PSI3472 Concepção e Implementação de Sistemas Eletrônicos Inteligentes
Segundo Semestre de 2023 Exercício-programa
Data de entrega: 08/10/2023 (domingo) até 23:59 horas

1. Introdução

Kaggle (www.kaggle.com) é uma plataforma de competição de ciência de dados e uma comunidade online de cientistas de dados e profissionais de aprendizado de máquina da Google. Kaggle permite que os usuários encontrem e publiquem conjuntos de dados, explorem e construam modelos em um ambiente de ciência de dados baseado na web, trabalhem com outros cientistas de dados e engenheiros de aprendizado de máquina e participem de competições para resolver desafios de ciência de dados (retirado da Wikipedia).

Kaggle fornece um ambiente semelhante ao Google Colab com GPU/TPU. O único problema de usar desafios de Kaggle como exercício-programa é que no plataforma tem soluções prontas. Mas vou confiar que vocês não vão dar "copy-paste" das soluções prontas. Para evitar apresentar algum programa da Kaggle como sendo sua solução, vou estabelecer algumas funções que devem ser implementadas obrigatoriamente.

Em Kaggle, há um conjunto de dados de raio-x de pulmão, COVID-QU-Ex dataset:
https://www.kaggle.com/datasets/anasmohammedtahir/covidqu
Todas as imagens estão no formato .PNG com 256×256 pixels. Este BD contém imagens de acordo com a tabela 1.

Tabela 1: Quantidade de imagens de BD COVID-QU-Ex.

	Train	Val	Test	Total
Covid-19	7658	1903	2395	11956
Non-Covid	7208	1802	2253	11263
Normal	6849	1712	2140	10701
Total	21715	5417	6788	33920

Como vamos treinar durante poucas épocas, não vamos usar os dados de validação para verificar se o modelo ficou bom. Vamos usar as imagens de validação/teste conjuntamente para calcular a acuracidade.

O objetivo deste EP é classificar as imagens validação/teste em Covid-19, infecção viral ou bacteriana não-Covid ou normal (paciente sadio), treinando o modelo com as imagens de treino. A figura 1 mostra algumas imagens desse BD. Note que é difícil classificar as imagens visualmente. Vamos fazer 3 classificações:

1) Você irá olhar as amostras de treino para aprender a distinguir visualmente as 3 classes. Depois, irá classificar visualmente 16 imagens de validação/teste escolhidas aleatoriamente. Anote a sua taxa de acerto.
2) Crie alguma rede neural inicializando os pesos aleatoriamente. Treine usando as imagens de treino e classifique as imagens de validação/teste. Para que o treino não demore excessivamente, vamos estabelecer que pode treinar o modelo somente durante 16 épo-

cas usando batch_size=16. Anote as taxas de acerto de treino e de validação/teste. Obtive taxas de acerto de validação/teste de aproximadamente 88,5%.

3) Faça transfer learning de alguma rede pré-treinada no ImageNet (por exemplo VGG, ResNet, EfficientNet, Xception, etc). Para que o treino não demore excessivamente, vamos estabelecer que pode treinar o modelo durante somente 8 épocas para o ajuste grosso (com modelo-base congelado) e 8 épocas para o ajuste fino (com modelo-base descongelado) sempre usando batch_size=16. Obtive taxa de acerto de validação/teste de aproximadamente 94,4%.

