PSI5796: Processamento e Análise de Imagens e Vídeos
Primeiro período de 2019 2º exercício-programa Prof. Hae
Data de entrega: 02/06/2019 (domingo) 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.
Nota: Inicialmente, este EP era para ser sobre reconhecimento de faces. Só que, para isso, precisa de conceitos que não vimos. Por isso mudei de problema.

O objetivo deste exercício é usar rede neural convolucional (ou algum outro método de aprendizagem de máquina) para fazer super-resolução de imagens de rostos (isto é, aumentar a resolução das imagens).

Vamos usar o banco de dado de faces humanas do prof. Carlos E. Thomaz da FEI:
http://fei.edu.br/~cet/frontalimages_manuallyaligned_part1.zip
http://fei.edu.br/~cet/frontalimages_manuallyaligned_part2.zip
com 400 imagens JPG coloridas, com 360x260 pixels, de rostos frontais alinhados manualmente. Renomeei essas imagens, para que todas tenham nomes com 4 caracteres, por exemplo, 1a.jpg torne-se 001a.jpg. Reduzi a resolução dessas imagens para 48x32 pixels. Converti cada uma das imagens para sistema CieLAB, normalizei o componente L dessas imagens de forma que o pixel mais escuro se torne 0 e o mais claro se torne 255, e reconverti para sistema BGR, gerando 400 imagens ????y.png. Reduzi a resolução dessas imagens para 12x8, resultando em 400 imagens ????x.png.

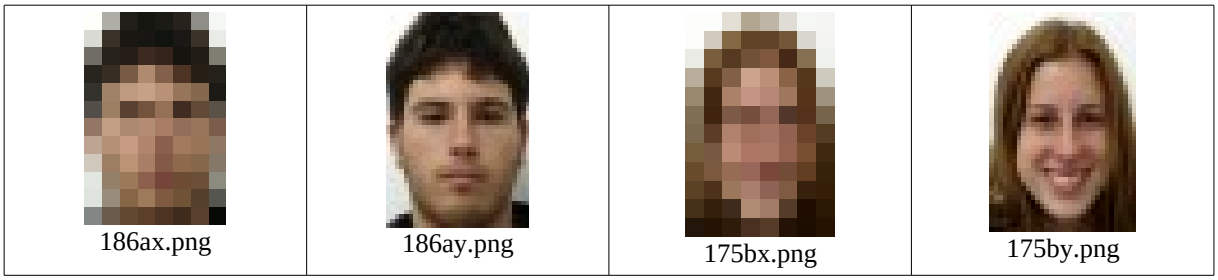

Figura 1: Imagens de entrada 12x8 (X) e as imagens de saída ideais 48x32 (Y).

O objetivo deste exercício é desenvolver um programa que recebe uma imagem 12x8 ????x.png e aumenta a resolução para 48x32, gerando a imagem processada ????p.png que seja a mais semelhante possível da imagem ideal ????y.png.

A diferença entre a imagem ideal *Y* e a processada *P* será medida pela métrica MSE (média do erro quadrático):

$$MSE(Y,P)=\frac{\sum_{l}\sum_{c}\sum_{b}[Y(l,c,b)-P(l,c,b)]^2}{L\times C\times B}$$

onde as imagens *Y* e *P* são imagens coloridas (3 bandas) em ponto flutuante com os valores no intervalo [0; 1], isto é, na conversão da imagem colorida 8 bits sem sinal para ponto flutuante 0 é mapeado em 0,0 e 255 é mapeado em 1,0; *Y*(*l*,*c*,*b*) indica o valor da imagem *Y* linha *l*, coluna *c* e banda *b*; *L*, *C* e *B* são os números de linhas, colunas e bandas das imagens *Y* e *P*.

Também criei 3 arquivos CSV (comma separated values), especificando quais pares de imagens devem ser usadas no treino, na validação e no teste:
treino.csv: 200 pares de imagens x y
valida.csv: 100 pares de imagens x y
teste.csv : 100 pares de imagens x y

Para ilustração, as seis primeiras linhas do treino.csv são:

```
001ax.png;001ay.png
001bx.png;001by.png
002ax.png;002ay.png
002bx.png;002by.png
005ax.png;005ay.png
005bx.png;005by.png
```

Esses arquivos estão em:

www.lps.usp.br/hae/psi5796/ep2-2019/feiFrontRed.zip

Faça três programas Python ou C++: treino, supres e mse. O programa treino.cpp ou treino.py deve fazer o treino utilizando as pares de imagens listadas no arquivo treino.csv como amostras de treinamento, através de algum método de aprendizagem de máquina. Durante o treino, você pode ajustar os parâmetros da rede usando as pares de imagens listadas no arquivo valida.csv. Durante o treino, é proibido olhar as imagens listadas em teste.csv.

