Welcome everyone!

Let’s talk first about the expected outcomes of this project. It should be noted that this is supposed to serve as the warm up project with the TX2 and embedded workflows.

Student Outcomes:

- Using Embedded Workflow
- Inference vs Accuracy
- Data Collection
- Proper Documentation
- Portfolio Ready Project

A lot of the teaching materials are still in progress but I am releasing to you guys because well to be honest I know that you will be able to take on the challenge :)

So let’s talk about each of the student outcomes and how it will tie in with the project.

- Using the embedded workflow:
    a) You will be using Nvidia DIGITS 6.0 on a aws AMI in order to take data you collect and produce a network capable of classification or detection if you so wish.
    b) In fact you could use both, one network to detect the objects and a second to classify. Like so:
        Here's a fun application:
            https://github.com/S4WRXTTCS/jetson-inference

    - You will also be provided a data set in the ami that after training a network on it you will run the command `evaluate` to see the networks inference speed and accuracy. This part of the project is to simple make sure that there is a little bit of a challenge.

- Inference vs Accuracy:
    - A big point to drive away in this project is the importance of inference time vs accuracy. Thinking on the cases in which one aspect may be more important than another. Also how you may only be able to choose one in the real world so which to choose.

- Data Collection:
    - You will be collecting your own data whether from a webcam, raspi cam, tx2 camera or even splicing a video apart.
    - You will need to decided what data will be best for your idea.

- Proper Documentation:

    - In the end you will need to write up your finding. This is important because no matter how cool something you create it is. Without proper documentation you may lose over half your audience.
    - We have laid out the points to be addressed on the rubric.

- Portfolio Ready Project
    - Having a tangible project to show employers. Being able to rapidly prototype an idea to speed up efficiency in the workplace. The Kaizen way!

So here are the project steps:

0) Use Digits workflow on provided data to achieve maximum accuracy and minimum inference time.. Run auto grade script `evaluate` when ready to test. Data is at home directory level and just needs to be brought into digits. Use the pseudo workflow below to guide you.

- For personal portion.

Pick an idea for a real time network you would like to use. Here are some ideas that I think would be useful.

- Real time pill classification
- Detection net to find items and then classify the items to make a decision
- Classify non defective item vs defective item.
- Any sort of classification or detection that you might actuate from with a robotic system.

- Here is a preview of what I have been cooking up:
    - It's the data you will be working with for the test set. Its a conveyer system that separates bottles from candy boxes. Lame I know. Working on other ideas but the infrastructure is there and it runs in real time!

    - Some pictures: https://imgur.com/a/6OpdI
    - A video of testing: https://drive.google.com/open?id=1dcjXc0Ut5e6mHqmf2XLbsslFoNUvTKRN
    - Final video coming after I redo the wiring harness for the pwm driver :P

1) Gather your own data
    - Use script (no helper code for student)
    - Use any libraries desired
    - Python or c++ (If you want a challenge)
    - All in png format
    - Color (bigger files + longer training) or Gray (smaller files + shorter training)
    - Collect as many photos as you think necessary to train a good network.

- Use webcam, tx2, other camera, or video splicing.

Create final folder structure like so:

```
Office_Supplies/
├──── Stapler/
│    ├──── Stapler_1.png
│    └──── Stapler_2.png
├──── Paper/
│    ├──── Paper_1.png
│    └──── Paper_2.png
└──── Scissors/
    ├──── Scissors_1.png
    └──── Scissors_2.png
```

More info here:
https://github.com/NVIDIA/DIGITS/blob/master/docs/ImageFolderFormat.md

When done compress your data.

2) Get Udacity Digit AMI called: Udacity-Nvidia DIGITS Server
- Choose one of the available regions: Oregon, North Virginia, Frankfurt, South America, Tokyo, Ohio, Seoul and Sydney
- Then click **Continue**
- Get **p2.xlarge** instance
- Add tcp security rule, open port 5000 **everywhere**
- Use a key pair you made in Term 1 or create one.

3) ssh into AWS instance
- ssh -i "key_name.pem" ubuntu@ec2-34-215-113-119.us-west-2.compute.amazonaws.com
- enter digits command

4) scp your files into your instance into home directory

```
scp -i "key_name.pem" /Users/KyleF/Desktop/Data.tar.gz
ubuntu@ec2-35-166-126-17.us-west-2.compute.amazonaws.com:/home/ubuntu
```

5) Enter ` digits` command into a connected ssh terminal to start the digits server. Enter **AMI public ip** and port 5000 like so:
- 52.37.132.96:5000

6) Create new data set (from 2DTD instructions)

More info here:

https://github.com/dusty-nv/jetson-inference/blob/master/README.md#importing-classification-dataset-into-digits
7) Create new model (from 2DTD)

More info here:

https://github.com/dusty-nv/jetson-inference/blob/master/README.md#creating-image-classification-model-with-digits

8) Observe test accuracy of your model and check inference test for the supplied data only not yours (for now).

For TX2 Students:

9) Download your model to your tx2

10) Create a folder with model name and extract your downloaded contents into the folder with `tar -xzvf`

11) Create NET variable and set it equal to the model path

12) Launch imagenet with custom variables like so:

```
        ./imagenet-camera --prototxt=$NET/deploy.prototxt --model=$NET/your_name.caffemodel
--labels=$NET/labels.txt --input_blob=data     --output_blob=softmax
```

13) Observe real time results.

14) Actuate if desired :) (More info to come. GPIO, I2C, Serial, ect)

15) Also explain the usage of TensorRT to increase inference time

----

I know that some items may be less clear than others. I will do best to update and be more clear and also to help you all individually. Thanks again for all your help and happy learning. Feel free to comment on the doc as well. :)