fire-detection-jetson-save-ca

February 16, 2018

```
In [1]: from sklearn.datasets import load_files
        from keras.utils import np_utils
        import numpy as np
        from glob import glob
        import cv2
        import matplotlib.pyplot as plt
        #from keras.applications.resnet50 import preprocess_input, decode_predictions
        #from keras.applications.xception import Xception, preprocess_input
        #from keras.applications.inception_v3 import InceptionV3, preprocess_input
        #from keras.applications.vgg16 import VGG16, preprocess_input
        from keras.applications.vgg19 import VGG19, preprocess_input
```

```
Using TensorFlow backend.
```

0.0.1 Display nodes

```
In [2]: from __future__ import print_function
        def display_nodes(nodes):
            for i, node in enumerate(nodes):
                print('%d %s %s' % (i, node.name, node.op))
                [print(u'  %d   %s' % (i, n)) for i, n in enumerate(node.input)]
```

0.0.2 Work around GPU memory issues (not needed with GTX1080)

```
In [3]: NEED_GPU_MEM_WORKAROUND = False
        if (NEED_GPU_MEM_WORKAROUND):
            print('Working around TF GPU mem issues')
            import tensorflow as tf
            import keras.backend.tensorflow_backend as ktf

            def get_session(gpu_fraction=0.6):
                gpu_options = tf.GPUOptions(per_process_gpu_memory_fraction=gpu_fraction,
                                        allow_growth=True)
                return tf.Session(config=tf.ConfigProto(gpu_options=gpu_options))

            ktf.set_session(get_session())
```

```
In [4]: img_width, img_height = 224, 224 # change based on the shape/structure of your images
        num_classes = 2 # Fire or Safe

        # define function to load train, test, and validation datasets
        def load_dataset(path):
            data = load_files(path)
            fire_files = np.array(data['filenames'])
            fire_targets = np_utils.to_categorical(np.array(data['target']), num_classes)
            return fire_files, fire_targets

        # load train, test, and validation datasets
        train_files, train_targets = load_dataset('fireImages/train')
        valid_files, valid_targets = load_dataset('fireImages/valid')
        test_files, test_targets = load_dataset('fireImages/test')

        # load list of fire classes
        class_names = [item[21:-1] for item in sorted(glob("fireImages/train/*/"))]

        # print statistics about the dataset
        print('There are %d total fire categories.' % len(class_names))
        print('There are %s total fire images.\n' % len(np.hstack([train_files, valid_files, tes
        print('There are %d training fire images.' % len(train_files))
        print('There are %d validation fire images.' % len(valid_files))
        print('There are %d test fire images.'% len(test_files))
```

```
There are 2 total fire categories.
There are 1069 total fire images.

There are 1015 training fire images.
There are 45 validation fire images.
There are 9 test fire images.
```

```
In [5]: class_names
```

```
Out[5]: ['Fire', 'Safe']
```

```
In [6]: from keras.preprocessing import image
        from tqdm import tqdm

        def path_to_tensor(img_path):
            # loads RGB image as PIL.Image.Image type
            img = image.load_img(img_path, target_size=(img_width, img_height))
            # convert PIL.Image.Image type to 3D tensor with shape (224, 224, 3)
            x = image.img_to_array(img)
            # convert 3D tensor to 4D tensor with shape (1, 224, 224, 3) and return 4D tensor
            return np.expand_dims(x, axis=0)

        def paths_to_tensor(img_paths):
            list_of_tensors = [path_to_tensor(img_path) for img_path in tqdm(img_paths)]
            return np.vstack(list_of_tensors)
```

```
In [7]: from PIL import ImageFile
        from PIL import Image
        ImageFile.LOAD_TRUNCATED_IMAGES = True

        # pre-process the data for Keras
        train_tensors = preprocess_input( paths_to_tensor(train_files) )
        valid_tensors = preprocess_input( paths_to_tensor(valid_files) )
        test_tensors  = preprocess_input( paths_to_tensor(test_files) )
```

```
100%|| 1015/1015 [00:30<00:00, 33.17it/s]
100%|| 45/45 [00:01<00:00, 31.89it/s]
100%|| 9/9 [00:00<00:00, 55.78it/s]
```