Você deve gravar a rede obtida pelo treino como arquivo “rede.net” (C++/tiny_dnn) ou "rede.h5" (Python/keras/tensorflow) no diretório default.

```
$ treino diretorio_onde_esta_o_banco_de_dados
$ python treino.py diretorio_onde_esta_o_banco_de_dados
```

(Usa banco de dados no diretório "diretorio_onde_esta_banco_de_dados" para gerar rede.net ou rede.h5 no diretório default.)

A saída de treino.cpp deve ser algo como:

MSE médio de validacao: 0.00432426
Tempo de processamento: 12.4 minutos.

Depois do treino, supres.cpp deve carregar rede.net do diretório default e uma imagem 12x8 e gravar a imagem 48x32 obtida pela super-resolução.

```
$ supres ../diretorio/004bx.png 004bp.png
$ python supres.py ../diretorio/004bx.png 004bp.png
```

(Usa rede.net do diretório default para aumentar resolução da imagem 12x8 ../diretorio/004bx.png e grava a imagem 48x32 004bp.png no diretório default.)

O programa mse.cpp deve fazer superresolução de todas as imagens de treino, validação e teste e calcular a média dos MSEs obtidos para cada conjunto de imagens. MSE deve ser calculado considerando que os valores dos pixels são pontos flutuantes de 0 a 1.

```
$ mse diretorio_onde_esta_banco_de_dados
```

(Usa rede.net do diretório default e banco de dados do diretório "diretorio_onde_esta_banco_de_dados" para calcular a média dos MSEs de treino.csv, valida.csv e teste.csv)

```
Media de MSE de treino.csv: 0.00389246
Media de MSE de valida.csv: 0.00432426
Media de MSE de teste.csv:  0.00434647
```

Todos os programas devem emitir uma mensagem amigável se os argumentos do programa não estiverem no formato esperado, assim como se o programa não encontrar algum arquivo necessário.

Nota: Antes de começar a programar, sugiro fortemente que leiam o artigo [Dong2016] ou outros materiais sobre a super-resolução usando rede neural convolucional. Caso contrário, é muito provável que acabem usando uma estrutura da rede inadequada para o problema.

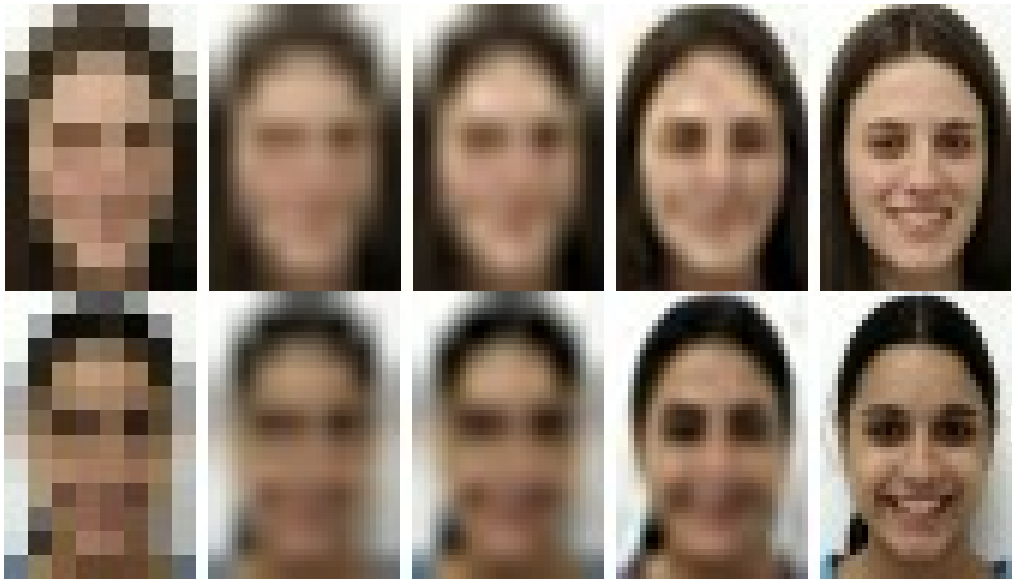

Figura 2: Da esquerda para direta: Imagem original 16x8; interpolação bilinear; interpolação bicúbica; super-resolução com rede neural convolucional; saída ideal 48x32.

Tabela 1: As médias de MSEs de diferentes métodos de reamostragem (considerando pixels de 0 a 1):

	treino.csv	valida.csv	teste.csv
nearest	0.0183094	0.018546	0.0188638
linear	0.0149238	0.0151444	0.015358
cubic	0.0126045	0.0127348	0.0128601
rede convoluc.	0.00389246	0.00432426	0.00434647

O erro da rede convolucional é da ordem de 3 vezes menor do que o erro dos métodos cúbico ou linear.

