FIAP

MBA em Inteligência Artificial & Machine Learning

Tecnologia de Processamento de Imagem

Projeto Final de Avaliação Substitutiva

Avaliação de métodos de reconhecimento facial utilizando visão computacional (Eigenfaces, Fisherfaces e Local Binary Patterns Histograms**)**

Aluno: Aneilson Benicio

Professor: Michel Pereira Fernandes

São Paulo

2019

Sumário

1. Objetivo

Este projeto tem por objetivo aprofundar o conhecimento nos classificadores de faces presentes na biblioteca OpenCV (versão 3): Eigenfaces, Fisherfaces e Local Binary Patterns Histograms.

2. Eigenfaces

2.1. Motivação

Considerada por muitos como a primeira tecnologia de reconhecimento facial em funcionamento foi desenvolvida por Sirovich e Kirby em 1987 para detecção e reconhecimento de faces. Desde o seu desenvolvimento inicial, surgiram muitas extensões para o método original e muitos novos desenvolvimentos em sistemas automáticos de reconhecimento facial. Eigenfaces ainda é frequentemente considerado como um método de comparação de linha de base para demonstrar o desempenho mínimo esperado de tal sistema.

Este método realiza uma projeção linear do espaço da imagem em um espaço de dimensões menores. Seu funcionamento é similar ao PCA levando essa projeção linear que maximiza a dispersão de todas as imagens projetadas reduzindo o processamento necessário para fazer o cálculo de seus auto-vetores e auto-valores.

2.2. Funcionamento

Seja $X = \{x_1, x_2, \ldots, x_n\}$ um vetor aleatório com observações: $x_i \in R^d$.

1. Calcule a média μ

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

2. Calcule a matriz de covariância *S*

$$S = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T$$

3. Calculo os autovalores λ_i e autovetores v_i de S

$$Sv_i = \lambda_i v_i, i = 1, 2, \ldots, n$$

4. Ordene seus autovetores descendendo pelo seu autovalor. Os k principais componentes são os autovetores correspondentes aos k maiores autovalores.

Os k componentes principais do vetor observado x são dados por:

$$y = W^T(x - \mu)$$

Onde: $W = (v_1, v_2, \ldots, v_k)$.

A reconstrução a partir da base do PCA é dada por:

$$x = Wy + \mu$$

Onde: $W = (v_1, v_2, \ldots, v_k)$.

O método eigenface executa o reconhecimento facial seguindo os pontos abaixo:

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

2.3. Pontos Positivos

Um dos métodos mais antigos e conhecido na área de reconhecimento facial. Insensitivo à pouca variação de iluminação.

2.4. Pontos Negativos

A utilização do PCA (Principal Component Analysis) para redução de dimensionalidade, gera um impacto negativo neste classificador pois alterações na luminosidade são consideradas componentes de variação entre as faces**.**

Levando em consideração que o algoritmo compara e calcula a distância da imagem de entrada com todas as imagens de treinamento, podemos citar o alto custo computacional como fator negativo, pois quanto maior a base de treinamento maior será o custo computacional para o reconhecimento.

2.5. Exemplos

Notebook em python no diretório lab.

3. Fisherfaces

3.1. Motivação

Também utiliza redução de dimensionalidade com o LDA (Linear Discriminant Analysis), que tem por objetivo minimizar variações na própria classe e maximizar entre as classes. É um bom discriminador de múltiplas faces.

3.2. Funcionamento

Seja X um vetor aleatório com amostras retiradas de classes c:

$$X = \{X_1, X_2, \ldots, X_c\}$$
$$X_i = \{x_1, x_2, \ldots, x_n\}$$

1. As matrizes de dispersão S_B e *Sw* são calculadas como:

$$S_B = \sum_{i=1}^{c} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

$$S_W = \sum_{i=1}^{c} \sum_{x_j \in X_i} (x_j - \mu_i)(x_j - \mu_i)^T$$

Onde μ é a média total:

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$$

E μ_i é a média da classe $i \in \{1, \ldots, c\}$:

$$\mu_i = \frac{1}{|X_i|}\sum_{x_j \in X_i} x_j$$

2. O algoritmo agora procura uma projeção W, que maximize o critério de separabilidade de classes:

$$W_{opt} = \arg\max_W \frac{|W^T S_B W|}{|W^T S_W W|}$$

3. A solução para esse problema de otimização é dada pela resolução geral do problema de autovalores (Eigenvalue):

$$\begin{aligned} S_B v_i &= \lambda_i S_w v_i \\ S_W^{-1} S_B v_i &= \lambda_i v_i \end{aligned}$$

4. A classificação de Sw é no máximo (N-c), com N amostras e c classes. Em problemas de reconhecimento de padrões, o número de amostras N é quase sempre menor do que a dimensão dos dados de entrada (o número de pixels), então a matriz de dispersão Sw se torna singular. A solução se dá executando uma análise de Componente Principal nos dados e projetando as amostras no espaço (N-c). Uma Análise Linear Discriminante será então executada nos dados reduzidos, porque Sw não é mais singular.

