Visão Computacional

Eigenfaces

Fisherface

Local Binary Patterns

Erick Lima Correia

RM: 333671

Eingenfaces

O problema com a representação da imagem que nos é dada é sua alta dimensionalidade. As imagens bidimensionais em escala de cinza abrangem um espaço vetorial tridimensional ; portanto, uma imagem com pixels já está em um espaço tridimensional da imagem. A questão é: todas as dimensões são igualmente úteis para nós? Só podemos tomar uma decisão se houver alguma variação nos dados; portanto, o que estamos procurando são os componentes que representam a maioria das informações. A Análise de Componentes Principais (PCA) foi proposta independentemente por Karl Pearson (1901) e Harold Hotelling(1933) para transformar um conjunto de variáveis possivelmente correlacionadas em um conjunto menor de variáveis não correlacionadas. A idéia é que um conjunto de dados de alta dimensão seja frequentemente descrito por variáveis correlacionadas e, portanto, apenas algumas dimensões significativas sejam responsáveis pela maior parte das informações. O método PCA localiza as direções com a maior variação nos dados, chamados componentes principais.

O método Eigenfaces executa o reconhecimento de face por:

- Projetando todas as amostras de treinamento no subespaço PCA.
- Projetando a imagem da consulta no subespaço PCA.
- Localizando o vizinho mais próximo entre as imagens de treinamento projetadas e a imagem de consulta projetada.

Fisherfaces

A Análise de Componentes Principais (PCA), que é o núcleo do método Eigenfaces, encontra uma combinação linear de recursos que maximiza a variação total dos dados. Embora essa seja claramente uma maneira poderosa de representar dados, ela não considera nenhuma classe e, portanto, muitas informações discriminatórias podem ser perdidas ao jogar componentes fora. Imagine uma situação em que a variação nos seus dados seja gerada por uma fonte externa, que seja a luz. Os componentes identificados por um PCA não contêm necessariamente nenhuma informação discriminatória; portanto, as amostras projetadas são unidas e uma classificação se torna impossível (consulte http://www.bytefish.de/wiki/pca_lda_with_gnu_octave, por exemplo).

A Análise Discriminante Linear realiza uma redução de dimensionalidade específica da classe e foi inventada pelo grande estatístico Sir RA Fisher . Ele o usou com sucesso para classificar flores em seu artigo de 1936, *O uso de múltiplas medidas em problemas taxonômicos* [Fisher36] . Para encontrar a combinação de recursos que separa melhor as classes, a Análise Discriminante Linear maximiza a proporção de dispersão entre classes para dentro das classes, em vez de maximizar a dispersão geral. A idéia é simples: as mesmas classes devem se agrupar firmemente, enquanto classes diferentes estão o mais longe possível uma da outra na representação de dimensão inferior. Isso também foi reconhecido por Belhumeur , Hespanha eKriegman e eles aplicaram uma Análise Discriminante para enfrentar o reconhecimento em [BHK97] .

O método Fisherfaces aprende uma matriz de transformação específica da classe, para que eles não capturem a iluminação tão obviamente quanto o método Eigenfaces. A Análise Discriminante,

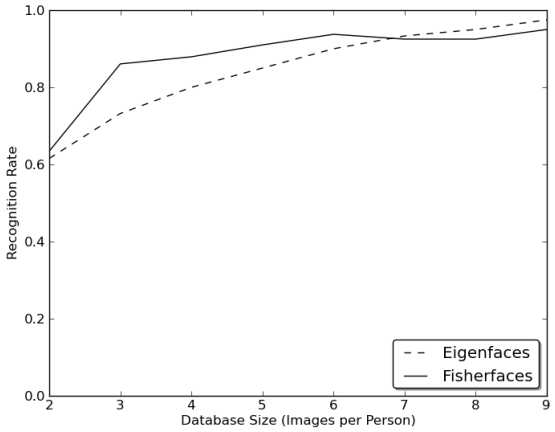
em vez disso, encontra as características faciais para discriminar entre as pessoas. É importante mencionar que o desempenho dos Fisherfaces também depende muito dos dados de entrada. Praticamente dito: se você aprender os Fisherfaces apenas para fotos bem iluminadas e tentar reconhecer rostos em cenas mal iluminadas, é provável que o método encontre os componentes errados (apenas porque esses recursos podem não ser predominantes em imagens mal iluminadas). Isso é um tanto lógico, pois o método não teve chance de aprender a iluminação.

Os Fisherfaces permitem uma reconstrução da imagem projetada, assim como os Eigenfaces. Porém, como identificamos apenas os recursos para distinguir os assuntos, você não pode esperar uma boa reconstrução da imagem original. Para o método Fisherfaces, projetaremos a imagem de amostra em cada um dos Fisherfaces. Portanto, você terá uma boa visualização, que apresenta cada um dos Fisherfaces descritos.

