escolha deste filtro específico nem por que as imagens em escala de cinza ainda precisam ser usadas em conjunto. Também não fica claro que função as imagens do conjunto negativo desempenham no sistema. No conjunto positivo há claramente um padrão a ser extraído pelo PCA: o de pedestres. Já o conjunto negativo é formado for imagens diversas, que podem conter qualquer objeto—exceto pedestres—então que padrão pode ser extraído deste conjunto? Ele é realmente necessário para a classificação?

Neste capítulo, investigamos estas questões e propomos uma variação do critério de classificação para aumentar a acurácia do sistema. A Seção 3.1 descreve o sistema

de detecção original e analisa seu funcionamento; a Seção 3.2 avalia a contribuição do conjunto positivo e do conjunto negativo para o desempenho do classificador; a Seção 3.3 avalia a contribuição do conjunto de imagens em escala de cinza e de imagens de borda para a classificação; e a Seção 3.4 descreve uma variação do critério de classificação original, o *erro de reconstrução ponderado*, proposto para melhorar o desempenho do classificador.

3.1 Sistema de Detecção

O objetivo do sistema é determinar se uma dada imagem desconhecida contém ou não um pedestre. Para isto, ele usa um classificador previamente treinado com imagens em escala de cinza e imagens de borda, de pedestres e não-pedestres, que gera reconstruções com PCA e compara estas reconstruções para determinar se a imagem em questão pertence ou não à classe de pedestres.

Esta seção descreve o classificador proposto por Malagón-Borja e Fuentes [MBF09]. Inicialmente a notação usada é descrita, bem como a geração de reconstruções de imagens com PCA. Em seguida discute-se as razões para o uso de imagens de borda em conjunto com imagens de escala de cinza. Por fim, descreve-se como o classificador pode usar reconstruções para decidir a que classe uma dada imagem pertence.

3.1.1 Reconstrução de Imagens com PCA

Considere um conjunto de n imagens em escala de cinza, cada uma de tamanho $w \times h$ pixels, cada pixel com valor entre 0 e 254. Toda imagem $\boldsymbol{I}_i$ pode ser representada como um vetor-coluna $\boldsymbol{v}_i$ de tamanho wh pelo rearranjo de seus pixels. A média $\boldsymbol{\mu}$ e a matriz de covariância $\boldsymbol{S}$ do conjunto podem ser calculadas de acordo com as definições da Seção 2.1.

Seja $\boldsymbol{P}$ a matriz cujas colunas correspondem aos k autovetores de maior autovalor de $\boldsymbol{S}$. A *projeção* $\hat{\boldsymbol{u}}$ de uma imagem $\boldsymbol{u}$ no espaço definido por estes PCs é dada por

$$\hat{\boldsymbol{u}} = \boldsymbol{P}^T(\boldsymbol{u} - \boldsymbol{\mu}). \tag{3.1}$$

A partir desta projeção, pode-se tentar reconstruir a imagem original. A reconstrução $\boldsymbol{u}'$ da imagem projetada $\hat{\boldsymbol{u}}$ é definida como a transformação de volta para o espaço original

$$\boldsymbol{u}' = \boldsymbol{P}\hat{\boldsymbol{u}} + \boldsymbol{\mu} = \boldsymbol{P}\boldsymbol{P}^T(\boldsymbol{u} - \boldsymbol{\mu}) + \boldsymbol{\mu}. \tag{3.2}$$

Como normalmente $k \ll wh$, ou seja, o número de PCs usados para a projeção é bem menor que a quantidade total de PCs obtidos, há perda de informação no processo e a reconstrução é apenas uma aproximação da imagem original. Desta forma, cada reconstrução tem um erro associado, o *erro de reconstrução*, dado por

$$d = |\boldsymbol{u} - \boldsymbol{u}'| = \sqrt{\sum(\boldsymbol{u}_i - \boldsymbol{u}'_i)}, \tag{3.3}$$

ou seja, o erro de reconstrução é a norma da diferença entre a imagem original e sua reconstrução, considerando que $\boldsymbol{u}_i$ representa cada pixel da imagem, $1 \leq i \leq wh$.

3.1.2 Imagens de Borda

Pedestres podem usar vários tipos de roupa, de diversas cores, estampas e texturas. Seu tom de pele pode variar consideravelmente, sendo afetado inclusive pelas condições de iluminação na qual a imagem foi registrada. Por estas razões, não é aconselhável usar características baseadas em cores ou texturas para detectar pedestres [MBF09].

Para filtrar estas informações, o classificador usa imagens de borda de forma a obter a silhueta típica de um pedestre, diminuindo a variabilidade desnecessária entre as imagens. A Figura 3.1 mostra algumas imagens em escala de cinza, acompanhadas da imagem de bordas correspondente. Há uma clara diminuição da variabilidade entre as imagens em escala de cinza com relação às imagens de borda.

Figura 3.1 Imagens em escala de cinza de pedestres (acima) acompanhadas da imagem de borda correspondente (abaixo). Observe que apesar da cor e do fundo das imagens originais serem bastante distintos, as imagens de borda apresentam pouca diferença de uma para outra. As imagens de borda foram obtidas pela soma das magnitudes dos filtros Sobel horizontal e vertical (equação 4.1); um comportamento semelhante é observado com os outros filtros testados.

3.1.3 Classificação por Reconstrução com PCA

Por definição, o PCA procura pelo conjunto de componentes principais (PCs) que melhor descreve a distribuição dos dados que estão sendo analisados. Estes PCs devem preservar melhor a informação de imagens que foram usadas para computar o PCA ou de imagens semelhantes. Portanto, um conjunto de PCs gerados a partir de imagens que contêm apenas pedestres deve reconstruir melhor imagens de outros pedestres que qualquer outro tipo de imagem. Da mesma forma, se têm-se um conjunto de PCs

obtidos a partir de imagens que contêm qualquer coisa exceto pedestres, a reconstrução de imagens de pedestres não deve ser tão boa [MBF09]. Isto pode ser observado na Figura 3.2, tanto para imagens em escala de cinza como para imagens de borda.

Assim, o erro de reconstrução de uma imagem que contém um pedestre deve ser menor para o conjunto de PCs positivos e maior para o conjunto de PCs negativos. O contrário acontece com uma imagem que não contém um pedestre. Partindo desta observação, é possível criar um classificador baseado em reconstrução de imagens com PCA, que usa erros de reconstrução como critério de classificação. Na fase de treinamento, o PCA deve ser executado separadamente para quatro conjunto de imagens, que resulta no seguinte conjunto de PCs:

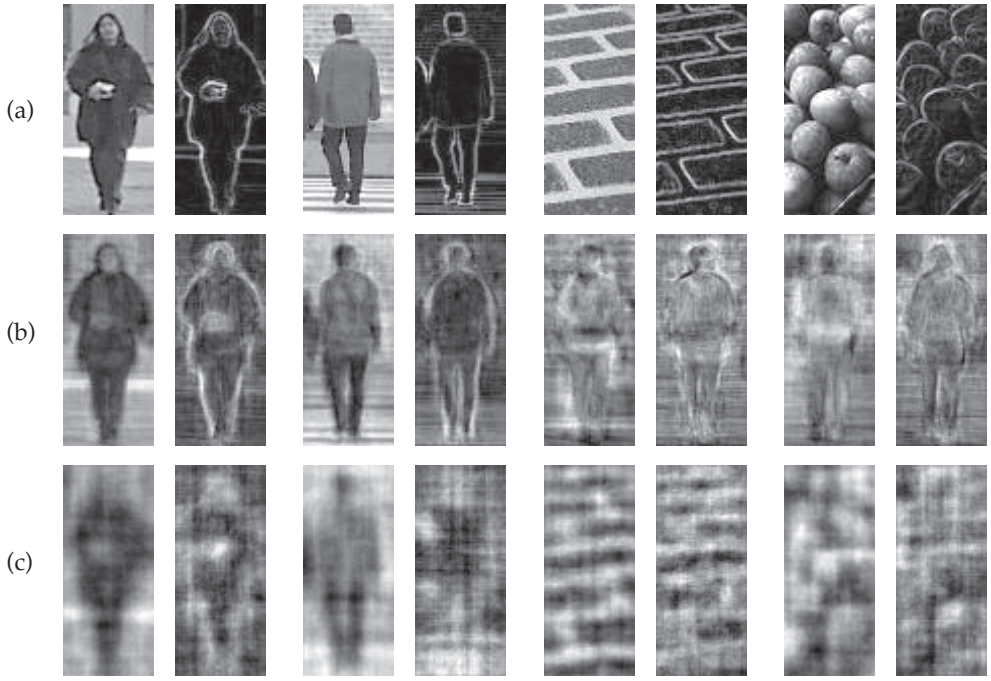

Figura 3.2 Quatro imagens usadas para treinamento do classificador, duas de pedestres e duas de não-pedestres, acompanhadas de suas imagens de borda e reconstruções. A primeira linha (a) mostra as imagens originais em escala de cinza com suas respectivas imagens de borda, obtidas com o filtro Sobel. A segunda linha (b) mostra as imagens reconstruídas usando 100 PCs positivos (P_{gp} para imagens em escala de cinza e P_{ep} para imagens de borda). A terceira linha (c) mostra as imagens reconstruídas usando 100 PCs negativos (P_{gn} para imagens em escala de cinza e P_{en} para imagens de borda). Observe que as imagens de pedestres são melhor reconstruídas com os componentes principais do conjunto positivo.

