Digital Image Processing

Assignment 5

SAI KUMAR MURARSHETTI
LEWIS ID: L30079224

```
import tensorflow as tf
import numpy as np
from matplotlib import pyplot as plt
import sys, time, imageio, h5py, skimage, glob, os, shutil
```

```
# Download the pre-trained model
# overwrite anyway
if os.path.isdir('upload_images'):
    shutil.rmtree('upload_images')
os.mkdir('upload_images') # to save temp output
!wget -O model/TomoGAN.h5 https://raw.githubusercontent.com/AIScienceTutorial/Denoising/main/model/TomoGAN.h5
```

```
--2024-02-21 19:48:23--  https://raw.githubusercontent.com/AIScienceTutorial/Denoising/main/model/TomoGAN.h5
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.109.133, 185.199.111.133, 185.199.108.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.109.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2854872 (2.7M) [application/octet-stream]
Saving to: ‘model/TomoGAN.h5’

model/TomoGAN.h5    100%[===================>]   2.72M  --.-KB/s    in 0.06s

2024-02-21 19:48:23 (42.1 MB/s) - ‘model/TomoGAN.h5’ saved [2854872/2854872]
```

```
# overwrite anyway
if os.path.isdir('model'):
    shutil.rmtree('model')
os.mkdir('model') # to save temp output

!wget -O model/TomoGAN.h5 https://raw.githubusercontent.com/AIScienceTutorial/Denoising/main/model/TomoGAN.h5

# Load the pre-trained model

TomoGAN_mdl = tf.keras.models.load_model('model/TomoGAN.h5')
TomoGAN_mdl.summary()
```

```
--2024-02-21 19:48:23--  https://raw.githubusercontent.com/AIScienceTutorial/Denoising/main/model/TomoGAN.h5
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2854872 (2.7M) [application/octet-stream]
Saving to: ‘model/TomoGAN.h5’

model/TomoGAN.h5    100%[===================>]   2.72M  --.-KB/s    in 0.07s

2024-02-21 19:48:23 (40.7 MB/s) - ‘model/TomoGAN.h5’ saved [2854872/2854872]

WARNING:tensorflow:No training configuration found in the save file, so the model was *not* compiled. Compile it manually.
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   Connected to
=================================================================
 input_1 (InputLayer)        [(None, None, None, 1)]   0         []

 conv2d (Conv2D)             (None, None, None, 8)     16        ['input_1[0][0]']

 conv2d_1 (Conv2D)           (None, None, None, 32)    2336      ['conv2d[0][0]']

 conv2d_2 (Conv2D)           (None, None, None, 32)    9248      ['conv2d_1[0][0]']

 max_pooling2d (MaxPooling2  (None, None, None, 32)    0         ['conv2d_2[0][0]']
 D)

 conv2d_3 (Conv2D)           (None, None, None, 64)    18496     ['max_pooling2d[0][0]']

 conv2d_4 (Conv2D)           (None, None, None, 64)    36928     ['conv2d_3[0][0]']

 max_pooling2d_1 (MaxPoolin  (None, None, None, 64)    0         ['conv2d_4[0][0]']
 g2D)

 conv2d_5 (Conv2D)           (None, None, None, 128)   73856     ['max_pooling2d_1[0][0]']

 conv2d_6 (Conv2D)           (None, None, None, 128)   147584    ['conv2d_5[0][0]']
```

```
max_pooling2d_2 (MaxPoolin  (None, None, None, 128)      0         ['conv2d_6[0][0]']
g2D)

conv2d_7 (Conv2D)           (None, None, None, 128)      147584    ['max_pooling2d_2[0][0]']

up_sampling2d (UpSampling2  (None, None, None, 128)      0         ['conv2d_7[0][0]']
D)

concatenate (Concatenate)   (None, None, None, 256)      0         ['conv2d_6[0][0]',
                                                                    'up_sampling2d[0][0]']

conv2d_8 (Conv2D)           (None, None, None, 64)       147520    ['concatenate[0][0]']

conv2d_9 (Conv2D)           (None, None, None, 64)       36928     ['conv2d_8[0][0]']

up_sampling2d_1 (UpSamplin  (None, None, None, 64)       0         ['conv2d_9[0][0]']
g2D)

concatenate_1 (Concatenate  (None, None, None, 128)      0         ['conv2d_4[0][0]',
)                                                                   'up_sampling2d_1[0][0]']
```

```
# Import necessary libraries
import matplotlib.pyplot as plt
import numpy as np
import sys
from skimage.transform import resize

# Load the noisy sample image
sample1_img = plt.imread('download 1.png')

# Check the dimensions of the image and convert to grayscale if needed
if sample1_img.ndim == 3:
    # If RGB, convert to grayscale by taking the mean across color channels
    sample1_img_gray = np.mean(sample1_img, axis=2, keepdims=True)
elif sample1_img.ndim == 2:
    # If already grayscale, just add a channel dimension
    sample1_img_gray = sample1_img[..., np.newaxis]
else:
    # Unsupported image format
    print("Unsupported image format. Please provide a grayscale or RGB image.")