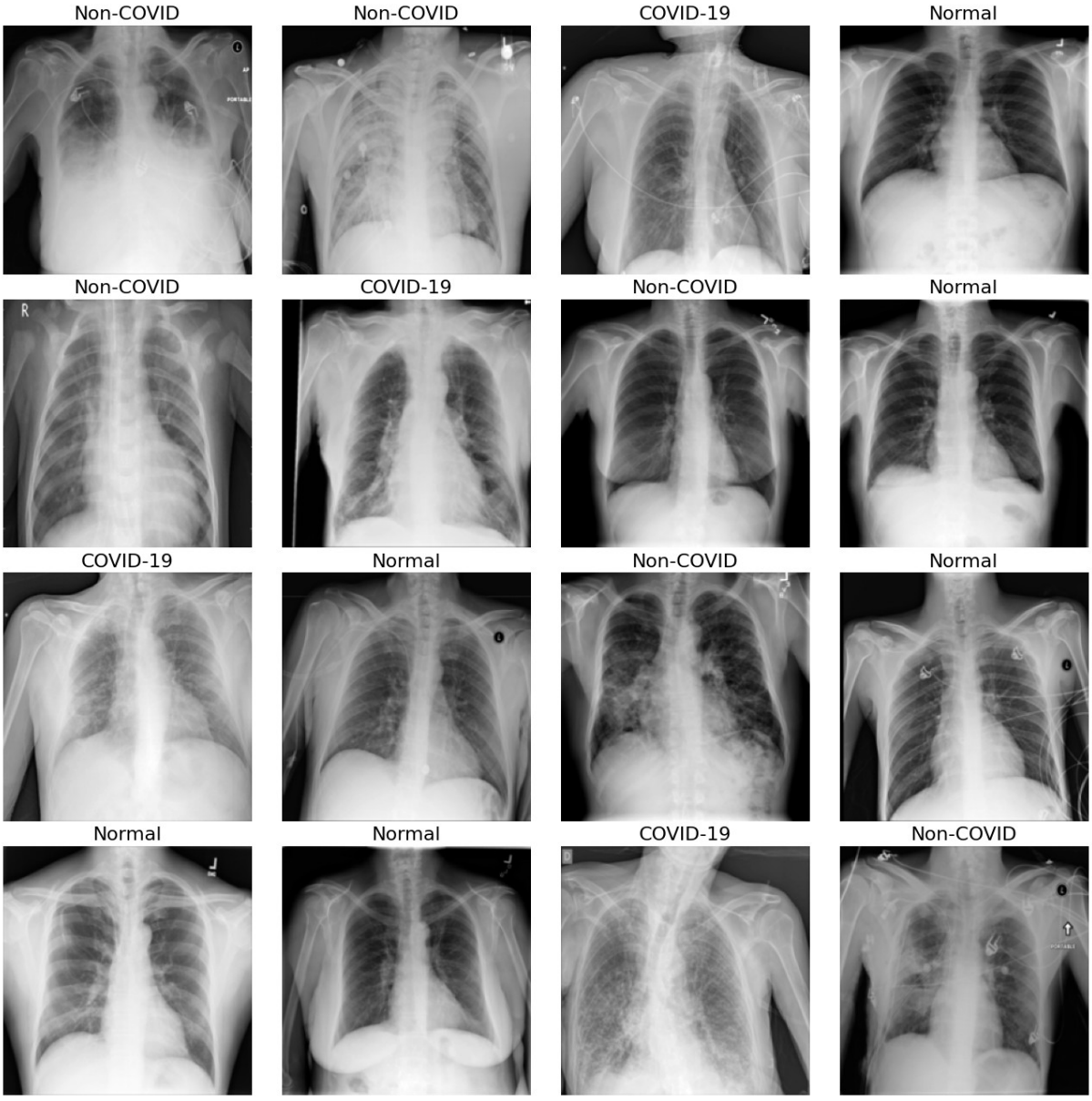

Figura 1: Exemplos de imagens de COVID-QU-Ex.

2. Fazer download do BD da Kaggle

Vou escrever o enunciado supondo que vocês vão usar o ambiente Colab. Se quiserem, podem usar o seu computador local, o ambiente da própria Kaggle ou algum outro ambiente. Neste caso, precisam fazer pequenas adaptações ao enunciado.

O site abaixo descreve como baixar um BD da Kaggle no Colab ou no seu computador local:
https://www.geeksforgeeks.org/how-to-import-kaggle-datasets-directly-into-google-colab/
Resumindo as informações desse site, você primeiro precisa criar uma conta na Kaggle:
https://www.kaggle.com/
Depois de criar a conta, na página principal da Kaggle, clique em "Account" que levará à página "Settings". Nessa página, clique em "Create New Token". Isso baixará um arquivo "kaggle.json", onde você obterá o seu *username* e uma *key*, algo como:

```
{"username":"seunome","key":"9e9f84b521cd43657418455fce99120b"}
```

Agora, no seu ambiente Colab, copie o conteúdo da célula 1.

```
#https://www.geeksforgeeks.org/how-to-import-kaggle-datasets-directly-into-google-colab/
!pip3 install -q kaggle
!mkdir ~/.kaggle
!echo '{"username":"seunome","key":"9e9f84b521cd43657418455fce99120b"}' > ~/.kaggle/kaggle.json
!chmod 600 ~/.kaggle/kaggle.json
!kaggle datasets download anasmohammedtahir/covidqu
!unzip -u covidqu.zip
```

Célula 1: Célula para baixar o BD da Kaggle. Substitua parte amarela pelas suas informações.

Você deve substituir o conteúdo em amarelo pelas suas informações obtidas de "kaggle.json". A execução dessa célula irá baixar o BD no seu diretório default Colab (/content) e o descompactará.

3. Leitura do BD

Vou propor que vocês criem obrigatoriamente algumas funções, para que não possam apresentar um programa da internet como sua solução do EP.