- Os componentes principais $\boldsymbol{P}_{gp}$ e a média $\boldsymbol{\mu}_{gp}$ do conjunto de imagens em escala de cinza de pedestres;
- Os componentes principais $\boldsymbol{P}_{ep}$ e a média $\boldsymbol{\mu}_{ep}$ do conjunto de imagens de borda de pedestres;
- Os componentes principais $\boldsymbol{P}_{gn}$ e a média $\boldsymbol{\mu}_{gn}$ do conjunto de imagens em escala de cinza de não-pedestres;
- Os componentes principais $\boldsymbol{P}_{en}$ e a média $\boldsymbol{\mu}_{en}$ do conjunto de imagens de borda de não-pedestres.

Na fase de operação, para classificar uma nova imagem, o primeiro passo é obter sua imagem de boda e gerar quatro reconstruções, uma para cada conjunto de PCs. Então, os erros de reconstrução devem ser calculados e combinados para produzir o erro de reconstrução total, que funciona como o escore de classificação. Se o erro total é maior ou igual a zero, a imagem é classificada como *pedestre*; caso contrário, a imagem é classificada como *não-pedestre*. O procedimento passo-a-passo é descrito abaixo.

1. Obtenha a imagem de borda $\boldsymbol{e}$ a partir da imagem em escala de cinza $\boldsymbol{g}$ usando o operador Sobel;

2. Gere quatro reconstruções:

   (a) $\boldsymbol{u}'_{gp} = \boldsymbol{P}_{gp}\boldsymbol{P}_{gp}^T(\boldsymbol{g} - \boldsymbol{u}) + \boldsymbol{\mu}_{gp}$
   (b) $\boldsymbol{u}'_{ep} = \boldsymbol{P}_{ep}\boldsymbol{P}_{ep}^T(\boldsymbol{e} - \boldsymbol{u}) + \boldsymbol{\mu}_{ep}$
   (c) $\boldsymbol{u}'_{gn} = \boldsymbol{P}_{gn}\boldsymbol{P}_{gn}^T(\boldsymbol{g} - \boldsymbol{u}) + \boldsymbol{\mu}_{gn}$
   (d) $\boldsymbol{u}'_{en} = \boldsymbol{P}_{en}\boldsymbol{P}_{en}^T(\boldsymbol{e} - \boldsymbol{u}) + \boldsymbol{\mu}_{en}$

3. Calcule os erros de reconstrução correspondentes:

   (a) $d_{gp} = |\boldsymbol{u}'_{gp} - \boldsymbol{g}|$
   (b) $d_{ep} = |\boldsymbol{u}'_{ep} - \boldsymbol{e}|$
   (c) $d_{gn} = |\boldsymbol{u}'_{gp} - \boldsymbol{g}|$
   (d) $d_{en} = |\boldsymbol{u}'_{en} - \boldsymbol{e}|$

4. Compute o *erro de reconstrução total*, dado por

$$d_t = d_{gn} + d_{en} - d_{gp} - d_{ep} \tag{3.4}$$

5. Classifique a imagem de acordo com:

$$\text{classe}(g) = \begin{cases} \text{Pedestre}, & d_t \geq 0 \\ \text{Não-pedestre}, & d_t < 0 \end{cases} \tag{3.5}$$

3.2 Papel das Imagens de Não-Pedestres

Quando analisamos o funcionamento do classificador, é fácil entender o papel dos PCs gerados pelas amostras positivas. Como as imagens contêm um padrão recorrente do objeto-alvo (pedestres, neste caso), o PCA é capaz de capturar este padrão e reconstruir imagens similares satisfatoriamente, gerando erros de reconstrução menores para outras imagens de pedestres. Por outro lado, o papel dos PCs negativos não é tão claro. Como as imagens negativas contêm apenas objetos aleatórios e não representam um conceito específico, a questão é: que padrão pode ser capturado a partir destas imagens? Inicialmente, imaginou-se que os erros das reconstruções obtidas a partir dos PCs negativos teria uma importância mínima e portanto sua eliminação não causaria um grande impacto na acurácia do classificador. Entretanto, os experimentos contrariam esta hipótese: os erros negativos têm uma grande relevância na classificação, maior inclusive que a relevância dos erros positivos na maior parte dos casos.

Os resultados sugerem que a importância dos PCs negativos não está no padrão que o PCA captura, mas sim no padrão que ele *não* captura. Desta forma, os erros de reconstrução negativos tendem a ser grandes para imagens que contêm o objeto-alvo enquanto os erros de reconstrução positivos tendem a ser pequenos para as mesmas imagens, contribuindo para um erro total maior, o que classifica a imagem corretamente. Para imagens que não contêm o objeto-alvo, acontece exatamente o oposto. Assim, ambos os componentes do erro total (originados de PCs positivos e de PCs negativos) trabalham juntos para a classificação. Por esta razão, não é viável construir um classificador que use apenas erros de reconstrução positivos. Os experimentos que levaram a esta conclusão e os resultados obtidos são descritos na Seção 4.2.

3.3 Papel das Imagens de Borda

O classificador original usa imagens de borda em conjunto com as imagens em escala de cinza originais de forma a aumentar a quantidade de informação disponível para a classificação, o que facilitaria a separação entre as classes de pedestres e não-pedestres. Entretanto, como a imagem de bordas é gerada a partir da imagem em escala de cinza, é razoável imaginar que haja uma forte correlação entre elas, fazendo com que estes dois conjuntos compartilhem uma grande quantidade de informação. Na verdade, as imagens em escala de cinza devem possuir uma maior relevância para a classificação, já que eliminam texturas e variações desnecessárias.

Os experimentos, de fato, confirmam estas hipóteses: é possível eliminar imagens de borda ou imagens em escala de cinza sem afetar significativamente o desempenho do classificador, sendo que a eliminação de imagens em escala de cinza causa um impacto ainda menor. Isto indica que as imagens de borda são mais relevantes para a classificação, embora em uma magnitude muito menor do que a cogitada inicialmente. Os experimentos que levaram a esta conclusão e os resultados obtidos são descritos na Seção 4.3.

3.4 Erro de Reconstrução Ponderado

No erro de reconstrução total (equação 3.4), cada componente contribui igualmente para o resultado final. Entretanto, os experimentos e a discussão nas duas seções anteriores mostram que esta relação não é observada na prática: dos quatro erros de reconstrução usados pelo classificador, alguns são mais relevantes que outros e portanto precisam ser ajustados de acordo. Desta forma, propomos o *erro de reconstrução total ponderado*, dado por

$$d_t^w = w_{gn}d_{gn} + w_{en}d_{en} - w_{gp}d_{gp} - w_{ep}d_{ep} \tag{3.6}$$

onde w_{gn}, w_{en}, w_{gp}, w_{ep} são pesos que ajustam a importância de cada erro para a classificação (erros mais importantes devem ter maior peso). Eles podem ser definidos manualmente ou encontrados por alguma técnica de otimização com relação ao conjunto de treinamento.

É possível ainda generalizar esta noção e definir erros parciais, ponderados ou não:

Erro de Reconstrução Parcial Positivo

$$d_p = d_{gp} + d_{ep} \tag{3.7}$$

Erro de Reconstrução Parcial Negativo

$$d_n = d_{gn} + d_{en} \tag{3.8}$$

Erro de Reconstrução Parcial de Escala de Cinza

$$d_g = d_{gn} - d_{gp} \tag{3.9}$$

Erro de Reconstrução Parcial de Bordas

$$d_e = d_{en} - d_{ep} \tag{3.10}$$

Erro de Reconstrução Parcial Positivo Ponderado

$$d_p^w = w_{gp}d_{gp} + w_{ep}d_{ep} \tag{3.11}$$

Erro de Reconstrução Parcial Negativo Ponderado

$$d_n^w = w_{gn}d_{gn} + w_{en}d_{en} \tag{3.12}$$

Erro de Reconstrução Parcial de Escala de Cinza Ponderado

$$d_g^w = w_{gn}d_{gn} - w_{gp}d_{gp} \tag{3.13}$$

Erro de Reconstrução Parcial de Bordas Ponderado

$$d_e^w = w_{en}d_{en} - w_{ep}d_{ep} \tag{3.14}$$

É importante notar também que o classificador pode ser modificado para ganhar em flexibilidade se introduzirmos um limiar na equação (3.5). Assim, o escore de classificação seria comparado a este limiar, ao invés de mantê-lo fixo sempre em zero. Desta forma, os usuários podem ajustar a especificidade do sistema, dependendo de sua aplicação em particular.

CAPÍTULO 4

Experimentos

Para entender melhor o funcionamento do classificador e testar as hipóteses levantadas no capítulo anterior, alguns experimentos foram realizados. Os testes foram feitos no contexto da tarefa de *detecção em imagens*, isto é, dada uma imagem não-rotulada, o classificador deve determinar se ela apresenta ou não o objeto-alvo (pedestres, neste caso específico).

Para treinar o classificador, é necessário um conjunto de imagens de *pedestres* e um conjunto de imagens de *não-pedestres* (imagens que podem conter qualquer objeto, exceto pedestres). A fase de treinamento consiste em computar as imagens de borda e aplicar o PCA aos quatro conjuntos separadamente, o que produz quatro conjuntos de componentes principais (PCs), como descrito na seção 3.1. Eles são usados para gerar as respectivas reconstruções na fase de operação do classificador.

As imagens de pedestres foram obtidas a partir da base MIT CBCL (*MIT CBCL pedestrian database*, em inglês) [Cen00], que contém 924 imagens coloridas de pedestres em posição frontal e traseira, sob diferentes condições de iluminação. Estas imagens foram convertidas para escala de cinza e recortadas para eliminar parte do fundo, de modo a remover parte da variação irrelevante entre as imagens. As amostras finais ficaram com 45×105 pixels.

