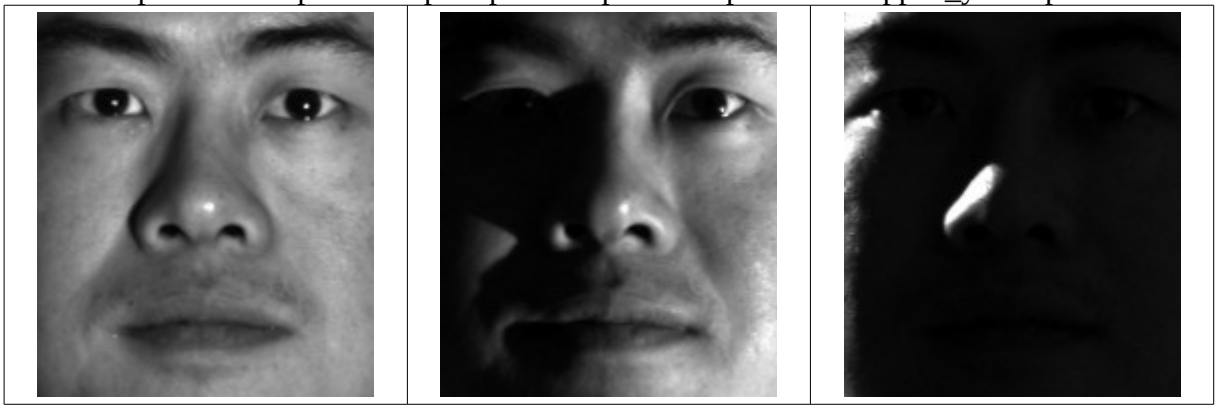
PSI-3471: Fundamentos de Sistemas Eletrônicos Inteligentes
Primeiro semestre de 2017 2º exercício-programa Prof. Hae
~~**Data de entrega: 18/06/2017 (domingo) até 24:00 horas**~~
Data de entrega: 22/06/2017 (quinta) até 24:00 horas
Obs. 1: Cada dia de atraso acarreta uma perda de 1 ponto no exercício.
Obs. 2: Este EP deve ser resolvido individualmente. EPs iguais receberão nota zero.

São dados dois bancos de dados de faces:

- AT&T face database. Este banco de imagens contém 10 imagens de cada um dos 40 indivíduos, com fundo escuro. É muito fácil reconhecer as faces neste banco de dados pois mesmo um algoritmo muito simples atingiu taxa de erro 1,75%. Consequentemente serve apenas para testes iniciais.
  http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html
  Cópia local: http://www.lps.usp.br/hae/psi3471/ep2-2017/att_faces.zip

- Yale face database B cropped. Este banco de imagens contém 2414 imagens de 38 indivíduos, alinhados e recortados manualmente. Há mudança de condição de iluminação que dificulta o reconhecimento. Nota: As imagens "Ambient" não fazem parte do banco de dados.
  http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html
  Cópia local: http://www.lps.usp.br/hae/psi3471/ep2-2017/cropped_yale.zip



O objetivo deste exercício é fazer um programa C++ para reconhecer as faces usando alguma técnica de extração de características e aprendizagem de máquina. Vamos fazer os testes de reconhecimento usando a validação cruzada “leave one out”. Isto é, vamos retirar uma imagem do banco de dados para teste, treinar algum método de aprendizagem de máquina usando todas as imagens restantes como amostras de treinamento (excluindo a imagem teste escolhida) e tentar reconhecer o indivíduo da imagem-teste. Vamos repetir esse processo usando como imagem-teste cada uma das imagens do banco de dados. O programa deve imprimir os nomes dos arquivos classificados incorretamente, a classificação correta e a classificação atribuída pelo algoritmo. Também deve imprimir a taxa de erros. Por exemplo:

```
diretorio>ep2 c:\diretorio\att_faces\todos.txt parametro1 parametro2 ...
Arquivo s35/3.pgm. Correta=34. Classificada=14.
Arquivo s35/1.pgm. Correta=34. Classificada=14.
Arquivo s19/9.pgm. Correta=18. Classificada=10.
Arquivo s1/2.pgm.  Correta=0.  Classificada=17.
Arquivo s1/10.pgm. Correta=0.  Classificada=12.
Arquivo s1/1.pgm.  Correta=0.  Classificada=15.
Arquivo s28/8.pgm. Correta=27. Classificada=36.
totalErros=7  nTestes=400  porcentagemErros=1.75%
```

Deixei os arquivos "todos.txt" contendo em cada linha a informação "nomeImagem;classificação". Você pode usar esses arquivos para ler mais facilmente todas as imagens amostras com as respectivas classificações verdadeiras. Nos arquivos "quatro.txt" há informações de apenas os quatro primeiros indivíduos. Sugiro que façam testes iniciais rapidamente usando arquivos "quatro.txt" e ajustem os parâmetros para minimizar o erro. Depois, rodem os testes finais (que demoram) em "todos.txt" usando os parâmetros escolhidos na fase anterior.