Obs. 1: Pode usar (se quiser) a biblioteca OpenCV/Tiny_dnn ou OpenCV/Keras

Obs. 2: Entregue os 3 programas-fontes (treino.cpp, supres.cpp e mse.cpp), a redes obtida (rede.net ou rede.h5) e um documento PDF (relatorio.pdf) com os comentários descrevendo o funcionamento do programa, o tempo de processamento e as taxas de erro obtidas. O envio do relatório é obrigatório (veja o anexo).

(a) Se você fez o programa no ambiente usado na classe (C++/Cekeikon/OpenCV/Tiny_dnn, Python/OpenCV/Keras/Tensorflow), basta entregar os programas-fontes para poder corrigir o seu programa.

(b) Se você quiser usar alguma biblioteca diferente ou programar em outro ambiente, converse antes com o professor.

Obs. 3: Compacte todos os arquivos como SeuNome_Sobrenome.ZIP e envie um email para:

- hae@lps.usp.br

Se você enviar mais de um email, considerarei somente o último email enviado, apagando os anteriores.

Referências:

- [Dong2016] Dong, C., Loy, C. C., He, K., & Tang, X. (2016). Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, *38*(2), 295-307.

Anexo: Relatórios dos exercícios programas

O mais importante numa comunicação escrita é que o leitor entenda, sem esforço e inequivocamente, o que o escritor quis dizer. O texto ficar "bonito" é um aspecto secundário. Se uma (pseudo) regra de escrita dificultar o entendimento do leitor, essa regra está indo contra a finalidade primária da comunicação. No site do governo americano [1], há regras denominadas de "plain language" para que comunicações governamentais sejam escritas de forma clara. As ideias por trás dessas regras podem ser usadas em outros domínios, como na escrita científica. Resumo abaixo algumas dessas ideias.

(1) Escreva para a sua audiência. No caso do relatório, a sua audiência será o professor ou o monitor que irá corrigir o seu exercício. Você deve enfocar na informação que o seu leitor quer conhecer. Não precisa escrever informações que são inúteis ou óbvias para o seu leitor.

(2) Organize a informação. Você é livre para organizar o relatório como achar melhor, porém sempre procurando facilitar o entendimento do leitor. Seja breve. Quebre o texto em seções com títulos claros. Use sentenças curtas. Elimine as frases e palavras que podem ser retiradas sem prejudicar o entendimento. Use sentenças em ordem direta (sujeito-verbo-predicado).

(3) Use palavras simples. Use o tempo verbal o mais simples possível. Evite cadeia longa de nomes, substituindo-os por verbos (em vez de "desenvolvimento de procedimento de proteção de segurança de trabalhadores de minas subterrâneas" escreva "desenvolvendo procedimentos para proteger a segurança dos trabalhadores em minas subterrâneas"). Minimize o uso de abreviações (para que o leitor não tenha que decorá-las). Use sempre o mesmo termo para se referir à mesma realidade (pode confundir o leitor se usar termos diferentes para se referir a uma mesma coisa). O relatório não é obra literária, não tem problema repetir várias vezes a mesma palavra.

(4) Use voz ativa. Deixe claro quem fez o quê. Se você utilizar oração com sujeito indeterminado ou na voz passiva, o leitor pode não entender quem foi o responsável (Ex: "Criou-se um novo algoritmo" - Quem criou? Você? Ou algum autor da literatura científica?). O site do governo americano diz: "Passive voice obscures who is responsible for what and is one of the biggest problems with government writing."

(5) Use exemplos, diagramas, tabelas, figuras e listas. Ajudam bastante o entendimento.

O relatório deve conter pelo menos as seguintes informações:

Identificação

Seu nome, número USP, nome da disciplina, etc.

Breve enunciado do problema

Apesar do enunciado do problema ser conhecido ao professor/monitor, descreva brevemente o problema que está resolvendo. Isto tornará o documento compreensível para alguma pessoa que não tem o enunciado do EP à mão.

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

Descreva quais técnicas você usou para resolver o problema. Se você mesmo inventou a técnica, descreva a sua ideia, deixando claro que a ideia foi sua. Se você utilizou alguma técnica já conhecida, utilize o nome próprio da técnica (por exemplo, filtragem Gaussiana, algoritmo SIFT, etc.) juntamente com alguma referência bibliográfica onde a técnica está descrita. Use elementos gráficos como ima-

1 https://plainlanguage.gov/guidelines/

gens intermediárias e diagramas, pois ajudam muito a compreensão. Não "copie-e-cole" código-fonte, a não ser que seja relevante. Use preferencialmente o pseudo-código.

Ambiente de desenvolvimento utilizado

Em qual plataforma você desenvolveu o programa? Como o professor/monitor pode compilar o programa? Você utilizou que bibliotecas?

Operação

Como o professor/monitor pode executar o programa? Que argumentos são necessários para a execução do programa? Há parâmetros que devem ser 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? Qual foi a taxa de erro obtida? O problema foi resolvido de forma satisfatória?

Referências

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