As imagens de não-pedestres foram extraídas de 120 fotografias variadas que não continham pedestres. Cada uma delas foi cortada em fatias de 45×105 pixels e 5.000 fatias foram aleatoriamente selecionadas para compor o conjunto negativo. Este conjunto é mais que quatro vezes maior que o conjunto positivo; isto está relacionado com a capacidade de representação de cada um. O problema de detecção de objetos é naturalmente assimétrico, já que os exemplos positivos representam um tipo de objeto específico enquanto os exemplos negativos representam o "resto do mundo." É por esta razão que precisamos de mais exemplos negativos que positivos [Jia09].

Os conjuntos foram divididos da seguinte forma: 75% das imagens de pedestres (693 amostras) foram selecionadas para treinamento e o restante (231 amostras) foi usado para teste; 80% das imagens de não-pedestres (4.000 amostras) foram selecionadas para treinamento e o restante (1.000 amostras) foi usado para teste. Este esquema em particular foi escolhido de maneira empírica, apresentando os melhor compromisso entre generalização e *overfitting* dentre as distribuições testadas.

Verifica-se que a qualidade de uma imagem de borda varia consideravelmente de acordo com o método escolhido para gerá-la. Para avaliar como este fator afeta o desempenho do sistema, testamos o classificador com imagens de borda geradas por quatro algoritmos distintos. O classificador e todos os procedimentos auxiliares foram

implementados e executados no ambiente Matlab, da Mathworks. O código resultante está disponível para download em http://github.com/lailsonbm/pca_reconstruction/.

Neste capítulo, descrevemos os experimentos realizados e analisamos os resultados obtidos. A Seção 4.1 examina os autoespectros de cada conjunto de treinamento para entender melhor o comportamento dos dados; a Seção 4.2 avalia a importância das imagens de não-pedestres e mostra por que este conjunto é necessário para a classificação; a Seção 4.3 compara a importância das imagens em escala de cinza e das imagens de borda no desempenho do classificador; a Seção 4.4 discute como a ponderação dos erros pode melhorar a acurácia de classificação e como encontrar os pesos usando algoritmos genéticos; e a Seção 4.5 analisa como diferentes algoritmos de borda afetam o classificador.

4.1 Autoespectros

Antes de iniciar os experimentos, é interessante analisar os autoespectros (Seção 2.3) gerados no treinamento. Isto permite entender melhor como os conjuntos se comportam e nos ajuda a determinar a quantidade de componentes principais que devem ser usados nos experimentos. O autoespectro de cada conjunto é mostrado na Figura 4.1. Todos apresentam curvas típicas de conjuntos de imagens: os primeiros PCs descrevem uma alta variância, que diminui rapidamente conforme o número de PCs aumenta.

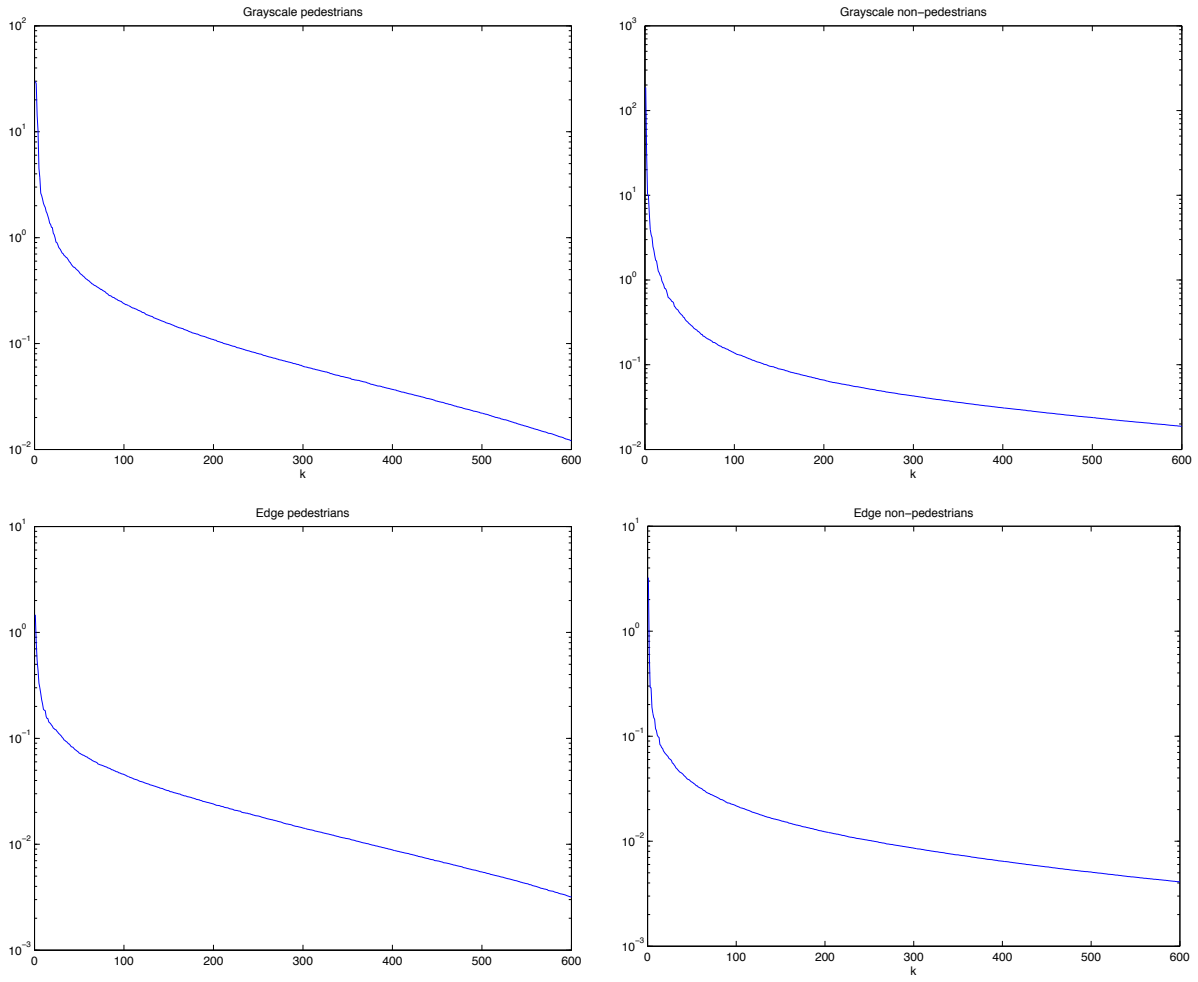

Figura 4.1 Autoespectros dos 600 primeiros componentes principais do conjunto de imagens em escala de cinza e de bordas de pedestres (gráficos superior e inferior esquerdo, respectivamente) e do conjunto de imagens em escala de cinza e de bordas de não-pedestres (gráficos superior e inferior direito, respectivamente). O eixo vertical representa os autovalores e o eixo horizontal o número de PCs (k). Os autovalores decaem muito rapidamente (o eixo y está em escala logarítmica), indicando que a maior parte da informação do conjunto está concentrada nos primeiros PCs. Observe que as imagens de não-pedestres possuem uma curva de queda ligeiramente mais suave, especialmente para os maiores valores de k; isto é reflexo da maior variabilidade entre as imagens do conjunto. Note também que os conjuntos de imagens de borda concentram ainda menos informações nos maiores PCs (o menor valor da escala vertical dos gráficos inferiores é 10^{-3}), resultado da eliminação de grande parte da variabilidade desnecessária deste conjunto. As imagens de borda foram obtidas com o filtro Sobel; um comportamento semelhante é observado para os demais algoritmos testados.

4.2 Imagens de Pedestres *versus* Imagens de Não-Pedestres

Começamos investigando o impacto na classificação dos erros de reconstrução obtidos a partir de PCs negativos e de PCs positivos. Como já discutido, inicialmente questionamos o papel dos PCs negativos, já que eles são computados de imagens completamente sem relação e não representam um objeto em particular. Entretanto, quando classificamos as amostras de teste usando apenas o erro de reconstrução parcial positivo (equação 3.7) ou o erro de reconstrução parcial negativo (equação 3.8)—mas ainda usando imagens em escala de cinza e imagens de borda—fica claro que nenhum deles é capaz de classificar bem as imagens isoladamente. Porém, quando os erros são combinados de acordo com o erro de reconstrução total (equação 3.4), tem-se uma boa acurácia. As curvas ROC (*receiver operating characteristic*, em inglês) [Faw06] dos três classificadores são mostradas na Figura 4.2 para 100 PCs. Obtivemos um padrão similiar para os outros valores de k testados.

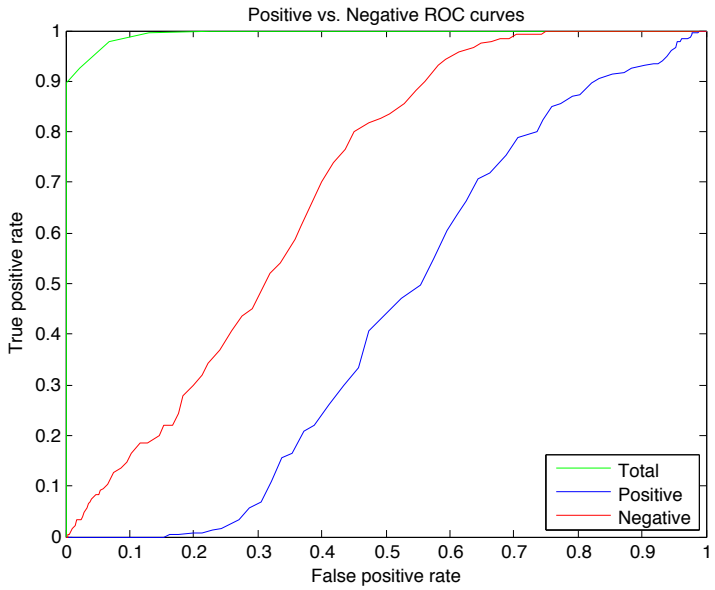

Figura 4.2 Comparação das curvas ROC positivas e negativas. Da esquerda para a direita: a primeira curva (verde) se refere ao classificador que usa o erro de reconstrução total (eq. 3.4) e tem AUC de 0,9936; a segunda curva (vermelha) se refere ao classificador que usa apenas os PCs negativos (erro de reconstrução parcial negativo, eq. 3.8) e tem AUC de 0,6900; e a terceira curva (azul) se refere ao classificador que usa apenas os PCs positivos (erro de reconstrução parcial positivo, eq. 3.7) e tem AUC de 0,4476. Nos três classificadores, apenas os 100 primeiros componentes principais são usados, isto é, $k = 100$. Ambos os classificadores positivos e negativos obtêm um resultado ruim, longe do desempenho do classificador completo. O mais surpreendente é que o classificador negativo acaba gerando um resultado melhor que o classificador positivo. As imagens de borda foram obtidas com o filtro Sobel; um comportamento semelhante é observado para os demais algoritmos testados.

