TensorFlow Lab

Checkpoint 1: Verify Your TensorFlow

For the first checkpoint of this lab, we'll work on just getting a proper installation of the tools we will need to use.

My recommendation is to use a docker container. Feel free to use the tensorflow container I used during lecture and follow the instructions in the lecture to provision it. Alternately, I have created a Dockerfile here that you can use to create a fully provisioned (for our purposes) docker image or you are welcome to run the image (wdturner/tensorflow_oss:latest) that I put on Docker Hub.

If you use any of these docker images, I recommend that you run it by binding it to a local directory. That way your work will remain once you kill the container. You can do this by running the following commands:

```
mkdir lab11
docker run -it -p 8888:8888 -v "$PWD"/lab11:/home/tensorflow -e "DISPLAY"=host.docker.interr
   wdturner/tensorflow_oss:latest
```

When you exit the container, all of your work will remain in the lab11 directory.

That said, tensorflow can be installed any number of ways and I am going to give you leeway to use whatever tools you want for this lab. If you are having problems with docker (or just want the experience), feel free to use a native install to your local version of python.

Regardless, you will need access to the python packages **tensorflow, keras, matplotlib and numpy, and probably pillow**. Install them (you can probably use pip or conda depending on your `python` install). Once you are done, start `python` and run:

Lastly, if none of these options work, feel free to use a google colab setup. This link here is to a Google Colab notebook. It's read only, so duplicate it, make a copy and run from there. You need to connect before running. ***Note that if you use colab, you will not be able to connect to it to check your work during the exam.***

Once you have your prefered setup provisioned and running, try running the following code using python.

```
from __future__ import absolute_import, division, print_function

# TensorFlow and tf.keras
import tensorflow as tf
from tensorflow import keras
from PIL import Image
from PIL import ImageOps
```

```
# Helper libraries
import numpy as np
import matplotlib.pyplot as plt

print(tf.__version__)
plt.plot((-2,4),(-6,6))
plt.show()
```

This should print out a version of TensorFlow in the console window and pop up a window:

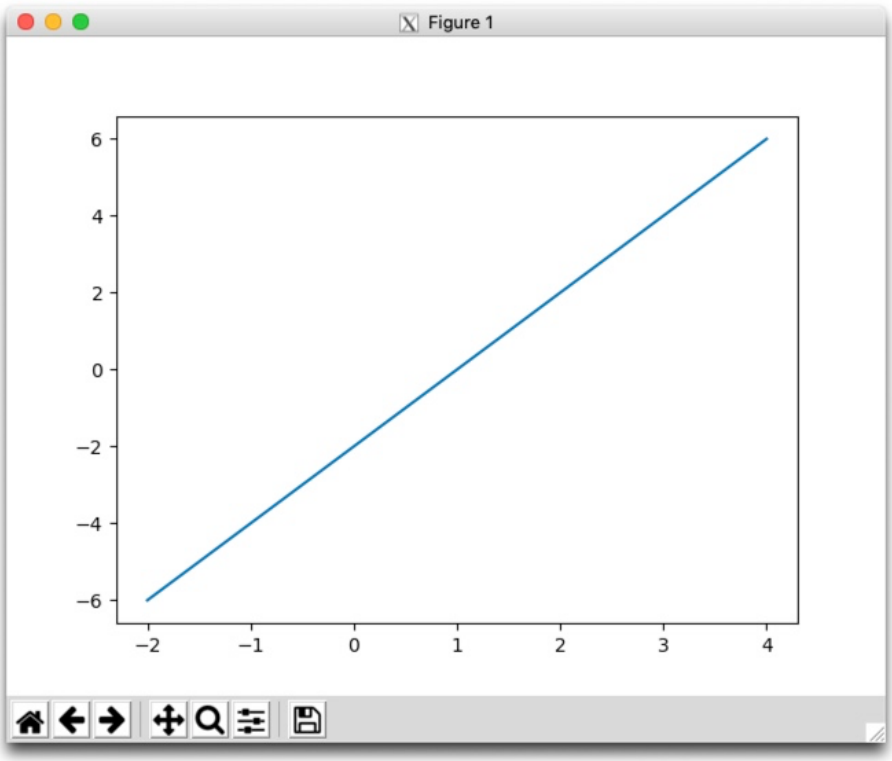


Figure 1: hello_tf.png

Take a screenshot of your pop-up and include it in your lab report.

When you are finished exit python and restart it to clear your work.

Checkpoint 2: Run a TensorFlow classification

Next go to https://www.tensorflow.org/tutorials/keras/classification and run through the fashion classification example in its entirety.

When you are done, modify the code block below to print out the information for the 15 images from 9000-9014 instead of for images 0-11:

```
# Plot the first X test images, their predicted label, and the true label
# Color correct predictions in blue, incorrect predictions in red
num_rows = 5
num_cols = 3
num_images = num_rows*num_cols
plt.figure(figsize=(2*2*num_cols, 2*num_rows))
for i in range(num_images):
  plt.subplot(num_rows, 2*num_cols, 2*i+1)
  plot_image(i, predictions, test_labels, test_images)
  plt.subplot(num_rows, 2*num_cols, 2*i+2)
  plot_value_array(i, predictions, test_labels)
plt.show()
```

Grab a screenshot of this plot and include it in your lab report.