0.0.3 Detect fire with the most accurate model

```
In [8]: ### Define the architecture.
        from keras.layers import Conv2D, MaxPooling2D, GlobalAveragePooling2D
        from keras.layers import Dropout, Flatten, Dense
        from keras.models import Sequential
        from keras import applications

        from keras.applications.resnet50 import preprocess_input, decode_predictions
        VGG16_model       = applications.VGG16(input_shape=(img_width, img_height, 3), weights =
        InceptionV3_model = applications.InceptionV3(input_shape=(img_width, img_height, 3), wei
        Xception_model    = applications.Xception(input_shape=(img_width, img_height, 3), weight
        ResNet50_model    = applications.ResNet50(input_shape=(img_width, img_height, 3), weight
        VGG19_model       = applications.VGG19(input_shape=(img_width, img_height, 3), weights =
```

0.0.4 Pick the most accurate model for this application

```
In [9]: #base_model = InceptionV3_model
        #base_model = Xception_model
        #base_model = ResNet50_model
        #base_model = VGG16_model
        base_model = VGG19_model
```

```
In [10]: base_model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_5 (InputLayer)         (None, 224, 224, 3)       0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
```

```
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv4 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv4 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv4 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
=================================================================
Total params: 20,024,384
Trainable params: 20,024,384
Non-trainable params: 0
_________________________________________________________________
```

0.0.5 Transfer learning (freeze base model layers)

```
In [11]: for i, layer in enumerate(base_model.layers):
            print(i, layer.name)
```

```
0 input_5
1 block1_conv1
2 block1_conv2
3 block1_pool
4 block2_conv1
5 block2_conv2
6 block2_pool
7 block3_conv1
8 block3_conv2
9 block3_conv3
10 block3_conv4
11 block3_pool
12 block4_conv1
13 block4_conv2
14 block4_conv3
15 block4_conv4
16 block4_pool
17 block5_conv1
18 block5_conv2
19 block5_conv3
20 block5_conv4
21 block5_pool
```

```
In [12]: # Freeze the layers which you don't want to train.
         for layer in base_model.layers:
             layer.trainable = False

         # Here I am freezing the first 5 layers.
         # for layer in model.layers[:5]:
         #    layer.trainable = False
```

```
In [13]: # add a global spatial average pooling layer
         x = base_model.output
         x = GlobalAveragePooling2D()(x)
         # x = Dropout(0.45)(x)
         # and a logistic layer we have num_classes classes
         predictions = Dense(num_classes, activation='softmax')(x)
```

```
In [14]: from keras.models import Model
         # this is the model we will train
         model = Model(inputs=base_model.input, outputs=predictions)
```

```
In [15]: # compile the model (should be done *after* setting layers to non-trainable)
         model.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy'
         #model.compile(loss='categorical_crossentropy', optimizer='adadelta', metrics=['accurac
In [16]: model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_5 (InputLayer)         (None, 224, 224, 3)       0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv4 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv4 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv4 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
```

```
_________________________________________________________________
global_average_pooling2d_1 ( (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 1026
=================================================================
Total params: 20,025,410
Trainable params: 1,026
Non-trainable params: 20,024,384
_________________________________________________________________
```

Note that we only train 1,539 params and the rest are frozen:

- Total params: 20,025,410
- Trainable params: 1,026
- Non-trainable params: 20,024,384

```
In [17]: from keras.callbacks import ModelCheckpoint
         from keras.callbacks import TensorBoard

         tbCallback = TensorBoard(log_dir='./Graph', histogram_freq=0, write_graph=True, write_i

         import keras.backend.tensorflow_backend as K

         # train the model
         checkpointer = ModelCheckpoint(filepath='firemodel.weights.best.hdf5', verbose=3, save_
         hist = model.fit(train_tensors, train_targets, batch_size=64, epochs=10,
               validation_data=(valid_tensors, valid_targets), callbacks=[checkpointer, tbCallba
               verbose=2)  #, shuffle=True)
```

```
Train on 1015 samples, validate on 45 samples
Epoch 1/10
Epoch 00001: val_loss improved from inf to 0.70525, saving model to firemodel.weights.best.hdf5
 - 12s - loss: 1.2621 - acc: 0.6473 - val_loss: 0.7052 - val_acc: 0.7556
Epoch 2/10
Epoch 00002: val_loss improved from 0.70525 to 0.44532, saving model to firemodel.weights.best.h
 - 6s - loss: 0.5690 - acc: 0.8158 - val_loss: 0.4453 - val_acc: 0.8000
Epoch 3/10
Epoch 00003: val_loss improved from 0.44532 to 0.25189, saving model to firemodel.weights.best.h
 - 6s - loss: 0.3952 - acc: 0.8700 - val_loss: 0.2519 - val_acc: 0.8889
Epoch 4/10
Epoch 00004: val_loss improved from 0.25189 to 0.18459, saving model to firemodel.weights.best.h
 - 6s - loss: 0.3114 - acc: 0.9025 - val_loss: 0.1846 - val_acc: 0.9111
Epoch 5/10
Epoch 00005: val_loss improved from 0.18459 to 0.13390, saving model to firemodel.weights.best.h
 - 6s - loss: 0.2497 - acc: 0.9153 - val_loss: 0.1339 - val_acc: 0.9111
Epoch 6/10
Epoch 00006: val_loss improved from 0.13390 to 0.10070, saving model to firemodel.weights.best.h
 - 6s - loss: 0.2169 - acc: 0.9320 - val_loss: 0.1007 - val_acc: 0.9333
```