Para entender por que isto acontece, é interessante analisar como os erros de reconstrução estão distribuídos. Na Figura 4.3, plotamos o erro de reconstrução de cada amostra de teste individualmente nas três situações descritas acima. É possível observar que as amostras estão misturadas nos dois primeiros gráficos, indicando que os respectivos erros não possuem poder discriminatório suficiente para separar as amostras. No entanto, quando os erros são combinados, a distribuição muda completamente e podemos perceber que amostras de diferentes classes ficam de fato agrupadas. Isto possibilita a alta taxa de classificação que obtivemos. Portanto, os erros de reconstrução de ambos os PCs positivos e negativos contribuem de forma significativa para a classificação e nenhum deles pode ser removido.

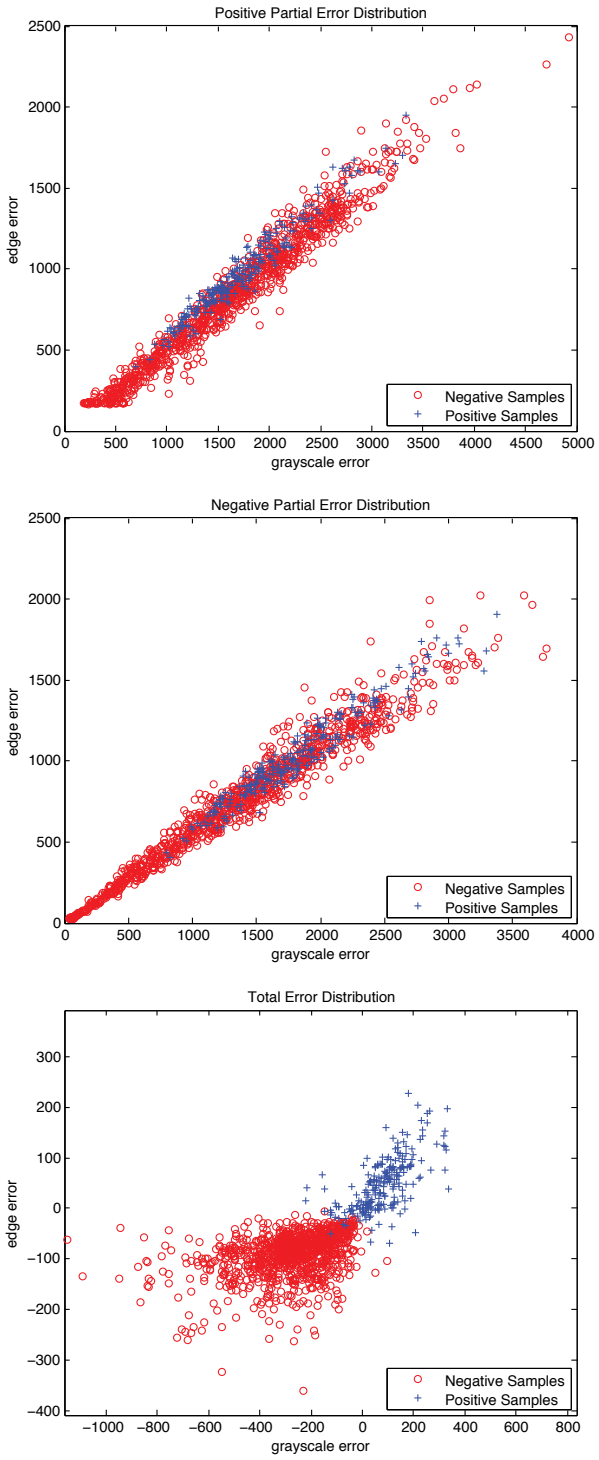

Figura 4.3 Comparação da distribuição dos erros de reconstrução positivos e negativos das amostras do conjunto de teste para $k = 100$. O primeiro gráfico mostra a distribuição dos erros de reconstrução parcial positivos; o segundo mostra a distribuição dos erros de reconstrução parcial negativos; e o terceiro mostra a distribuição dos erros de reconstrução total. Observe que as amostras estão completamente misturadas nos dois primeiros gráficos: é praticamente impossível separá-las usando apenas estes erros. Por outro lado, o terceiro gráfico mostra dois grupos bem definidos, o que viabiliza uma boa classificação.

4.3 Imagens em Escala de Cinza *versus* Imagens de Borda

Por outro lado, obtemos curvas ROC muito semelhantes quando o classificador considera separadamente os erros de reconstrução obtidos a partir de PCs de imagens em escala de cinza e de PCs de imagens de borda (agora sempre usando os erros dos PCs positivos e dos PCs negativos juntos). Isto é mostrado na Figura 4.4, que exibe três curvas ROC: uma do classificador que usa apenas erros de imagens em escala de cinza, uma do classificador que usa apenas erros de imagens de borda e uma do classificador que considera ambos (isto é, o erro de reconstrução total). Note que as curvas ROC do classificador de bordas e do classificador do erro total são quase indistinguíveis. Na Tabela 4.1, comparamos os classificadores usando sua area sob a curva ROC (*area under the curve*, AUC, em inglês), uma métrica comum para medir o desempenho de classificadores: no geral, quanto maior a AUC, melhor é o performance do classificador [Faw06].

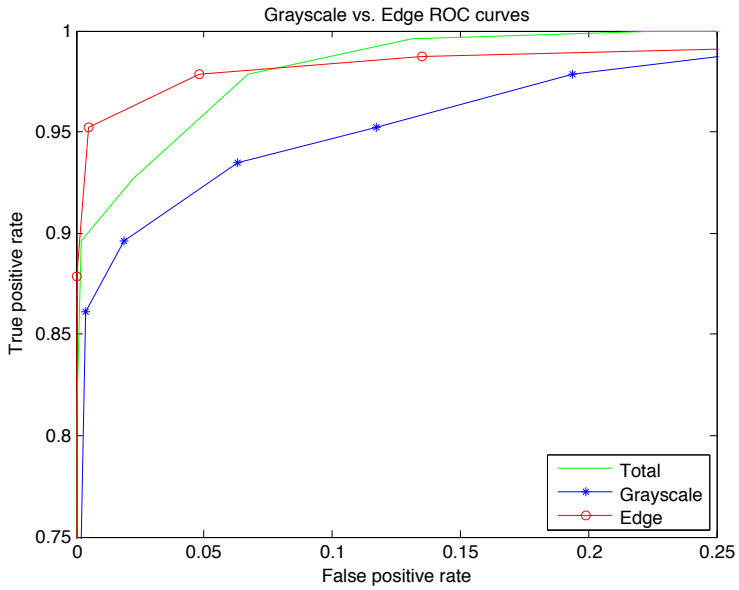

Figura 4.4 Comparação das curvas ROC do classificador que usa imagens em escala de cinza e do que usa imagens de bordas. Da esquerda para a direita: a curva verde se refere ao classificador que usa o erro de reconstrução total (eq. 3.4) e tem AUC de 0,9936; a curva vermelha se refere ao classificador que usa o erro parcial de bordas (eq. 3.10) e tem AUC de 0,9943; e a curva azul se refere ao classificador que usa o erro parcial de escala de cinza (eq. 3.9) e tem AUC de 0,9846. Nos três classificadores, $k = 100$. Observe que a escala dos eixos é diferente da usada na Figura 4.2; o canto superior esquerdo foi aproximado para mostrar melhor a pequena diferença entre as curvas. As imagens de borda foram obtidas com o filtro Sobel.

Novamente plotamos as distribuições dos erros de reconstrução, agora separando os erros de PCs em escala de cinza dos erros de PCs de borda. Estas distribuições são mostradas na Figura 4.5, que também mostra um gráfico do erro de reconstrução total. Desta vez, observamos um cenário distinto. Os três gráficos mostram uma fronteira

clara separando amostras de diferentes classes. Isto sugere que há redundância entre erros parciais de escala de cinza e de borda, o que é confirmado pelos resultados da Tabela 4.1.