Local Binary Patterns Histograms

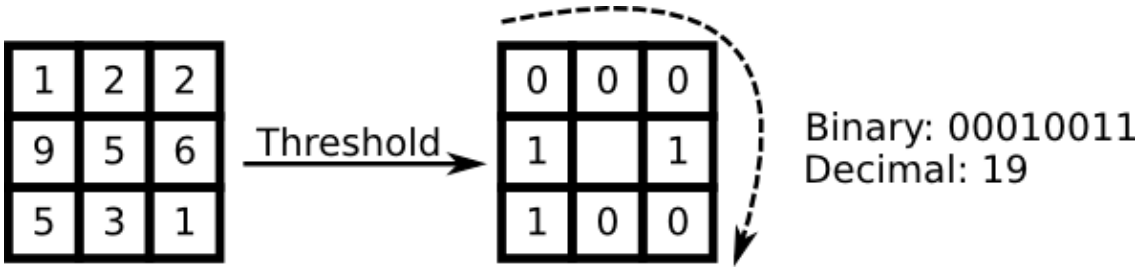
Os autofaces e os fisherfaces adotam uma abordagem um tanto holística para enfrentar o reconhecimento. Você trata seus dados como um vetor em algum lugar de um espaço de imagem de alta dimensão. Todos sabemos que a alta dimensionalidade é ruim e, portanto, um subespaço de menor dimensão é identificado, onde (provavelmente) informações úteis são preservadas. A abordagem Eigenfaces maximiza a dispersão total, o que pode levar a problemas se a variação for gerada por uma fonte externa, porque componentes com uma variação máxima em todas as classes não são necessariamente úteis para classificação (consulte http://www.bytefish.de / wiki / pca_lda_with_gnu_octave) Portanto, para preservar algumas informações discriminativas, aplicamos uma Análise Discriminante Linear e otimizamos conforme descrito no método Fisherfaces. O método Fisherfaces funcionou muito bem ... pelo menos no cenário restrito que assumimos em nosso modelo.

Agora a vida real não é perfeita. Você simplesmente não pode garantir configurações perfeitas de luz em suas imagens ou em 10 imagens diferentes de uma pessoa. E daí se houver apenas uma imagem para cada pessoa? Nossas estimativas de covariância para o subespaço *podem* estar terrivelmente erradas, assim como o reconhecimento. Lembra que o método Eigenfaces teve uma taxa de reconhecimento de 96% no banco de dados da AT&T? Quantas imagens realmente precisamos para obter estimativas úteis? Aqui estão as taxas de reconhecimento Rank-1 do método Eigenfaces e Fisherfaces no AT&T Facedatabase, que é um banco de dados de imagens bastante fácil:



Portanto, para obter boas taxas de reconhecimento, você precisará de pelo menos 8 (+ - 1) imagens para cada pessoa e o método Fisherfaces não ajuda realmente aqui. A experiência acima é um resultado validado em cruz de 10 vezes, realizado com a estrutura facerec em: https://github.com/bytefish/facerec . Como não é uma publicação, não apóio esses números com uma profunda análise matemática. Por favor, dê uma olhada em [KM01] para uma análise detalhada de ambos os métodos, quando se trata de pequenos conjuntos de dados de treinamento.

Portanto, algumas pesquisas se concentraram em extrair recursos locais de imagens. A idéia é não olhar para a imagem inteira como um vetor de alta dimensão, mas descrever apenas os recursos locais de um objeto. Os recursos que você extrai dessa maneira terão uma baixa dimensionalidade implicitamente. Uma ótima ideia! Mas você logo observará que a representação de imagem que recebemos não sofre apenas variações de iluminação. Pense em coisas como escala, tradução ou rotação nas imagens - sua descrição local deve ser pelo menos um pouco robusta contra essas coisas. Assim comoSIFT, a metodologia de padrões binários locais tem suas raízes na análise de textura 2D. A idéia básica dos padrões binários locais é resumir a estrutura local em uma imagem comparando cada pixel com sua vizinhança. Pegue um pixel como centro e limite seus vizinhos. Se a intensidade do pixel central for maior - igual ao seu vizinho, denote-o com 1 e 0, se não. Você terá um número binário para cada pixel, assim como 11001111. Portanto, com 8 pixels adjacentes, você terminará com 2 ^ 8 combinações possíveis, chamadas *padrões binários locais* ou, às vezes, chamados de *códigos LBP* . O primeiro operador LBP descrito na literatura realmente usou uma vizinhança 3 x 3 fixa, assim:



Descrição algorítmica do método LBPH

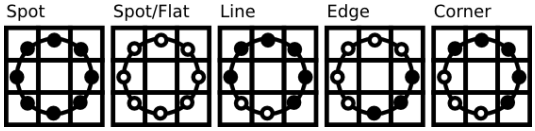
Uma descrição mais formal do operador LBP pode ser fornecida como:

$$L\ B\ P(\ xc,\ yc) = \sum p = 0P - 12ps\ (\ ip - euc)$$

, com (xc, yc)como pixel central com intensidade ic; e eunsendo a intensidade do pixel vizinho. s é a função de sinal definida como:

$$s\ (\ x\ ) = \{\ 10\ 0se\ \ x \geq 0outro$$

Essa descrição permite capturar detalhes de granulação muito fina nas imagens. De fato, os autores foram capazes de competir com os resultados mais avançados para a classificação de texturas. Logo após a publicação do operador, observou-se que uma vizinhança fixa falha ao codificar detalhes com diferentes escalas de escala. Portanto, o operador foi estendido para usar uma vizinhança variável em [3] . A idéia é alinhar um número abreviado de vizinhos em um círculo com um raio variável, o que permite capturar os seguintes bairros:



imagem

Para um dado ponto (xc, yc)a posição do vizinho (xp, yp) , p ∈ P pode ser calculado por:

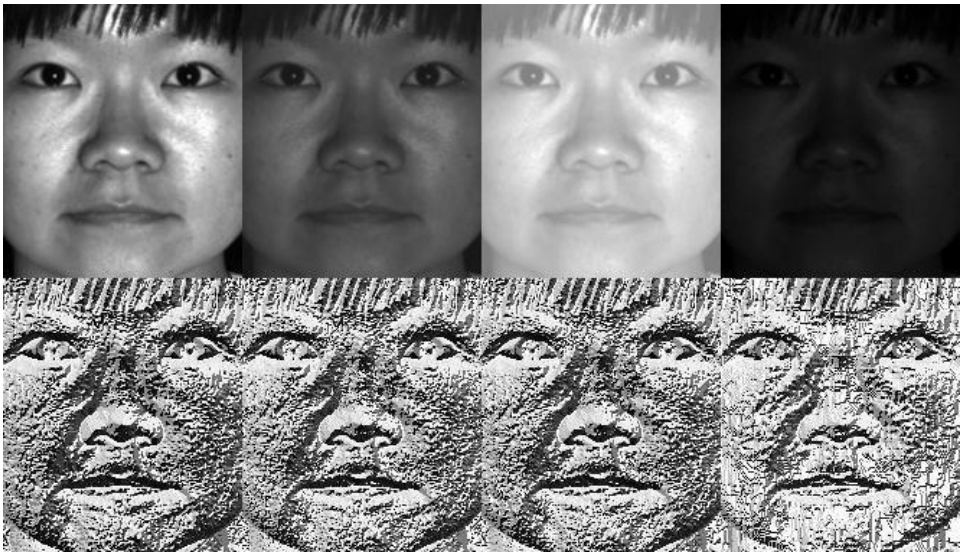
$$xpyp == xc + R\ cos(\ 2\ \pi pP)yc - R\ sin(\ 2\ \pi pP)$$

Onde Ré o raio do círculo e P é o número de pontos de amostra.

O operador é uma extensão dos códigos LBP originais, por isso às vezes é chamado de LBP estendido (também conhecido como LBP circular). Se uma coordenada de pontos no círculo não corresponder às coordenadas da imagem, o ponto será interpolado. A ciência da computação tem vários esquemas de interpolação inteligentes, a implementação do OpenCV faz uma interpolação bilinear:

$$f(\ x\ ,\ y) \approx [\ 1 - xx]\ [\ f(\ 0\ ,\ 0\ )f(\ 1\ ,\ 0\ )f(\ 0\ ,\ 1\ )f(\ 1\ ,\ 1\ )]\ [\ 1 - yy]\ .$$

Por definição, o operador LBP é robusto contra transformações monotônicas da escala de cinza. Podemos verificar isso facilmente, observando a imagem LBP de uma imagem modificada artificialmente (para ver como é uma imagem LBP!):



imagem

Portanto, o que resta fazer é como incorporar as informações espaciais no modelo de reconhecimento de rosto. A locais e extraia um histograma representação proposta por Ahonen et. al [3] é dividir a imagem LBP em mregiões de cada uma. O vetor de característica espacialmente aprimorado é então obtido concatenando os histogramas locais (não os mesclando). Esses histogramas são chamados de Local Binary Patterns Histograms.

Conclusão

Após observar o artigo " Reconhecimento de Rosto com OpenCV" e o caso de uso apresentado posso presumir que o metodo Fischerfaces resulta em valores de confiança muito mais baixos (mais precisos) do que o EigenFaces. Pórém o LBPH é um dos algoritmos de reconhecimento facial mais fáceis.
PCA é um algoritmo de redução de dimensão não supervisionado, enquanto o LDA é supervisionado
O PCA é bom em limpeza externa e a LDA pode aprender o desvio dentro da classe
Esses dois métodos extraem apenas 1
1ª e 2ª estatística momentos Combination A combinação de PCA e LDA poderia melhorar a desempenho
O PCA serve como o primeiro passo para o processamento de vários tipos técnica de reconhecimento facial
Técnicas de redução de dimensão são frequentemente usadas no reconhecimento facial

Referências

https://docs.opencv.org/3.4/da/d60/tutorial_face_main.html
https://www.learnopencv.com/eigenface-using-opencv-c-python/
https://tanay-choudhary.github.io/project/face-recognition
https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b