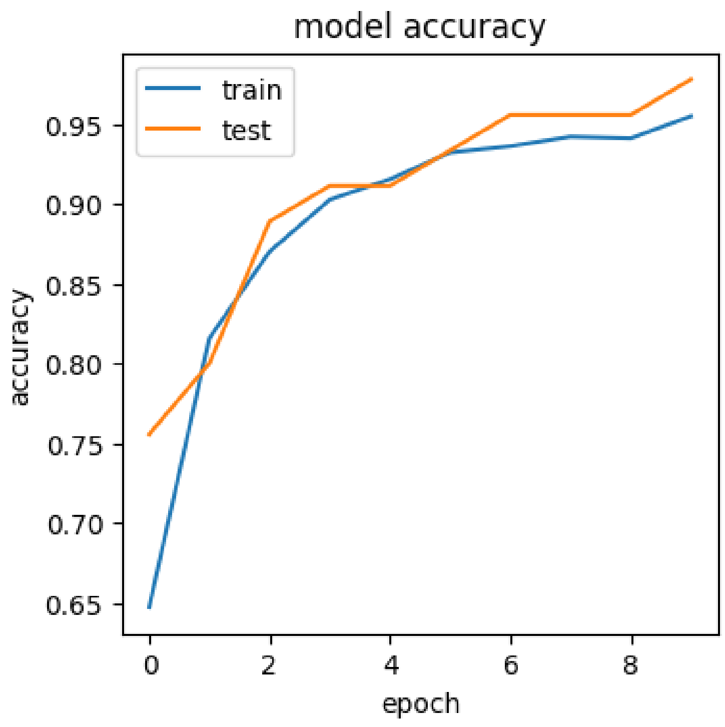
```
Epoch 7/10
Epoch 00007: val_loss did not improve
 - 6s - loss: 0.1907 - acc: 0.9360 - val_loss: 0.1242 - val_acc: 0.9556
Epoch 8/10
Epoch 00008: val_loss did not improve
 - 6s - loss: 0.1607 - acc: 0.9419 - val_loss: 0.1383 - val_acc: 0.9556
Epoch 9/10
Epoch 00009: val_loss improved from 0.10070 to 0.09260, saving model to firemodel.weights.best.h
 - 6s - loss: 0.1511 - acc: 0.9409 - val_loss: 0.0926 - val_acc: 0.9556
Epoch 10/10
Epoch 00010: val_loss improved from 0.09260 to 0.06809, saving model to firemodel.weights.best.h
 - 6s - loss: 0.1236 - acc: 0.9547 - val_loss: 0.0681 - val_acc: 0.9778
```

```
In [18]: for i, layer in enumerate(model.layers):
            print(i, layer.name)
```

```
0 input_5
1 block1_conv1
2 block1_conv2
3 block1_pool
4 block2_conv1
5 block2_conv2
6 block2_pool
7 block3_conv1
8 block3_conv2
9 block3_conv3
10 block3_conv4
11 block3_pool
12 block4_conv1
13 block4_conv2
14 block4_conv3
15 block4_conv4
16 block4_pool
17 block5_conv1
18 block5_conv2
19 block5_conv3
20 block5_conv4
21 block5_pool
22 global_average_pooling2d_1
23 dense_1
```