Desta forma, é possível compor um classificador que usa apenas erros de imagens em escala de cinza ou de imagens de borda e acelerar a classificação significativamente (já que duas reconstruções não terão mais que ser computadas) apenas com um pequeno impacto na acurácia de classificação. Os tempos de detecção são listados na Tabela 4.2 para cada classificador e número de PCs. No geral, os experimentos mostraram que o classificador que usa o erro parcial de borda diminui a acurácia em menos de 1% quando comparado ao classificador que usa o erro total. Na verdade, o classificador baseado apenas nos erros de borda em alguns casos apresentou um desempenho melhor até mesmo que o classificador completo em nossos experimentos (quando $k \geq 100$), como também mostrado na Tabela 4.1.

k	CINZA	BORDAS	TOTAL
400	0,9598	0,9874	0,9750
300	0,9682	0,9882	0,9800
200	0,9766	0,9897	0,9876
100	0,9846	0,9943	0,9936
50	0,9903	0,9946	0,9969
25	0,9929	0,9932	0,9983

Tabela 4.1 AUCs dos classificadores para diferentes quantidades de PCs. O primeiro classificador usa o erro de reconstrução parcial de escala de cinza; o segundo usa o erro de reconstrução parcial de bordas; e o terceiro usa o erro de reconstrução total. Observe que para $k \geq 100$, o classificador de bordas apresenta um desempenho melhor que classificador total, apesar deste último possuir mais informações. As imagens de borda foram obtidas com o filtro Sobel.

k	CINZA	BORDAS	TOTAL
400	11,90	13,51	25,48
300	9,09	10,72	19,79
200	6,23	7,82	14,04
100	2,92	4,51	7,55
50	1,22	2,95	4,12
25	0,62	2,24	3,10

Tabela 4.2 Tempo médio de detecção em milissegundos para uma imagem, para cada classificador e quantidade de PCs. As detecções foram executadas nas mesmas imagens, sob as mesmas condições, usando o filtro Sobel para obter as imagens de borda. A diferença entre os tempos de detecção dos dois classificadores parciais corresponde à computação das bordas.

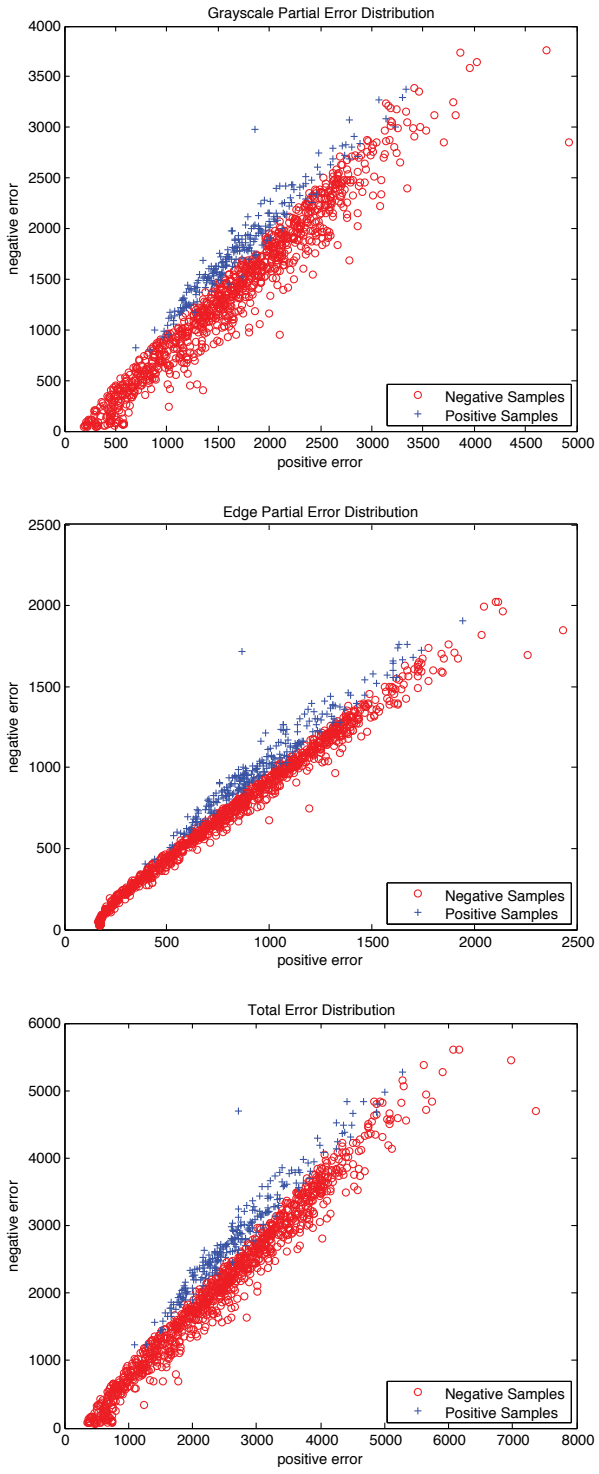

Figura 4.5 Comparação da distribuição dos erros de reconstrução de escala de cinza e de bordas das amostras do conjunto de testes para $k = 100$. O primeiro gráfico mostra os erros de reconstrução parcial de escala de cinza; segundo mostra a distribuição dos erros de reconstrução parcial de bordas; e o terceiro mostra a distribuição dos erros de reconstrução total. Observe que os três gráficos apresentam uma clara fronteira de separação entre as amostras positivas e negativas.

4.4 Erro Total Ponderado

Os resultados anteriores deixam claro que, dentre os quatro erros de reconstrução, alguns são mais relevantes que outros para a classificação. Esta foi a motivação para a criação do erros de reconstrução ponderados: atribuir pesos maiores aos erros mais importantes para melhorar a performance do classificador. O problema em usar pesos, no entanto, é como encontrá-los. É possível tentar alguns valores e seguir fazendo ajustes manualmente, teste após teste. Mas também é possível usar alguma técnica de otimização que procura por estes valores de forma automatizada. Neste trabalho, usamos um algoritmo genético [Gol89]: ele inicialmente escolhe um conjunto de valores aleatórios e usa operações inspiradas na biologia evolucionária, como seleção, *crossover* e mutação, para melhorá-los ao longo de várias gerações. Os indivíduos são avaliados usando uma função de aptidão (*fitness function*, em inglês) e os mais aptos têm mais chances de estar na geração seguinte e gerar descendentes.

Para usar um algoritmo genético, precisamos definir como os pesos serão codificados em cromossomos [Mit96]. Neste trabalho, cada cromossomo foi codificado como um vetor de pesos a ser multiplicado pelos erros de reconstrução, de acordo com a equação (3.6). Para calcular a aptidão de um cromossomo, cada imagem do conjunto de treinamento foi classificada usando os pesos representado pelo indivíduo. Por fim, a aptidão de um induvíduo foi definida como a quantidade de imagens incorretamente classificadas, já que a implementação usada tenta minimizar o resultado da função de aptidão.

Desta forma, a função visa classificar corretamente o maior número de amostras de treinamento usando um dado conjunto de pesos. Entretanto, ela também poderia ser ajustada para permitir que o classificador encontre mais pedestres, em troca de uma taxa maior de falsas detecções, por exemplo. Isto pode ser alcançado atribuindo uma maior importância às imagens de pedestre corretamente classificadas (verdadeiros positivos) na função de classificação. Este comportamento é útil para aplicações que necessitam de uma maior taxa de verdadeiros positivos e toleram mais detecções espúrias. Nos experimentos, usamos a implementação de algoritmos genéticos fornecida pelo Matlab.

Como este algortimo executa uma otimização local e portanto não é ótimo, os experimentos foram executados cinco vezes para cada classificador e valor de k. Então, o conjunto de pesos que produziu os melhores resultados foi escolhido entre estes cinco; eles são mostrados na Tabela 4.4. Finalmente, classificamos as amostras de teste em três cenários: usando erro total ponderado (equação 3.6), usando apenas o erro parcial de escala de cinza ponderado (equação 3.13) e usando apenas o erro parcial de bordas ponderado (equação 3.14). Todos os classificadores produziram curvas ROC que apresentaram maior área sob a curva que a versão correspondente sem pesos, para praticamente todos os valores de k, indicando que o erro ponderado é um melhor critério de classificação. O resultado completo é listado na Tabela 4.3.

k	PONDERADO			NÃO-PONDERADO		
	CINZA	BORDAS	TOTAL	CINZA	BORDAS	TOTAL
400	0,9612	0,9946	0,9958	0,9598	0,9874	0,9750
300	0,9703	0,9948	0,9976	0,9682	0,9882	0,9800
200	0,9791	0,9909	0,9941	0,9766	0,9897	0,9876
100	0,9859	0,9981	0,9986	0,9846	0,9943	0,9936
50	0,9906	0,9958	0,9983	0,9903	0,9946	0,9969
25	0,9920	0,9918	0,9993	0,9929	0,9932	0,9983

Tabela 4.3 AUCs do classificador ponderado para diferentes quantidades de PCs (k). Para facilitar a comparação, as AUCs dos classificadores originais também são mostradas. Observe que no classificador que usa o erro total, a versão ponderada sempre apresenta um melhor desempenho. Já nos classificadores parciais, a versão original superou a ponderada apenas quando $k = 25$.

k	CINZA		BORDAS		TOTAL			
	w_{gp}	w_{gn}	w_{ep}	w_{en}	w_{gp}	w_{ep}	w_{gn}	w_{en}
400	0,8594	0,8661	0,8367	0,8659	0,6016	0,5811	0,5078	0,8995
300	0,2352	0,2374	0,8341	0,8631	0,5555	0,4887	0,3782	0,9510
200	0,5607	0,5708	0,2769	0,2785	0,6337	0,6248	0,5971	0,7537
100	0,6848	0,6902	0,7220	0,7399	0,4182	0,8823	0,3949	0,9582
50	0,8212	0,8255	0,3809	0,3828	0,6265	0,9422	0,6473	0,9244
25	0,9980	0,9785	0,9158	0,9106	0,6173	1,3837	0,4962	1,6800

Tabela 4.4 Pesos obtidos com relação às amostras treinamento. O algoritmo genético foi executado separadamente para cada um dos três classificadores por cinco vezes para cada valor de k e o conjunto que produziu o melhor resultado foi selecionado. Os pesos funcionam efetivamente como uma medida da importância de cada erro. Observe que para os classificadores parciais os valores geralmente indicam que o conjunto negativo possui maior importância, corroborando o resultado sugerido pela Figura 4.2. Já no classificador do erro total, os pesos atribuem maior importância às imagens de borda, que foram obtidas com o filtro Sobel.

