JetsonNX/Jetson Nano TensorRT Donkey Setup

1. Converting a .h5 file to a tensorrt directory

a. Check that your Jetson or Docker Container has access to the GPUs

```
python
>>import tensorflow as tf
>>print("Available GPUs:", tf.config.list_physical_devices('GPU'))
```

If there is no GPU access, TensorRT can not be used. If the Docker can not access the GPU, you may need to make another container with the correct GPU flag in the bash script.

b. Once we know that there is GPU access we can continue to convert our .h5 model to a .savedmodel by running this code:

```
python
>>import tensorflow as tf
>>h5_model_path = “models/yourmodel.h5”
>>saved_model_path = “models/yourmodel_converted.savedmodel”
>>model = tf.keras.models.load_model(h5_model_path, compile=False)
>>model.save(saved_model_path)
>>exit()
```

Now we have a .savedmodel file.

2. We now convert this to a TensorRT Directory

c. Convert the .savedmodel file to a TensorRT directory by running this code:

<u>FOR PYTHON 3.8</u>

```
python
>>from tensorflow.python.compiler.tensorrt import trt_convert as trt
>>import os
>>saved_model_path = “models/yourmodel_converted.savedmodel”
>>tensorrt_model_path = “models/yourmodel”

>>os.makedirs(tensorrt_model_path, exist_ok=True)

>>converter = trt.TrtGraphConverterV2(
        input_saved_model_dir=saved_model_path,
        precision_mode=trt.TrtPrecisionMode.FP16
)

>>converter.convert()
>>converter.save(tensorrt_model_path)
```

FOR PYTHON 3.6

```
python
>>from tensorflow.python.compiler.tensorrt import trt_convert as trt
>>import os
>>saved_model_path = “models/yourmodel_converted.savedmodel”
>>tensorrt_model_path = “models/yourmodel”

>>os.makedirs(tensorrt_model_path, exist_ok=True)

>>conversion_params = trt.ConversionParams(precision_mode="FP16")

>>converter = trt.TrtGraphConverterV2(
   input_saved_model_dir=saved_model_path,
   conversion_params=conversion_params
)

>>converter.convert()
>>converter.save(tensorrt_model_path)
```

Now we have a tensorrt directory for Python 3.6 & 3.8 which we can run and test!

TROUBLESHOOTING:

When trying to use TensorRT models made in different versions of Python with different dependencies, there were errors running the model using manage.py. This error came from not being able to load the savedmodel path

```
OSError: SavedModel file does not exist at:
~/projects/d4/models/yourmodel_tensorrt/{saved_model.pbtxt|saved_model.pb}
```

To check whether your saved_model.pb will load run this code:

```
>>import tensorflow as tf
>>model = tf.saved_model.load("/home/jetson/projects/d4/models/suarez_tensorrt")
>>print(model.signatures)
```

If this fails, the model might not be correctly converted.