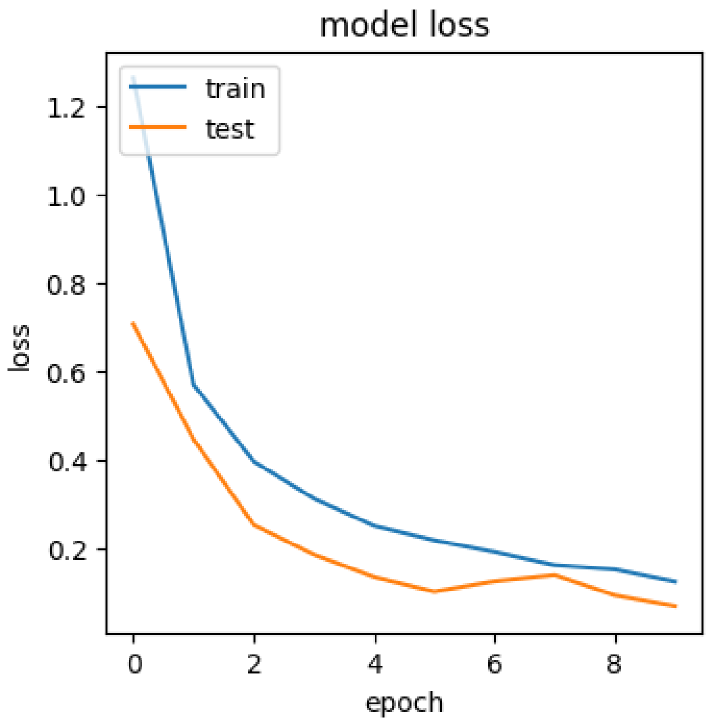
```
In [19]: import cv2
         import matplotlib.pyplot as plt

         # summarize history for accuracy
         plt.figure(figsize=(4,4), dpi=100 )
```

```
        plt.plot(hist.history['acc'])
        plt.plot(hist.history['val_acc'])
        plt.title('model accuracy')
        plt.ylabel('accuracy')
        plt.xlabel('epoch')
        plt.legend(['train', 'test'], loc='upper left')
        plt.show()
        plt.savefig('training1.png', dpi=300)
```



```
<matplotlib.figure.Figure at 0x7eff2a5ff190>
```

```
In [20]: # summarize history for loss
         plt.figure(figsize=(4,4), dpi=100 )
         plt.plot(hist.history['loss'])
         plt.plot(hist.history['val_loss'])
         plt.title('model loss')
         plt.ylabel('loss')
         plt.xlabel('epoch')
         plt.legend(['train', 'test'], loc='upper left')
         plt.show()
```



0.0.6 Load the Model with the Best Validation Loss

```
In [21]: ### Load the model weights with the best validation loss.
         model.load_weights('firemodel.weights.best.hdf5')
```

0.1 Test the Fire detection Model

0.1.1 Calculate classification accuracy on the test dataset.

```
In [22]: # get index of predicted fire class for each image in test set
         predictions = [np.argmax(model.predict(np.expand_dims(feature, axis=0))) for feature in

In [23]: # report test accuracy
         test_accuracy = 100*np.sum ( np.array( predictions)==np.argmax(test_targets, axis=1) )
         print('Test accuracy: %.4f%%' % test_accuracy)
```

```
Test accuracy: 100.0000%
```

```
In [24]: # test validation accuracy
         predictions = [np.argmax(model.predict(np.expand_dims(feature, axis=0))) for feature in
         valid_accuracy = 100*np.sum ( np.array( predictions)==np.argmax(valid_targets, axis=1)
         print('Validation accuracy: %.4f%%' % valid_accuracy)
```