A Figura 4.6 compara as AUCs do classificador do erro total ponderado e não-ponderado, mostrando que este primeiro produz uma curva mais estável para os diferentes números de PCs usados. Por fim, plotamos a distribuição do erro de reconstrução total ponderado, que pode ser visto na Figura 4.7. Desta vez, observa-se uma fronteira clara de separação entre as amostras de diferentes classes.

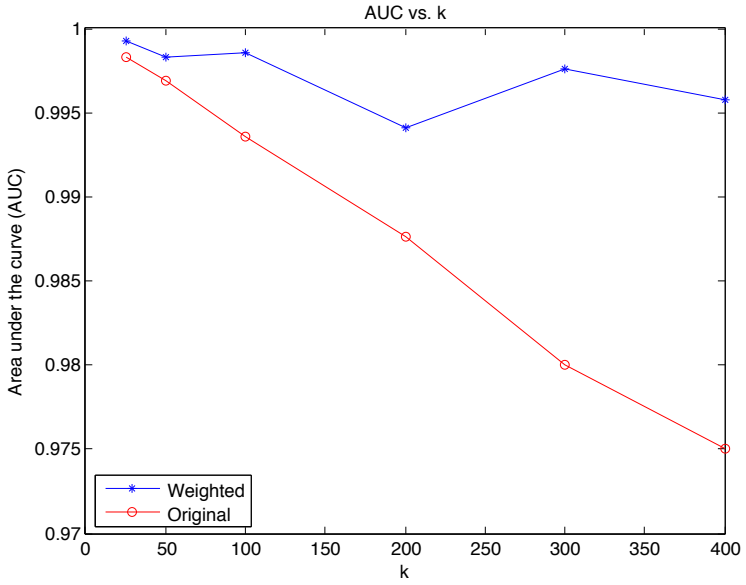

Figura 4.6 Comparação entre as AUCs do classificador ponderado e do classificador original para o erro total, em diferentes valores de k. A diferença é mais pronunciada para maiores valores de k e se torna pequena quando k diminiu. Ainda assim, a AUC do classificador ponderado é sempre maior, o que indica que o erro ponderado é um critério de classificação melhor que o erro não-ponderado. Observe também que a linha do classificador ponderado se mantém mais estável com a variação de k.

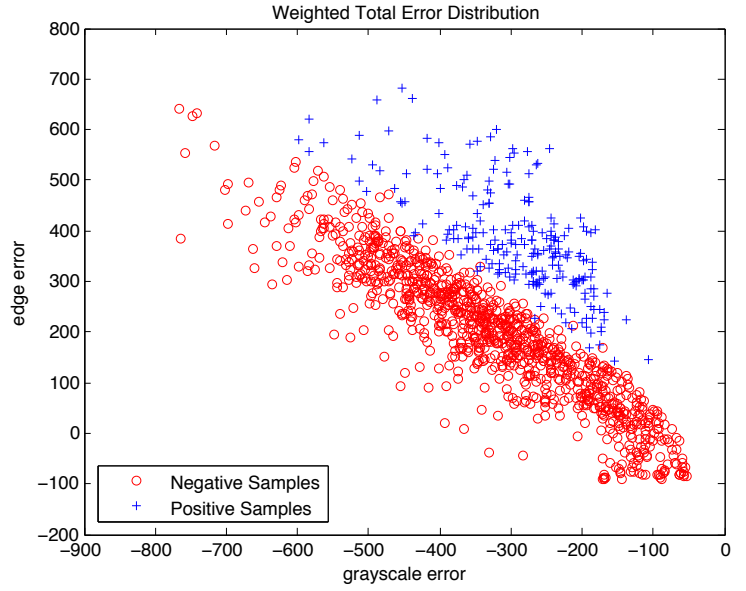

Figura 4.7 Distribuição dos erros de reconstrução total ponderadados das amostras de teste, para $k = 100$. Compare com o terceiro gráfico da Figura 4.3: aqui a fronteira entre os erros das amostras positivas e negativas é bem mais clara.

4.5 Comparação Entre os Algoritmos de Borda

Até agora, todos os experimentos foram executados usando imagens de borda geradas pelo filtro Sobel, como indicado no trabalho original de Malagón-Borja e Fuentes [MBF09]. Entretanto, esta escolha parece arbitrária: os autores não mencionam por que este filtro em particular foi escolhido dentre os vários existentes, alguns deles reconhecidamente superiores para a maioria dos casos, como o detector de Canny [GW07].

Para avaliar como a escolha da técnica de detecção de bordas afeta a classificação, comparamos o desempenho do sistema usando mais três algoritmos: operadores cruzados de Roberts, operadores de Prewitt e o detector de bordas de Canny, além dos operadores de Sobel já usados. Cada um destes métodos é avaliado em duas situações: na primeira, a imagem de bordas é formada diretamente pela soma dos gradientes verticais e horizontais obtidos (ou diagonais, no caso dos operadores de Roberts), segundo a equação (4.1); na segunda, a imagem de bordas é obtida a partir da binarização destes gradientes, após limiarização e afinamento (e, adicionalmente, supressão de não-máximos e histerese para o detector de Canny) [GW07]. Tem-se então, sete cenários distintos além do original, ilustrados pela Figura 4.8, que mostra novamente imagens de pedestres, acompanhadas das respectivas imagens de borda para cada algoritmo.

$$G = |G_x| + |G_y| \tag{4.1}$$

ALGORITMO	TEMPO (MS)
Sobel (Gradiente)	1,608
Sobel (Binarizado)	4,953
Roberts (Gradiente)	1,353
Roberts (Binarizado)	4,499
Prewitt (Gradiente)	1,597
Prewitt (Binarizado)	4,637
Canny (Gradiente)	4,812
Canny (Binarizado)	9,851

Tabela 4.5 Tempo médio em milissegundos para computação de uma imagem de borda, para cada algoritmo.

O processo de treinamento (Seção 3.1) foi executado em cada uma destes cenários, assim como o algoritmo genético para obtenção dos pesos dos erros ponderados (Seção 4.4). Então, procedemos com a classificação das amostras de teste para cada técnica, usando quatro tipos de erro: erro de reconstrução total (equação 3.4), erro de reconstrução total ponderado (equação 3.6), erro parcial de bordas (equação 3.10) e erro parcial de bordas ponderado (equação 3.14), para diferentes quantidades de PCs. O valor das AUCs de cada classificador é mostrado nas Tabelas 4.6, 4.7, 4.8 e 4.9.

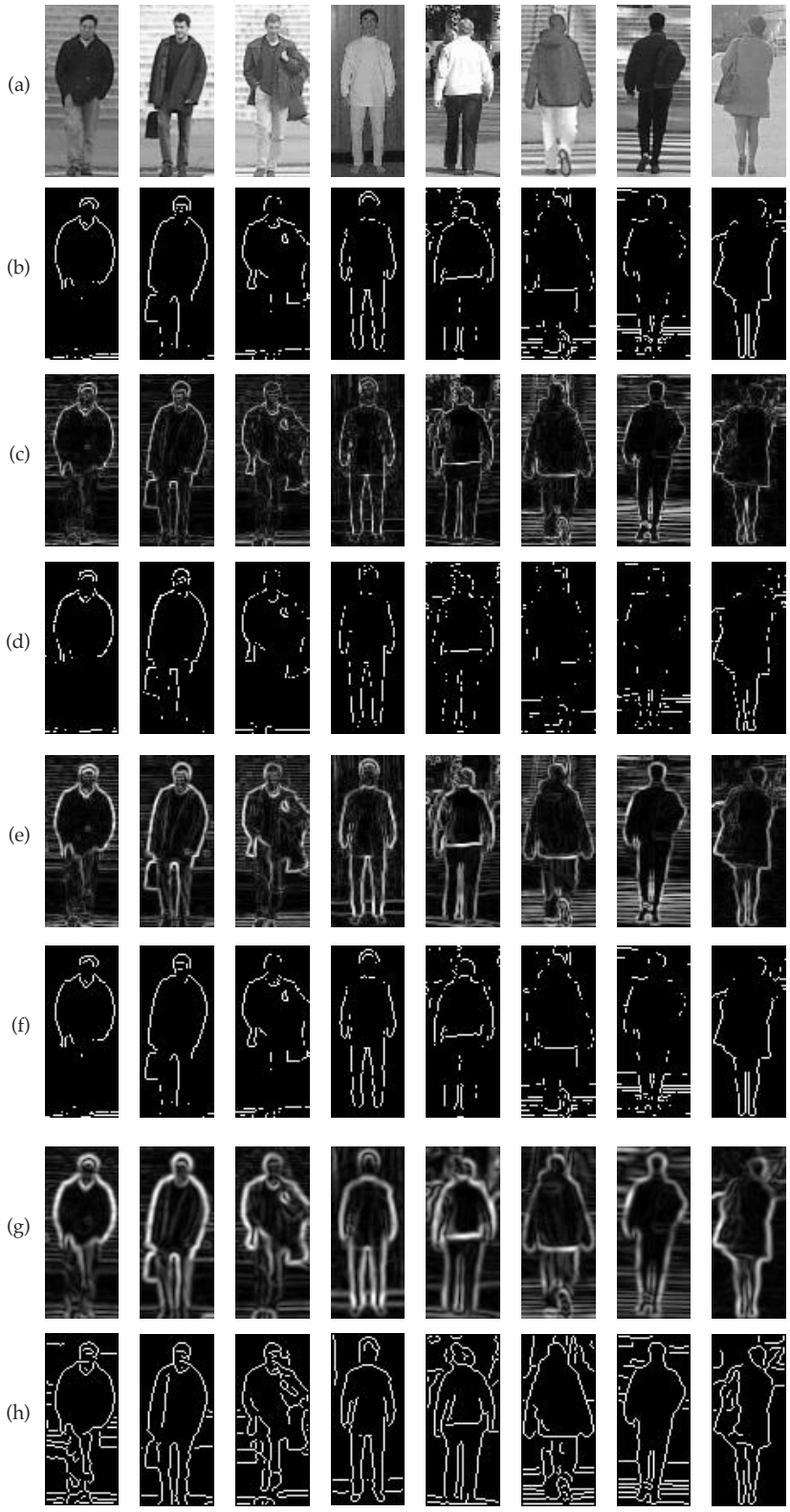

Figura 4.8 Imagens de bordas geradas por cada técnica. A linha (a) mostra as imagens em escala de cinza originais; a linha (b) mostra as imagens de borda obtidas com o filtro Sobel binarizado; a linha (c) o filtro de Roberts; a linha (d) o filtro de Roberts binarizado; a linha (e) o filtro de Prewitt; a linha (f) o filtro de Prewitt binarizado; a linha (g) o filtro de Canny; e a linha (h) o filtro de Canny binarizado.