Checkpoint 3: Curate some data

Browse the web, or take a picture to find at least 3 images of clothing, shoes, bags etc. The Google speakers assured us that obtaining and curating the data would be the hard part. We are going to see if they were correct. Take your images, process them to get them in the correct form for the model:

- greyscale
- 28x28 pixels - make sure you use cropping to get your image square before resizing it
- inverted (white is 0)
- scaled between 0 an 1 instead of 0 and 255

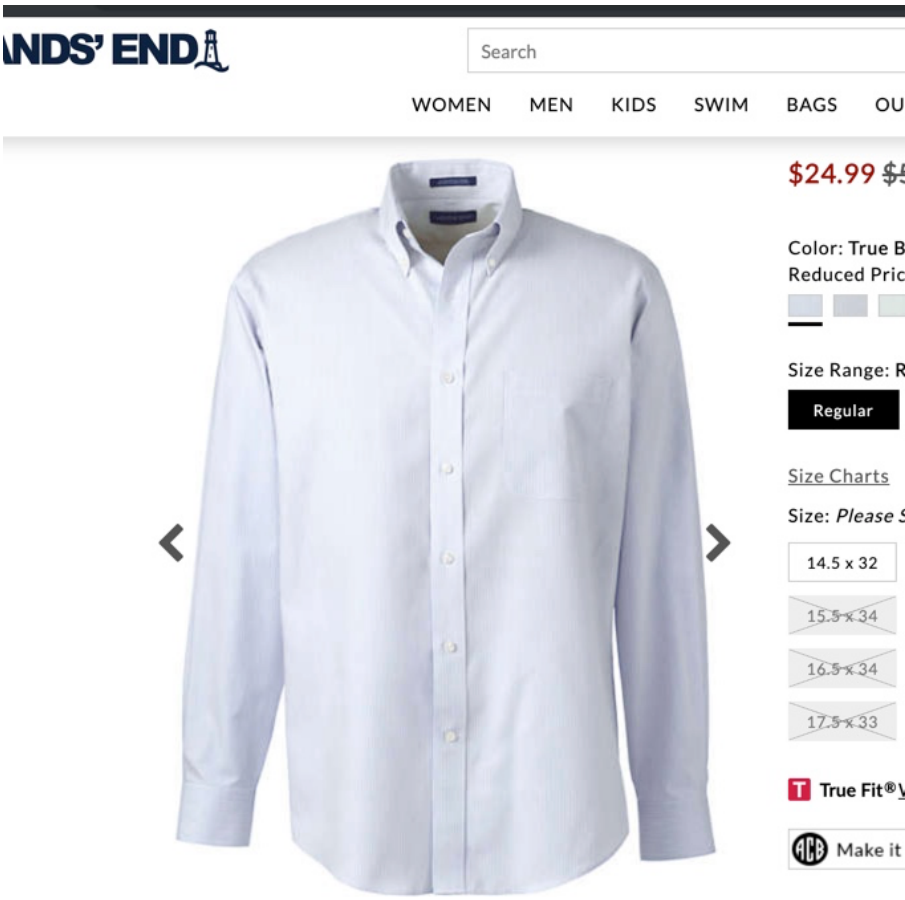
Then pass them into the classification model you generated. The Image and ImageOps modules from pillow will be very useful for this. It will help to view the images using the same tools you used to view the test and training images for the models just to be sure you have good models.

The following links may help you:

https://pillow.readthedocs.io/en/stable/

https://kite.com/python/examples/4887/pil-convert-between-a-pil-%60image%60-and-a-numpy-%60array%60

Include your original image, your greyscale 28x28 pixel image, and the results of your classification in your lab report.



Here are my first three attempts:

Figure 2: small_shirt1.png

```
[[1.4762013e-02 2.2030171e-04 5.1683296e-02 2.0069817e-02 6.9348738e-02
  1.3521332e-02 7.6108474e-01 1.0612813e-03 6.6945553e-02 1.3029455e-03]] 6 Shirt

[[3.5796508e-03 1.3106716e-05 3.2439572e-01 2.3899054e-02 4.4865412e-01
  8.4455935e-03 1.8258789e-01 6.8158122e-05 8.3385631e-03 1.8128438e-05]] 4 Coat

[[2.5140275e-06 3.3653549e-11 1.4402343e-04 5.7210900e-06 2.7189176e-03
  5.5948669e-07 9.9712771e-01 2.7641710e-11 4.2381637e-07 1.2968426e-09]] 6 Shirt
```

2 out of 3 isn't bad ... You are welcome to use my small images as part of your testing, although, I still expect you to get novel images as well. If you do use my images, note that they are already 28x28 and greyscale, but they have not

Figure 3: shirt2.png

Figure 4: small_shirt2.png

Figure 5: shirt3.png

Figure 6: small_shirt3.png

yet been inverted and scaled to [0, 1]. You will need to complete the operations to get good results.

When you are finished, push your report to your lab report and submit a link to your repository in Submitty.