Nota 1: Em OpenCV há três classes prontas que fazem reconhecimento de faces:
http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html
Vamos fazer este exercício ***sem usar essas classes*** ou qualquer outra biblioteca ou função pronta para reconhecimento de faces. Você pode usar essas classes para ver a taxa de erros que se pode atingir. Neste exercício, é permitido:

- Usar as classes prontas para aprendizagem de máquina de OpenCV ou de qualquer outra biblioteca.
- Ler artigos sobre reconhecimento de face e implementá-la.

Nota 2: Se você conseguir atingir taxas de erro "razoáveis" (tipo 3% de erro no primeiro banco de dados e 30% de erro no segundo banco de dados) já está ótimo. Se atingir taxas de erro muito menores do que isso, o exercício pode valer até 12. Para ter uma ideia, copio abaixo as taxas de erros que eu obtive (escolhendo adequadamente os parâmetros) usando um algoritmo simples de aprendizagem e usando as 3 classes prontas do OpenCV.

	ATT quatro.txt	ATT todos.txt	Yale quatro.txt	Yale todos.txt
Simples	0%	1.75%	14.45%	22.99%
EigenFaces	0%	1.50%	26.17%	46.77%
FisherFaces	0%	2.00%	3.12%	5.97%
LBPH	0%	1.00%	0.78%	8.53%

Se alguém quiser implementar o método FisherFaces, veja:
Belhumeur, P. N., Hespanha, J., and Kriegman, D. Eigenfaces vs. Fisherfaces: Recognition Using ClassSpecific Linear Projection. IEEE Transactions on Pattern Analysis and Machine Intelligence 19, 7 (1997),711–720.

Se alguém quiser implementar o método LBPH, veja:
Ahonen, T., Hadid, A., and Pietikainen, M. Face Recognition with Local Binary Patterns. Computer Vision - ECCV 2004 (2004), 469–481.

Dá a impressão de que LBPH é tranquilo para implementar.

Obs. 1: Pode usar (se quiser) a biblioteca Cekeikon/OpenCV.

Obs. 2: Entregue o programa-fonte (ep2.cpp) e um documento PDF (relatorio.pdf) ou DOC (relatorio.doc) com os comentários descrevendo o funcionamento do programa. O envio do relatório é obrigatório (veja o anexo). Destaque as taxa de erros obtidas nos quatro casos (ATT quatro, ATT todos, Yale quatro e Yale todos). Deixe explícito os parâmetros que geraram as taxas de erro descritas.

(a) Se você fez o programa no ambiente usado na classe (Cekeikon/OpenCV), não é necessário enviar o programa executável. Se você fez o programa usando só OpenCV em Linux ou Mac, também não é necessário enviar o programa executável (desde que você não utilize nenhuma função exclusiva desses sistemas).

(b) Se você quiser usar alguma biblioteca diferente, converse antes com o professor.

Obs. 3: Compacte todos os arquivos como `SeuNome_Sobrenome.ZIP` e envie via edisciplinas.

Obs. 4: Qualquer dúvida, entre em contato com o professor:

- `hae@lps.usp.br`

Anexo: Tópicos exigidos no relatório dos Exercícios Programas

Descrição do problema / objetivos

Descreva claramente o enunciado do problema a ser resolvido. Isso é importante para você se assegurar de que está resolvendo o problema pedido.

Técnica(s) utilizada(s) para resolver o problema

Descreva de forma clara quais algoritmos e técnicas foram necessários para resolver o problema. Utilize o nome adequado, se existir (por exemplo, filtragem Gaussiana, classificador SVM, algoritmo SIFT, etc.). Use elementos gráficos como imagens intermediárias e diagramas se necessário. Não copie e cole código fonte, a não ser que o mesmo seja de fato relevante. Se o fizer, comente-o. No relatório, o comentário é mais importante do que o código. Prefira o uso de pseudocódigo.

Ambiente de desenvolvimento utilizado

Em qual plataforma a solução foi desenvolvida? Em qual plataforma a solução será utilizada? Como o usuário pode compilar o programa? Quais bibliotecas foram utilizadas?

Operação

Como o usuário deve executar o programa? Quais os argumentos para execução? Há parâmetros necessários a serem configurados? Quais arquivos de entrada são necessários? Quais arquivos de saída são gerados?

Resultados Obtidos

Descreva os resultados obtidos. Qual é o tempo de processamento típico? O problema foi resolvido de forma satisfatória / robusta? Quais as limitações encontradas? Quais as sugestões de melhorias?

Referências

Descreva o material externo utilizado, como livros consultados, websites visitados, etc.