k	GRADIENTE				BINARIZADO			
	ORIGINAL		PONDERADO		ORIGINAL		PONDERADO	
	BORDAS	TOTAL	BORDAS	TOTAL	BORDAS	TOTAL	BORDAS	TOTAL
400	0,9874	0,9750	0,9946	0,9958	0,9757	0,9860	0,9702	0,9766
300	0,9882	0,9800	0,9948	0,9976	0,9772	0,9880	0,9762	0,9915
200	0,9897	0,9876	0,9909	0,9941	0,9828	0,9918	0,9829	0,9882
100	0,9943	0,9936	0,9981	0,9986	0,9848	0,9941	0,9823	0,9938
50	0,9946	0,9969	0,9958	0,9983	0,9845	0,9972	0,9820	0,9942
25	0,9932	0,9983	0,9918	0,9993	0,9842	0,9982	0,9831	0,9979
Máx.	0,9946	0,9983	0,9981	0,9993	0,9848	0,9982	0,9831	0,9979
Média	0,9912	0,9886	0,9943	0,9973	0,9815	0,9926	0,9795	0,9904

Tabela 4.6 AUCs dos classificadores que usam o filtro Sobel binarizado (à direita). Para facilitar a comparação, as AUCs do classificadores que usam o filtro Sobel na versão de gradientes, mostradas anteriormente, também são listadas (à esquerda).

k	GRADIENTE				BINARIZADO			
	ORIGINAL		PONDERADO		ORIGINAL		PONDERADO	
	BORDAS	TOTAL	BORDAS	TOTAL	BORDAS	TOTAL	BORDAS	TOTAL
400	0,9885	0,9756	0,9952	0,9830	0,9496	0,9804	0,9265	0,9746
300	0,9890	0,9810	0,9928	0,9870	0,9540	0,9852	0,9439	0,9001
200	0,9878	0,9870	0,9956	0,9917	0,9578	0,9881	0,9537	0,9822
100	0,9906	0,9928	0,9957	0,9951	0,9549	0,9924	0,9530	0,9734
50	0,9932	0,9962	0,9958	0,9987	0,9588	0,9971	0,9550	0,9942
25	0,9931	0,9977	0,9946	0,9978	0,9588	0,9972	0,9589	0,9858
Máx.	0,9932	0,9977	0,9958	0,9987	0,9588	0,9972	0,9589	0,9942
Média	0,9904	0,9884	0,9950	0,9922	0,9557	0,9901	0,9485	0,9684

Tabela 4.7 AUCs dos classificadores que usam o filtro de Roberts (na versão de gradientes, à esquerda, e na versão binarizada, à direita), para diversos valores de k.

k	GRADIENTE				BINARIZADO			
	ORIGINAL		PONDERADO		ORIGINAL		PONDERADO	
	BORDAS	TOTAL	BORDAS	TOTAL	BORDAS	TOTAL	BORDAS	TOTAL
400	0,9873	0,9727	0,9879	0,9882	0,9721	0,9857	0,9203	0,9850
300	0,9888	0,9790	0,9938	0,9863	0,9768	0,9887	0,9626	0,9886
200	0,9907	0,9864	0,9956	0,9989	0,9802	0,9916	0,9708	0,9922
100	0,9945	0,9938	0,9961	0,9974	0,9818	0,9943	0,9815	0,9954
50	0,9951	0,9967	0,9960	0,9992	0,9832	0,9970	0,9831	0,9976
25	0,9925	0,9982	0,9930	0,9990	0,9831	0,9980	0,9827	0,9939
Máx.	0,9951	0,9982	0,9961	0,9992	0,9832	0,9980	0,9831	0,9976
Média	0,9915	0,9878	0,9937	0,9948	0,9795	0,9926	0,9668	0,9921

Tabela 4.8 AUCs dos classificadores que usam o filtro de Prewitt (na versão de gradientes, à esquerda, e na versão binarizada, à direita), para diversos valores de k.

k	GRADIENTE				BINARIZADO			
	ORIGINAL		PONDERADO		ORIGINAL		PONDERADO	
	BORDAS	TOTAL	BORDAS	TOTAL	BORDAS	TOTAL	BORDAS	TOTAL
400	0,9910	0,9880	0,9816	0,9931	0,9967	0,9980	0,9312	0,9637
300	0,9940	0,9931	0,9927	0,9832	0,9973	0,9982	0,9974	0,9990
200	0,9953	0,9961	0,9951	0,9888	0,9976	0,9984	0,9949	0,8555
100	0,9968	0,9986	0,9959	0,9981	0,9978	0,9991	0,9976	0,9992
50	0,9970	0,9992	0,9970	0,9996	0,9965	0,9996	0,9968	0,9998
25	0,9966	0,9992	0,9964	0,9993	0,9952	0,9994	0,9956	0,9899
Máx.	0,9970	0,9992	0,9970	0,9996	0,9978	0,9996	0,9976	0,9998
Média	0,9951	0,9957	0,9931	0,9937	0,9969	0,9988	0,9856	0,9679

Tabela 4.9 AUCs dos classificadores que usam o detector de Canny (na versão de gradientes, à esquerda, e na versão binarizada, à direita), para diversos valores de k.

Verificamos que todos os algoritmos de borda são capazes de gerar sistemas com boas taxas de classificação. O melhor resultado foi obtido pelo classificador que usa o detector de Canny binarizado quando $k = 50$ (AUC $= 0,9998$), seguido pelo detector de Canny não-binarizado também quando $k = 50$ (AUC $= 0,9996$), pelo operador de Sobel não-binarizado quando $k = 25$ (AUC $= 0,9993$) e pelo operador de Prewitt binarizado quando $k = 50$ (AUC $= 0,9992$); todos usam o erro de reconstrução total ponderado.

No geral, quando consideramos o erro de reconstrução total não-ponderado, observamos que as maiores AUCs são obtidas pelos classificadores que usam o detector de Canny binarizado (média de 0,9988). Entretanto, quando consideramos o erro total ponderado, o operador de Sobel gera as maiores AUCs (média de 0,9973). Os demais algoritmos de borda apresentaram um desempenho ligeiramente inferior. Em todos os cenários, os melhores desempenhos são obtidos para pequenas quantidades de PCs, quando $k \leq 100$. Isto indica que a maior parte da informação sobre as imagens está concentrada nos primeiros PCs, confirmando o que mostram os autoespectros da Seção 4.1.

Observamos ainda que os erros ponderados são ineficientes para algoritmos de borda binarizados. Com exceção de Canny, todos os classificadores que usam técnicas de borda binarizadas tiveram seu desempenho piorado para erros ponderados, em praticamente todos os valores de k. O contrário acontece com os classificadores que usam técnicas de borda não-binarizadas. Neles, os erros ponderados melhoraram significativamente a performance do sistema. Com o detector de bordas de Canny, no entanto, este comportamento é menos regular, tanto para a versão binarizada como para a versão não-binarizada.

CAPÍTULO 5

Conclusão

Neste trabalho analisamos um classificador que usa reconstrução de imagens com PCA para detecção de pedestres. Nele, quatro conjuntos de componentes principais (PCs) são utilizados: dois originados de imagens em escala de cinza e de imagens de borda que contêm pedestres, respectivamente, e dois originados de imagens em escala de cinza e de bordas que não contêm pedestres, respectivamente. Para classificar uma imagem, o detector usa estes conjuntos de PCs para calcular quatro reconstruções de imagens e compara os respectivos erros de reconstrução para determinar se uma dada imagem contém ou não um pedestre.

Para entender como os erros individuais contribuem para o desempenho geral do sistema, comparamos dois classificadores: um que usa apenas as reconstruções obtidas a partir dos PCs positivos e outro que usa apenas as reconstruções dos PCs negativos. Nos experimentos, ambos apresentaram performances ruins quando comparados ao classificador completo, indicando que as reconstruções positivas e negativas são de fato relevantes para a classificação. O papel crucial das reconstruções do conjunto negativo é particularmente surpreendente, já que as amostras de não-pedestres não são relacionadas e não contêm um padrão específico. Portanto, concluímos que a importância das reconstruções negativas não está no que elas caracterizam, mas sim no que elas não caracterizam. Elas contribuem para aumentar o erro de reconstrução total quando a imagem não contém um pedestre, o que viabiliza uma boa classificação.