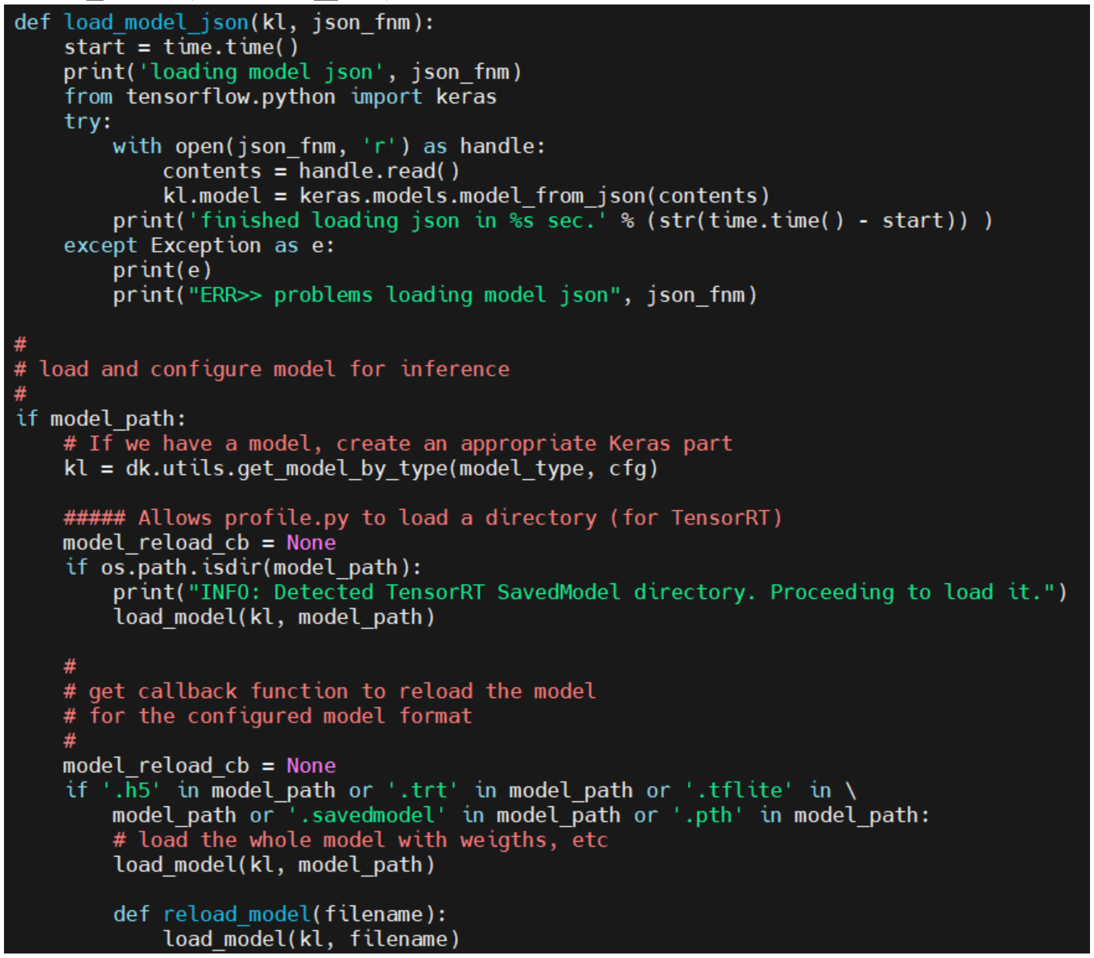
2. Benchmarking w/ TensorRT

To use profile.py with a directory, we must change how our manage.py loads a model.

In manage.py we change the load_model_json function and add under model_reload_cb = None:

model_reload_cb = None
###
if os.path.isdir(model_path):
   print("INFO: Detected TensorRT SavedModel directory. Proceeding to load it.")
   load_model(kl, model_path)
###

elif '.h5' in model_path or '.trt' in model_path or '.tflite' in \ model_path or '.savedmodel' in model_path or '.pth' in model_path:
   load_model(kl, model_path)



```python
def load_model_json(kl, json_fnm):
    start = time.time()
    print('loading model json', json_fnm)
    from tensorflow.python import keras
    try:
        with open(json_fnm, 'r') as handle:
            contents = handle.read()
            kl.model = keras.models.model_from_json(contents)
        print('finished loading json in %s sec.' % (str(time.time() - start)) )
    except Exception as e:
        print(e)
        print("ERR>> problems loading model json", json_fnm)

#
# load and configure model for inference
#
if model_path:
    # If we have a model, create an appropriate Keras part
    kl = dk.utils.get_model_by_type(model_type, cfg)

    ##### Allows profile.py to load a directory (for TensorRT)
    model_reload_cb = None
    if os.path.isdir(model_path):
        print("INFO: Detected TensorRT SavedModel directory. Proceeding to load it.")
        load_model(kl, model_path)

    #
    # get callback function to reload the model
    # for the configured model format
    #
    model_reload_cb = None
    if '.h5' in model_path or '.trt' in model_path or '.tflite' in \
        model_path or '.savedmodel' in model_path or '.pth' in model_path:
        # load the whole model with weigths, etc
        load_model(kl, model_path)

        def reload_model(filename):
            load_model(kl, filename)
```

Once this manage.py file is saved, we can run profile.py using this command line:
python profile.py --model=./models/yourmodel_tensorrt --type=tensorrt_linear

Drive Model:

python manage.py drive --model=./models/yourmodel_tensorrt --type=tensorrt_linear