    sys.exit(1)

# Set the expected input shape of the model
input_height, input_width = 256, 256

# Resize the input image to match the expected input shape of the model
resized_img = resize(sample1_img_gray, (input_height, input_width))

# Normalize the image data assuming input range is [0, 255]
normalized_img = resized_img / 255.0

# Predict using the pre-trained model (assumed to be named 'TomoGAN_mdl')
dn_img_sample1 = TomoGAN_mdl.predict(np.expand_dims(normalized_img, axis=0)).squeeze()

# Plotting
plt.figure(figsize=(15, 5))

# Plot the noisy/input image
plt.subplot(131)
plt.imshow(sample1_img_gray.squeeze(), cmap='gray')
plt.title('Noisy/Input (download 1.png)', fontsize=18)

# Load clean label image for download 1
clean_label_img = plt.imread('download 1.png')

# Convert the clean label image to grayscale if needed
if clean_label_img.ndim == 3:
    clean_label_img_gray = np.mean(clean_label_img, axis=2, keepdims=True)
elif clean_label_img.ndim == 2:
    clean_label_img_gray = clean_label_img[..., np.newaxis]
else:
    print("Unsupported image format. Please provide a grayscale or RGB image.")
    sys.exit(1)

# Plot the clean/label image
plt.subplot(132)
plt.imshow(clean_label_img_gray.squeeze(), cmap='gray')
plt.title('Clean/Label (download 1.png)', fontsize=18)

# Plot the denoised image
plt.subplot(133)
```

```
plt.imshow(dn_img_sample1.squeeze(), cmap='gray')
plt.title('Denoised (download 1.png)', fontsize=18)

# Adjust layout
plt.tight_layout()

# Display the plot
plt.show()
```

```
# Denoise sample1.png and display the result
sample1_img = plt.imread('download 1.png')
```

```
if sample1_img.ndim == 3:
    sample1_img_gray = np.mean(sample1_img, axis=2, keepdims=True)
elif sample1_img.ndim == 2:
    sample1_img_gray = sample1_img[..., np.newaxis]
else:
    print("Unsupported image format. Please provide a grayscale or RGB image.")