Também investigamos a relevância dos erros de imagens em escala de cinza e de imagens de borda. Em outro experimento, comparamos a acurácia de um classificador que usa apenas reconstruções em escala de cinza com outro classificador que usa apenas reconstruções de borda. Desta vez, ambos se saíram muito bem, com resultados próximos aos do classificador que usa o erro total. Isto sugere que há bastante redundância entre informações de imagens em escala de cinza e de imagens de bordas. Entre os classificadores parciais, o detector baseado apenas em bordas apresentou um melhor resultado, indicando que estas informações são mais relevantes. De fato, em alguns casos o classificador de bordas obteve inclusive um melhor desempenho que o classificador total e, quando seus resultados foram inferiores, a queda de acurácia foi menor que 1%. Desta forma, é possível acelerar significativamente o tempo de classificação usando apenas informações de borda e, ao mesmo tempo, manter a acurácia de classificação praticalmente inalterada.

Como os experimentos mostram que cada erro tem uma relevância distinta para a classificação, propomos um classificador que usa uma versão ponderada do erro de reconstrução total. O problema com pesos, no entanto, é como encontrar valores

adequados para eles. Para obter uma aproximação de forma automatizada, usamos um algoritmo genético com uma função de aptidão que objetiva atingir a melhor taxa de classificação com relação às amostras de treinamento. Este classificador apresentou um desempenho melhor e mais estável que a versão original em diversos cenários, para praticamente os números de PCs testados nos experimentos.

O classificador por reconstrução original gera as imagens de borda usando apenas os operados de Sobel, sem, no entanto, justificar por que este filtro em particular foi escolhido dentre os demais existentes. Para observar como a escolha do algoritmo de geração de bordas afeta o desempenho do sistema, testamos o classificador com mais três técnicas de detecção de bordas: Roberts, Prewitt e Canny, além do Sobel original. Cada um deles foi usado em duas versões: uma em que a imagem de bordas é resultado direto dos gradientes gerados e outra em que estes gradientes são binarizados. Os experimentos mostram que todos os algoritmos de borda são capazes de atingir uma performance satisfatória, embora alguns saiam-se melhores que outros. O melhor desempenho foi obtido pelo classificador que usa o algoritmo de Canny binarizado, seguido por Sobel e Prewitt, ambos não-binarizados. Entre eles, o filtro de Sobel parece ser mais vantajoso para a maioria das aplicações, já que é capaz de obter excelentes resultados e é pelo menos seis vezes mais rápido que o algoritmo de Canny.

Este trabalho obteve conclusões importantes, mas a pesquisa sobre o tema deve continuar. Trabalhos futuros incluem executar os experimentos em outros conjuntos de imagens e verificar se a abordagem por reconstrução é adequada a outros problemas de detecção em visão computacional, como detecção de faces. Pretendemos continuar desenvolvendo o classificador, transformando-o em um sistema de detecção completo, capaz de localizar pedestres em imagens maiores. Também é possível usar variações do PCA para calcular as reconstruções, como o 2D-PCA [LY05].

Referências Bibliográficas

[Alp04] E. Alpaydin. *Introduction to machine learning*. Adaptive computation and machine learning. MIT Press, 2004. 2.1, 2.1, 2.1, 2.2

[AT07] Yali Amit and Alain Trouvé. Pop: Patchwork of parts models for object recognition. *International Journal of Computer Vision*, 75:267–282, 2007. 1.2

[Cen00] Center for Biological and Computional Learning. CBCL Pedestrian Database #1, 2000. 4

[DT05] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886 –893 vol. 1, 2005. 1.2

[DTS06] Navneet Dalal, Bill Triggs, and Cordelia Schmid. Human detection using oriented histograms of flow and appearance. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, volume 3952 of *Lecture Notes in Computer Science*, pages 428–441. Springer Berlin / Heidelberg, 2006. 1.2

[FA10] A. Frank and A. Asuncion. UCI Machine Learning Repository, 2010. 2.3

[Faw06] Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861 – 874, 2006. 4.2, 4.3

[FFJS08] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid. Groups of adjacent contour segments for object detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(1):36 –51, 2008. 1.2

[FMR08] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1 –8, 2008.

[FPZ03] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–264 – II–271 vol.2, 2003. 1.2

[FTVG06] Vittorio Ferrari, Tinne Tuytelaars, and Luc Van Gool. Object detection by contour segment networks. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, volume 3953 of *Lecture Notes in Computer Science*, pages 14–28. Springer Berlin / Heidelberg, 2006. 1.2

[Gav07] D.M. Gavrila. A bayesian, exemplar-based approach to hierarchical shape matching. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(8):1408 –1421, 2007.

[GLSG10] D. Gerónimo, A.M. López, A.D. Sappa, and T. Graf. Survey of pedestrian detection for advanced driver assistance systems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(7):1239 –1258, 2010. 1.1

[Gol89] DE Goldberg. Genetic Algorithms in Search Optimization and Machine Learning, 0201157675, 1989. 4.4

[GP99] D.M. Gavrila and V. Philomin. Real-time object detection for ldquo;smart rdquo; vehicles. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, 1999. 1.2

[GW07] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Prentice Hall, 3rd edition, Boston, MA, USA, 2007. 4.5

[Hot33] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417 – 441, 1933. 2.5

[HTWM04] Weiming Hu, Tieniu Tan, Liang Wang, and S. Maybank. A survey on visual surveillance of object motion and behaviors. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 34(3):334 –352, 2004. 1.1

[Jia09] Xudong Jiang. Asymmetric principal component and discriminant analyses for pattern classification. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(5):931 –937,, May 2009. 2.3, 4

[Jol02] I.T. Jolliffe. *Principal component analysis*. Springer series in statistics. Springer, 2002. 2, 2.5

[KD07] Wenxiong Kang and Feiqi Deng. Research on intelligent visual surveillance for public security. In *Computer and Information Science, 2007. ICIS 2007. 6th IEEE/ACIS International Conference on*, pages 824 –829, 2007. 1.1

[KTZ05] M.P. Kumar, P.H.S. Ton, and A. Zisserman. Obj cut. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 18 – 25 vol. 1, 2005. 1.2

[LSS05] B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 878 – 885 vol. 1, 2005.

[LY05] M. Li and B. Yuan. 2D-LDA: A statistical linear discriminant analysis for image matrix. *Pattern Recognition Letters*, 26(5):527–532, 2005. 5

[MBF09] L. Malagón-Borja and O. Fuentes. Object detection using image reconstruction with PCA. *Image and Vision Computing*, 27(1-2):2–9, 2009. 3, 3.1, 3.1.2, 3.1.3, 4.5

[Mit96] Melanie Mitchell. *An Introduction to Genetic Algorithms*. Ignatius Press, San Francisco, 1996. 4.4

[MPP01] A. Mohan, C. Papageorgiou, and T. Poggio. Example-based object detection in images by components. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(4):349 –361, April 2001. 1.2

[MSZ04] Krystian Mikolajczyk, Cordelia Schmid, and Andrew Zisserman. Human detection based on a probabilistic assembly of robust part detectors. In Tomás Pajdla and Jirí Matas, editors, *Computer Vision - ECCV 2004*, volume 3021 of *Lecture Notes in Computer Science*, pages 69–82. Springer Berlin / Heidelberg, 2004. 1.2

[OPZ06] Andreas Opelt, Axel Pinz, and Andrew Zisserman. A boundary-fragment-model for object detection. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, volume 3952 of *Lecture Notes in Computer Science*, pages 575–588. Springer Berlin / Heidelberg, 2006.

[Pea01] K. Pearson. LIII. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series 6*, 2(11):559–572, 1901. 2.5

[PP99] C. Papageorgiou and T. Poggio. Trainable pedestrian detection. In *Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on*, 1999. 1.2

[RW06] C.E. Rasmussen and C.K.I. Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, 2006. 1

[SBC05] J. Shotton, A. Blake, and R. Cipolla. Contour-based learning for object detection. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 503 – 510 Vol. 1, 2005.

[Shl09] J. Shlens. A tutorial on principal component analysis. *Systems Neurobiology Laboratory, University of California at San Diego*, 2009. 2.2

[SK87] L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. *Journal of the Optical Society of America A*, 4(3):519–524, Mar 1987. 2.5, 3

[SLS06] E. Seemann, B. Leibe, and B. Schiele. Multi-aspect detection of articulated objects. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, 2006.

[SNRD07] V.D. Shet, J. Neumann, V. Ramesh, and L.S. Davis. Bilattice-based logical reasoning for human detection. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1 –8, 2007. 1.2

[SS02] B. Schölkopf and A.J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. Adaptive computation and machine learning. MIT Press, 2002. 1

[Str03] G. Strang. *Introduction to linear algebra*. Wellesley-Cambridge Press, 2003. 2.2

[TK09] S. Theodoridis and K. Koutroumbas. *Pattern recognition*. Elsevier/Academic Press, 2009. 2, 2.3, 2.3

[TP91] Mattew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–89, 1991. 2.5, 3

[TPM07] O. Tuzel, F. Porikli, and P. Meer. Human detection via classification on riemannian manifolds. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1 –8, 2007. 1.2

[VJ01] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, 2001. 1.2

[VJS03] P. Viola, M.J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 734 –741 vol.2, 2003. 1.2

[WN07] Bo Wu and R. Nevatia. Simultaneous object detection and segmentation by boosting local shape feature based classifier. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1 –8, 2007. 1.2

[WS06] J. Winn and J. Shotton. The layout consistent random field for recognizing and segmenting partially occluded objects. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 37 – 44, 2006. 1.2

[WYH05] Ying Wu, Ting Yu, and Gang Hua. A statistical field model for pedestrian detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 1023 – 1030 vol. 1, 2005. 1.2

[ZD05] Liang Zhao and L.S. Davis. Closely coupled object detection and segmentation. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 454 – 461 Vol. 1, 2005. 1.2

Este volume foi tipografado em LaTeX na classe UFPEThesis (www.cin.ufpe.br/~paguso/ufpethesis).
Para detalhes sobre este documento, clique aqui.