San José State University
Computer Science Department
CS156, Introduction to Artificial Intelligence, Spring 2021

Homework #10

Objective:

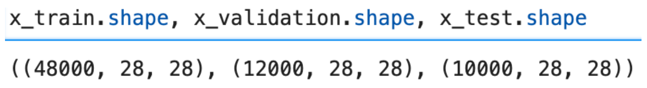
This homework's objective is to implement autoencoders to work with fashion images.

Details:

For this assignment you will be working with *fashion_mnist* dataset available in *keras*. This dataset provides 2-D black and white images of fashion items (clothing, shoes, etc.). You can load images in this dataset with the following python code:

(x_train_valid, y_train_valid), (x_test, y_test) = keras.datasets.fashion_mnist.load_data()

Use scikit learn *train_test_split()* function to split x_train_valid data (in the code above) into training and validation sets. You will have training, validation, and test sets of the following sizes:



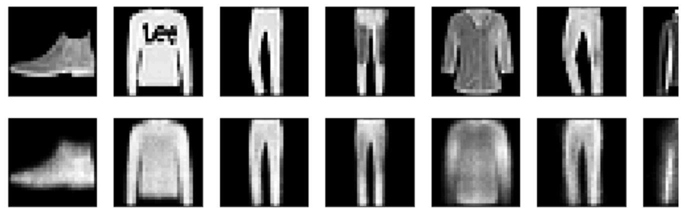
```
x_train.shape, x_validation.shape, x_test.shape

((48000, 28, 28), (12000, 28, 28), (10000, 28, 28))
```

For this assignment you can utilize a lot of code snippets from the *Autoencoders.MNIST.ipynb* Jupyter notebook demonstrated in class.

<u>For this assignment you need to submit implementations for 2 tasks (in a single notebook):</u>

1. Create an autoencoder model for encoding and decoding images and demonstrate its performance by showing original and reconstructed images for the first 10 images in your test set (original on top and reconstructed on the bottom). The code examples in *Autoencoders.MNIST.ipynb* Jupyter notebook show how to plot images in this way:



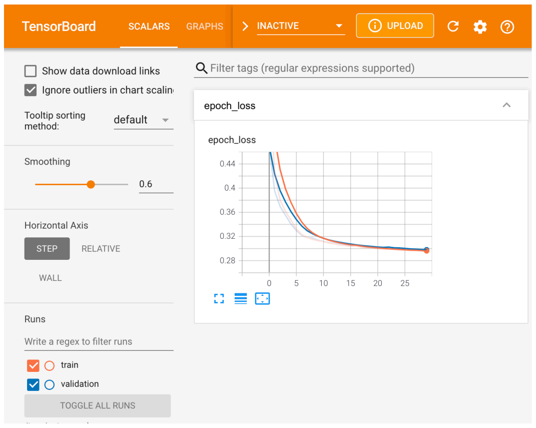
…

For this task your image data should be flattened to a single dense input vector of size 784. Your model should have 3 layers (128, 64, 32) for the encoder and 3 layers (64, 128, 784) for the decoder. See the following summary of the model you have to create:

```
Layer (type)                 Output Shape              Param #
=================================================================
input_63 (InputLayer)        [(None, 784)]             0
_________________________________________________________________
dense_205 (Dense)            (None, 128)               100480
_________________________________________________________________
dense_206 (Dense)            (None, 64)                8256
_________________________________________________________________
dense_207 (Dense)            (None, 32)                2080
_________________________________________________________________
dense_208 (Dense)            (None, 64)                2112
_________________________________________________________________
dense_209 (Dense)            (None, 128)               8320
_________________________________________________________________
dense_210 (Dense)            (None, 784)               101136
=================================================================
```

For visualizing the loss when training this model you need to utilize tensorboard and display this visualization with:
%load_ext tensorboard
%tensorboard --logdir=/tmp/autoencoder

The above should bring up this view:



☐ Show data download links
☑ Ignore outliers in chart scalin
Tooltip sorting method: default
Smoothing 0.6
Horizontal Axis: STEP
Runs
☑ train
☑ validation

Don’t forget to use *tensorflow.keras.callbacks.TensorBoard()* function and specify the *callbacks* parameter in *fit()* function. Again, the *Autoencoders.MNIST.ipynb* notebook has examples of how to do this.

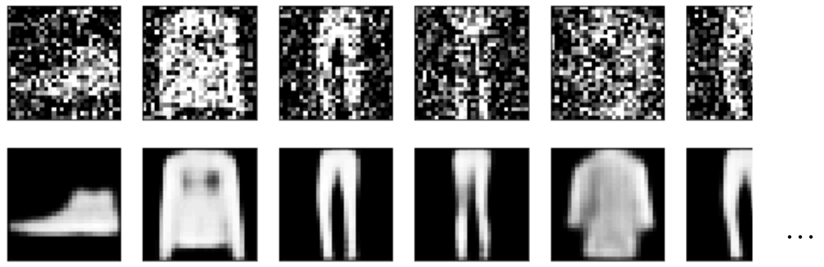
2. Use an autoencoder model to denoise noisy images. You will use the same fashion mnist images as in part 1 but will add some noise to them, then will denoise the images using an autoencoder model. First, you will need to reshape your images to 2-D from the flattened shape. For example, this will work for the training dataset named *x_train*:

   x_train.reshape(-1, 28, 28, 1)

Add some noise to all three sets of images (training, validation, and test) in a similar way that *Autoencoders.MNIST.ipynb* notebook shows, using *noise_factor = 0.4*. Train a model with the following architecture (2 convolutional layers with max pooling in between for the encoder, with mirror architecture for the decoder):

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_67 (InputLayer)        [(None, 28, 28, 1)]       0         
_________________________________________________________________
conv2d_12 (Conv2D)           (None, 28, 28, 32)        320       
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 14, 14, 32)        0         
_________________________________________________________________
conv2d_13 (Conv2D)           (None, 14, 14, 32)        9248      
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 7, 7, 32)          0         
_________________________________________________________________
conv2d_14 (Conv2D)           (None, 7, 7, 32)          9248      
_________________________________________________________________
up_sampling2d_4 (UpSampling2 (None, 14, 14, 32)        0         
_________________________________________________________________
conv2d_15 (Conv2D)           (None, 14, 14, 32)        9248      
_________________________________________________________________
up_sampling2d_5 (UpSampling2 (None, 28, 28, 32)        0         
_________________________________________________________________
conv2d_16 (Conv2D)           (None, 28, 28, 1)         289       
=================================================================
```

Finally, denoise and plot the first 10 images of the test set, in a manner similar to part 1 (noisy image on top, its counterpart denoised image below):



You can utilize any code examples shown in the following Jupyter notebooks:
- Autoencoders.MNIST.ipynb
- Variational_autoencoder.MNIST.ipynb

Submission:

Email your assignment submission to me at Yulia.Newton@sjsu.edu and the grader (Akshay Kajale) at akshay.kajale@sjsu.edu. Make sure to email this submission by 11:59pm on the due date listed in Canvas. Your sent email is the proof of submission. The subject of the email should say “CS156 Assignment 10”. In the body of the email list your name as it appears on the class roster and your student ID. Attach to this email both the pdf of your Jupyter notebook, which contains the solution for this homework assignment, as well as the notebook itself (the notebook file with .ipynb extension). Make sure to submit both files, otherwise the submission will not be considered complete.

Grading:

I will return the grades as fast as we can grade this homework. Normally it should not take more than a few weeks.

A total of 10 points are possible for this homework assignment.