    sys.exit(1)
```

```
# Find out the expected input shape of the model
input_height, input_width, _ = TomoGAN_mdl.input_shape[1:]
```

```
# Resize the input image to match the expected input shape of the model
input_height, input_width = 256, 256
```

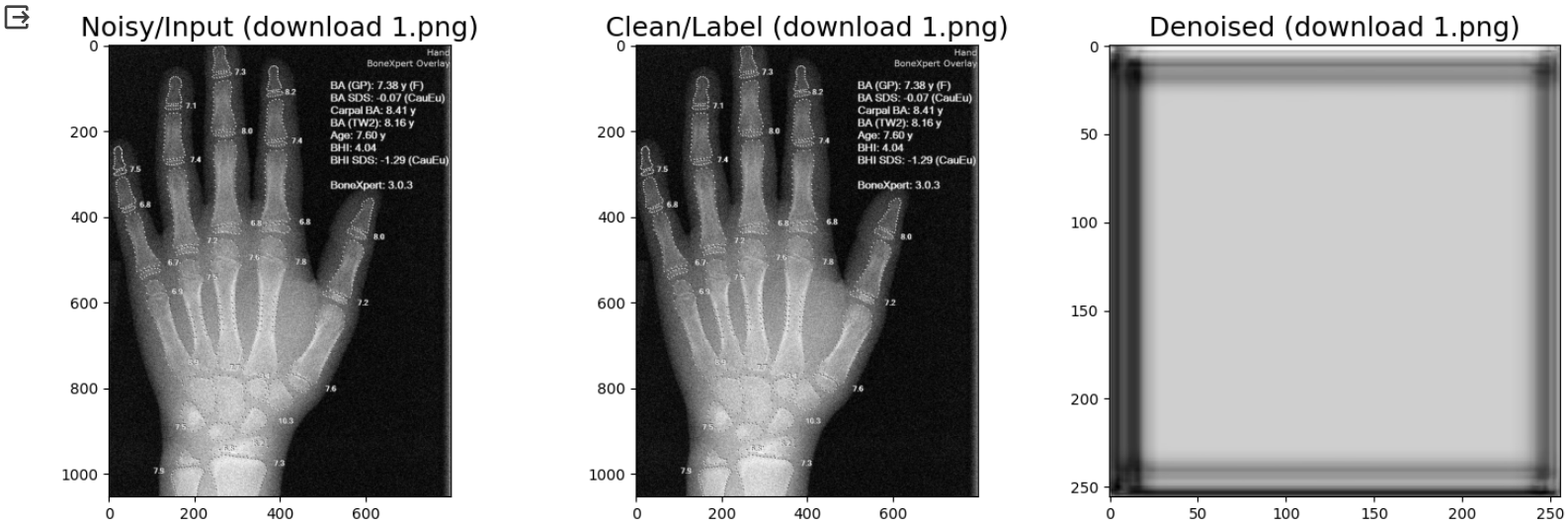
```
# Resize the input image to match the expected input shape of the model
from skimage.transform import resize

resized_img = resize(sample1_img_gray, (input_height, input_width))
```

```
# Normalize the image data if necessary
normalized_img = resized_img / 255.0  # Assuming input range is [0, 255]
```

```
# Predict using the pre-trained model
dn_img_sample1 = TomoGAN_mdl.predict(np.expand_dims(normalized_img, axis=0)).squeeze()
```

```
     1/1 [==============================] - 0s 462ms/step
```

```
plt.figure(figsize=(15, 5))
plt.subplot(131)
plt.imshow(sample1_img_gray.squeeze(), cmap='gray')
plt.title('Noisy/Input (download 1.png)', fontsize=18)

# Load clean label image for download 1
clean_label_img = plt.imread('download 1.png')

if clean_label_img.ndim == 3:
    clean_label_img_gray = np.mean(clean_label_img, axis=2, keepdims=True)
elif clean_label_img.ndim == 2:
    clean_label_img_gray = clean_label_img[..., np.newaxis]
else:
    print("Unsupported image format. Please provide a grayscale or RGB image.")
    sys.exit(1)

plt.subplot(132)
plt.imshow(clean_label_img_gray.squeeze(), cmap='gray')
plt.title('Clean/Label (download 1.png)', fontsize=18)

plt.subplot(133)
plt.imshow(dn_img_sample1.squeeze(), cmap='gray')
plt.title('Denoised (download 1.png)', fontsize=18)

plt.tight_layout()
plt.show()
```


Noisy/Input (download 1.png)
Clean/Label (download 1.png)
Denoised (download 1.png)
Hand
BoneXpert Overlay
BA (GP): 7.38 y (F)
BA SDS: -0.07 (CauEu)
Carpal BA: 8.41 y
BA (TW2): 8.16 y
Age: 7.60 y
BHI: 4.04
BHI SDS: -1.29 (CauEu)
BoneXpert: 3.0.3