Crie obrigatoriamente a função:

```
def leImagens(wildcards, classes, nl, nc):
```

que irá ler as imagens especificadas na lista "wildcards" e associar a elas classes especificadas na lista "classes". Por exemplo, a célula 2 deverá ler todas as imagens de treino e armazená-las em matrizes numpy *ax* (imagens) e *ay* (com rótulos 0=covid, 1=não-covid, 2=normal); também lerá todas as imagens/rótulos de validação/teste em matrizes *qx* e *qy*.

```
wildcards=['./Lung Segmentation Data/Lung Segmentation Data/Train/COVID-19/images/*.png', './Lung Segmentation Data/Lung Segmentation Data/Train/Non-COVID/images/*.png', './Lung Segmentation Data/Lung Segmentation Data/Train/Normal/images/*.png']
classes=[0, 1, 2]
ax,ay = leImagens(wildcards, categorias, nl=224, nc=224)
wildcards=['./Lung Segmentation Data/Lung Segmentation Data/Test/COVID-19/images/*.png', './Lung Segmentation Data/Lung Segmentation Data/Test/Non-COVID/images/*.png', './Lung Segmentation Data/Lung Segmentation Data/Test/Normal/images/*.png', './Lung Segmentation Data/Lung Segmentation Data/Val/COVID-19/images/*.png', './Lung Segmentation Data/Lung Segmentation Data/Val/Non-COVID/images/*.png', './Lung Segmentation Data/Lung Segmentation Data/Val/Normal/images/*.png']
classes=[0, 1, 2, 0, 1, 2]
qx,qy=leImagens(wildcards,categorias,nl=224,nc=224)
```

Célula 2: Código para ler todas as imagens de treino em matrizes numpy *ax* e *ay* e todas as imagens de validação/teste em matrizes *qx* e *qy*.

Nota: 33.920 imagens 224×224 em níveis de cinza ocupam 33920 × 224 × 224 = 1.701.969.920 bytes (1,7 GB). Como Colab básico fornece espaço RAM de 12 GB no computador (além de 16 GB de memória no GPU Tesla T4), as imagens cabem tranquilamente na memória do computador (se forem armazenadas com um canal e usando elementos tipo *uint8*). Pode não vai caber se:

1) Armazenar pixels como *float32* ou *float64* ou *int*.
2) Armazenar imagens com 3 bandas (imagens coloridas).
3) Criar várias cópias dos tensores.

Para trabalhar com números em *float32* dentro da rede neural, basta converter somente o lote de imagens atual para *float32* (talvez com 3 bandas) ao alimentar a rede neural ou fazer a conversão de *uint8* para *float32* (talvez com 3 canais) dentro da rede.

4. Classificação manual

Crie obrigatoriamente a função:

```
def mostraLote(x, y=None):
```

que mostra as primeiras 16 imagens do tensor *x* com classificações (se as classificações *y* forem fornecidas, figura 1) ou sem as classificações (se vetor *y* não for fornecido, figura 2). Para mostrar imagens aleatórias, você deve embaralhar *x* e *y* antes de chamar esta função.

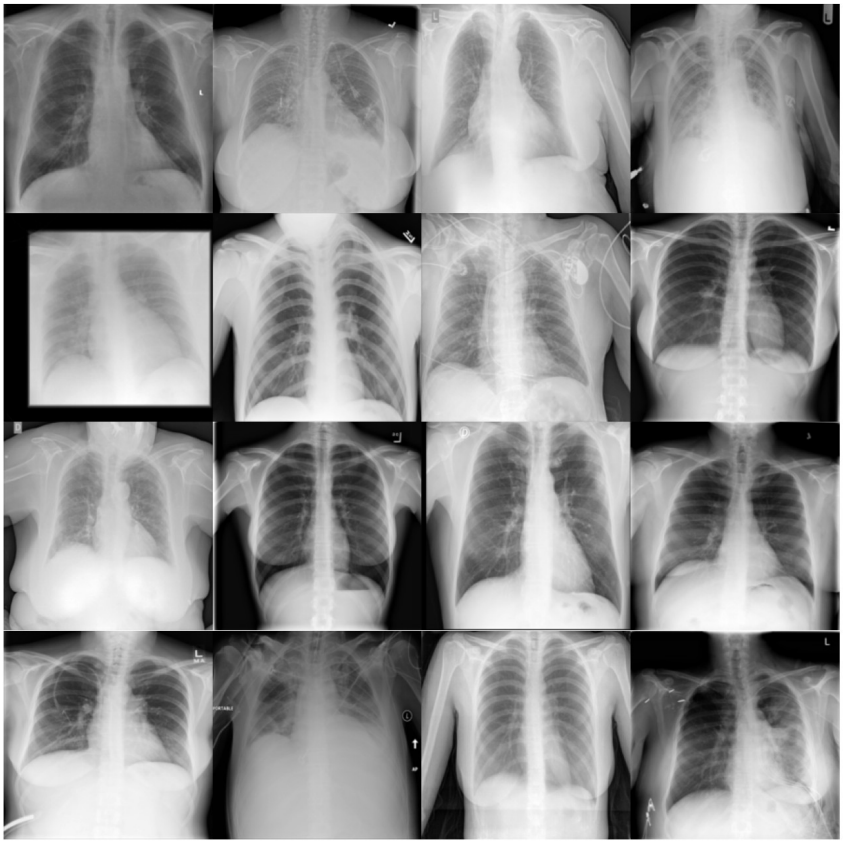

Figura 2: Imagens sem classificações.