   O problema de otimização pode ser reescrito como:

$$\begin{aligned} W_{pca} &= \arg\max_W |W^T S_T W| \\ W_{fld} &= \arg\max_W \frac{|W^T W_{pca}^T S_B W_{pca} W|}{|W^T W_{pca}^T S_W W_{pca} W|} \end{aligned}$$

A matriz de transformação W, que projeta uma amostra no espaço (c-1) é então dada por:

$$W = W_{\text{fld}}^{\text{T}} W_{\text{pca}}^{\text{T}}$$

O método fisherfaces executa o reconhecimento facial seguindo os pontos abaixo:

- Cálculo da face média por classe
- Cálculo de face média geral
- Transformação das imagens em vetores
- Construção da matriz de dispersão intra-classe
- Cálculo das fisherfaces (LDA)
- Cálculo dos vetores de características
- Cálculo da Similaridade

3.3. Pontos positivos

Insensitivo a certa variação de iluminação e expressões faciais. Apresenta melhor resultado que o eigenface.

3.4. Pontos negativos

A variação de pose e oclusão parcial pode prejudicar significativamente os resultados.

3.5. Exemplos

Notebook em python no diretório lab.

4. Local Binary Pattern Histogram (LBPH)

4.1. Motivação

O LBP é um operador de textura simples, porém eficiente, que rotula os pixels de uma imagem ao limitar a vizinhança de cada pixel e considera o resultado como um número binário.

Foi descrito pela primeira vez em 1994 (LBP) e, desde então, foi considerado um recurso poderoso para a classificação de textura. Ainda, quando o LBP é combinado com os histograms of oriented gradients (HOG), ele melhora o desempenho da detecção consideravelmente em alguns conjuntos de dados.

Usando o LBP combinado com histogramas, podemos representar as imagens do rosto como um vetor de dados simples.

Utiliza parâmetros locais preservando relações espaciais. Este algoritmo divide a imagem de um rosto em diversas janelas que posteriormente serão transformadas em histogramas.

4.2. Funcionamento

Uma descrição mais formal do LBPH pode ser dada como:

$$\mathrm{LBP}(x_c, y_c) = \sum_{p=0}^{P-1} 2^p s(i_p - i_c)$$

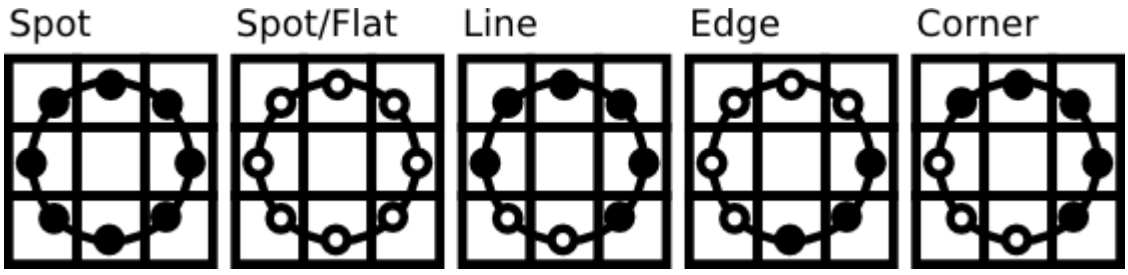
, onde (x_c, y_c) como pixel central com intensidade i_c e i_n sendo a intensidade do pixel vizinho. **s** é a função de sinal definica como:

$$s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{else} \end{cases} \qquad (1)$$

Esta descrição permite capturar detalhes muito finos nas imagens. De fato, os autores conseguiram competir com os resultados de última geração para classificação de texturas. Logo após a publicação do operador, notou-se que

uma vizinhança (neighborhood) fixa não codifica detalhes que diferem em escala. Então o operador foi estendido para usar uma vizinhança variável.

A ideia é alinhar um número abusivo de vizinhos em um círculo com um raio variável, o que permite capturar os seguintes bairros (neighborhoods):



Para um dado ponto (x_c, y_c), a posição do vizinho $(x_p, y_p), p \in P$, pode ser calculado por:

$$x_p = x_c + R\cos(\frac{2\pi p}{P})$$

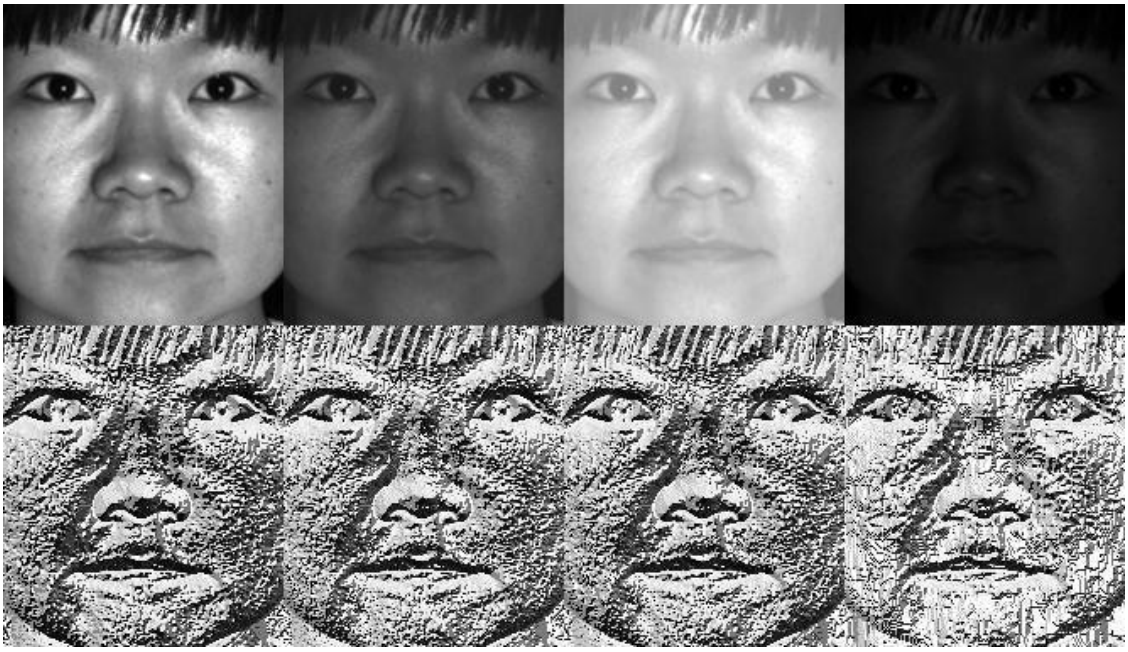
$$y_p = y_c - R\sin(\frac{2\pi p}{P})$$

, onde R é 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 um monte de esquemas inteligentes de interpolação, a implementação do OpenCV faz uma interpolação bilinear:

$$f(x,y) \approx \begin{bmatrix} 1-x & x \end{bmatrix} \begin{bmatrix} f(0,0) & f(0,1) \\ f(1,0) & f(1,1) \end{bmatrix} \begin{bmatrix} 1-y \\ y \end{bmatrix}.$$

Por definição, o operador LBP é robusto contra transformações monótonas em escala de cinza. Podemos facilmente verificar isso olhando para a imagem LBP de uma imagem modificada artificialmente (assim você pode ver como é uma imagem LBP):



Então, é dividir a imagem LBP em m regiões locais e extrair um histograma de cada uma delas. O vetor de recursos aprimorado espacialmente é então obtido concatenando os histogramas locais (não mesclando-os). Esses histogramas são chamados de Histog Patterns Binary Local

4.3. Pontos positivos

LBPH é um dos algoritmos de reconhecimento facial mais fáceis de compreender.

Pode representar características locais nas imagens.

É possível obter excelentes resultados (principalmente em um ambiente controlado).

É robusto contra transformações monotônicas em escala de cinza.

É o mais robusto dentre os 3, no OpenCV

4.4. Pontos negativos

4.5. Exemplos

Notebook em python no diretório lab.

5. CONCLUSÃO

Executando os algoritmos para detecção de face podemos afirmar que o eigenface possui o menor nível de confiança, sendo assim, não recomendamos para uma aplicação prática, pois a probabilidade de falha muito é grande e inaceitável como por exemplo reconhecimento fácil para acesso a ambientes ou sistemas.

O fisherface, apresenta resultados melhores que o eigenface porém ainda pensando em aplicações práticas o nível de confiança é bem similar. Com isso podemos afirmar que também é insatisfatório para aplicações práticas no mundo atual.

O algoritmo LBPH é o mais eficiente, sendo possível treinar o modelo até com pequenas quantidades de amostras. Mostrou muito eficiente para reconhecimento facial, enquanto os outros dois modelos não conseguiram reconhecer mesmo utilizando uma quantidade maior de amostrar e imagem em escala de cinza.

6. Referências Bibliográficas

BISSI, T. D. Reconhecimento Facial com os algoritmos Eigenfaces e

Fisherfaces. Uberlandia, Brasil. 2018.

FERNADES, M, P. Material didático fornecido no curso de MBA em Inteligencia Artificial e Machine Learning. São Paulo, 2018.

PRADO, S. P. Comparação de técnicas de reconhecimento facial para pidentificação de presença em um ambiente real e semicontrolado. São Paulo, Brasil. 2018.

Face Recognition with Open CV. Disponível em:

< https://docs.opencv.org/3.0-beta/modules/face/doc/facerec/facerec_tutorial.html#fisherfaces-in-opencv> Acesso em: 16 jan. 2019

Wikipedia. Disponível em:

< https://en.wikipedia.org/wiki/Local_binary_patterns>

Acesso em: 16 jan. 2019

Scholarpedia. Disponível em:

<http://www.scholarpedia.org/article/Eigenfaces>

Acesso em: 18 jan. 2019