Estude como classificar as imagens visualmente. Depois, rode a célula que mostra 16 imagens de validação/teste escolhidas aleatoriamente (sem rótulos como na figura 2) e faça a classificação manual. Compare com as classificações verdadeiras e anote quantas você acertou. Fiz

este teste e acertei 10 das 16 classificações, ou seja, a minha acuracidade foi 62,5%. Informe no seu relatório/vídeo a sua acuracidade.

5. Modelo sem transfer learning

Use um modelo inspirado nas ideias de alguma rede que vimos (LeNet, VGG, Inception, ResNet, EfficientNet, etc.) para classificar as imagens de teste/validação. Você deve treinar rede com pesos inicializados aleatoriamente. Execute o treino durante apenas 16 épocas, usando batch_size=16. Esta limitação é para vocês não gastarem um tempo excessivo no treino. Pode usar qualquer outra técnica para melhorar o desempenho do modelo de classificação, exceto transfer learning. Obtive taxas de acerto de teste/validação de aproximadamente 88,5%. Note que esta rede já é muito melhor do que a minha classificação visual. Informe no relatório/vídeo a técnica escolhida e a acuracidade obtida. Trabalhos com taxas de acerto maiores receberão notas maiores.

6. Modelo com transfer learning

Faça transfer learning de alguma rede pré-treinada (por exemplo, VGG, ResNet, EfficientNet, Xception, etc). Para que o treino não demore excessivamente, vamos estabelecer que pode treinar o modelo somente durante 8 épocas para fazer o ajuste grosso (com modelo-base congelado) e 8 épocas para fazer o ajuste fino (com modelo-base descongelado) usando batch_size=16. Obtive taxa de acerto de teste/validação de aproximadamente 94,4%. Informe no seu relatório/vídeo a técnica escolhida e a acuracidade obtida. Trabalhos com taxas de acerto maiores receberão notas maiores.

Nota: É impressionante como o desempenho da rede neural convolucional, treinada durante poucas épocas, já é muito melhor do que o desempenho da minha classificação visual.

Nota: Relatórios e vídeos melhor elaborados receberão notas maiores.

Obs. 1: Este exercício deve ser resolvido individualmente.

Obs. 2: Em princípio, deve usar o mesmo ambiente de programação utilizado em aula (Python, Keras, OpenCV). Se quiser utilizar outro ambiente, converse antes com o professor.

Obs. 3: Entregue um documento PDF de no máximo 4 páginas (relatorio.pdf) descrevendo o funcionamento dos seus programas, os resultados obtidos e as suas conclusões. O envio do relatório é obrigatório (veja o anexo B). O relatório é um documento Word/Latex/Writer convertido para PDF. Um notebook Python anotado não será aceito como relatório.

Obs. 4: Entregue um vídeo de no máximo dois minutos explicando o funcionamento dos seus programas e os resultados obtidos. No vídeo deve aparecer em algum momento o seu rosto e um documento seu. É necessário que o vídeo contenha áudio, pois é muito difícil entender um vídeo sem a explicação falada. O vídeo não pode ter sido acelerado para diminuir a duração, pois dificulta entender a fala. Você pode enviar o vídeo em si (.mkv, .mp4, .avi, etc.) ou um link para o vídeo (youtube, google drive, etc). No segundo caso, deve assegurar que todos os professores (Hae e Maurício - hae.kim@usp.br, mlisboa@usp.br) tenham acesso ao seu vídeo.

Obs. 5: Entregue os programas-fontes ou link para notebook Google Colab. Você deve assegurar que todos os professores tenham acesso a esse material. Não entregue o BD.

Obs. 6: Não se esqueça de escrever o seu nome no relatório, no início do vídeo e no início dos programas-fontes.

Obs. 7: Envie o material através de edisciplinas. Dentro do prazo, você pode substituir o material anterior por um novo.

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 curtas e simples. Escreva usando menos palavras possível. Use o tempo verbal o mais simples possível. Evite "verbos ocultos" (por exemplo, substitua "precisamos realizar uma revisão das contas da Agência" para "precisamos rever as contas da Agência"; substitua "fiz o pagamento do seu salário" por "paguei o seu salário"). 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 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

1 https://plainlanguage.gov/guidelines/

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 imagens 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? O problema foi resolvido de forma satisfatória?

Referências

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

Solução privada em:
https://colab.research.google.com/drive/1Buav7ANM7_RYYowO7rnIQ1jhosWfNMwx