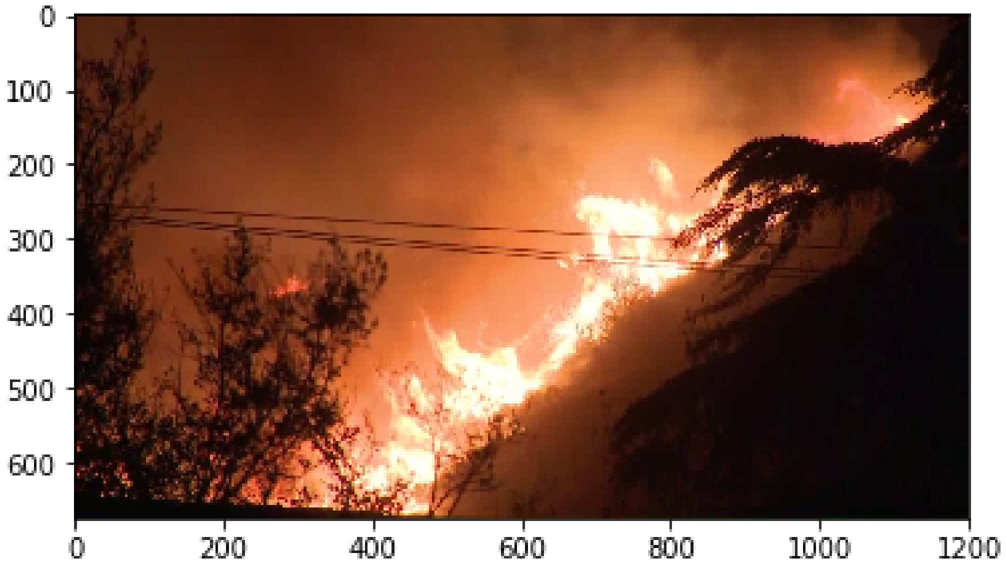
```
Validation accuracy: 97.0000%
```

```
In [25]: def detect_fire(img_path):
             predicted_vector = model.predict(preprocess_input(path_to_tensor(img_path)))
             return class_names[np.argmax(predicted_vector)]
```

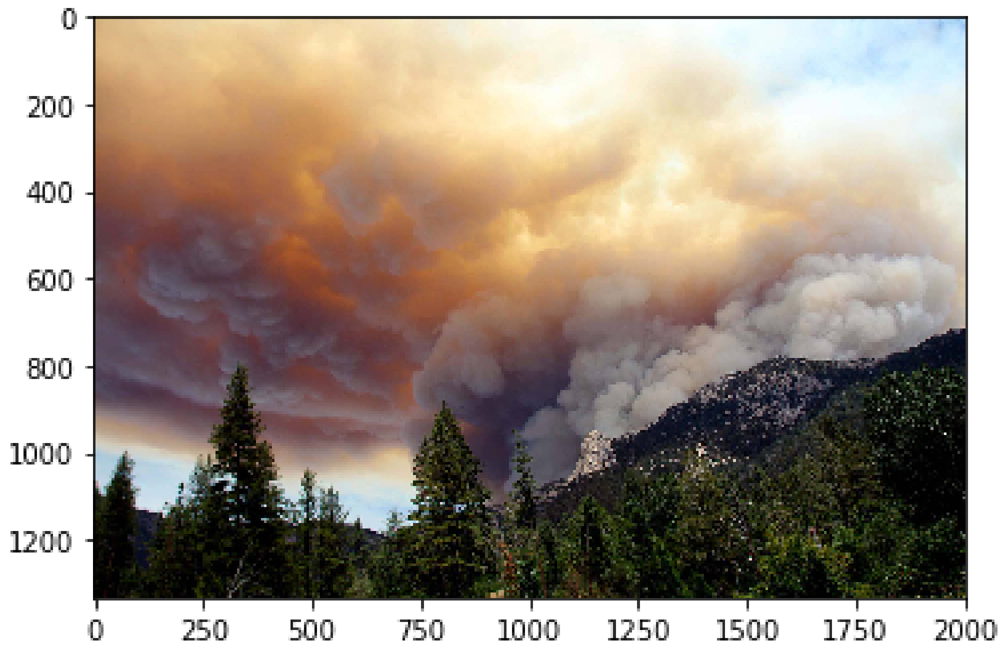
```
In [26]: import glob
         from PIL import Image
         from IPython import display

         path = "test_images/*"
         for fname in glob.glob(path):
             fire_detection = detect_fire( fname )
             display.display(plt.gcf())
             print('___________________________________________________________________________
             print('')
             if (fire_detection=='Fire'):
                 print ('\033[1m'+'\033[91m'+'ALARM: Detected '+fire_detection.replace("_", ""))
             elif (fire_detection=='Smoke'):
                 print ('\033[1m'+'\033[91m'+'ALARM: Detected '+fire_detection.replace("_", ""))
             else:
                 print ('\033[1m'+'\033[92m'+'Looking good: ' +fire_detection.replace("_", ""))
             plt.imshow(Image.open(fname))
             #raw_input()
```

```
<matplotlib.figure.Figure at 0x7eff1ca783d0>


_____________________________________________________________________________________________

Looking good: Safe
```

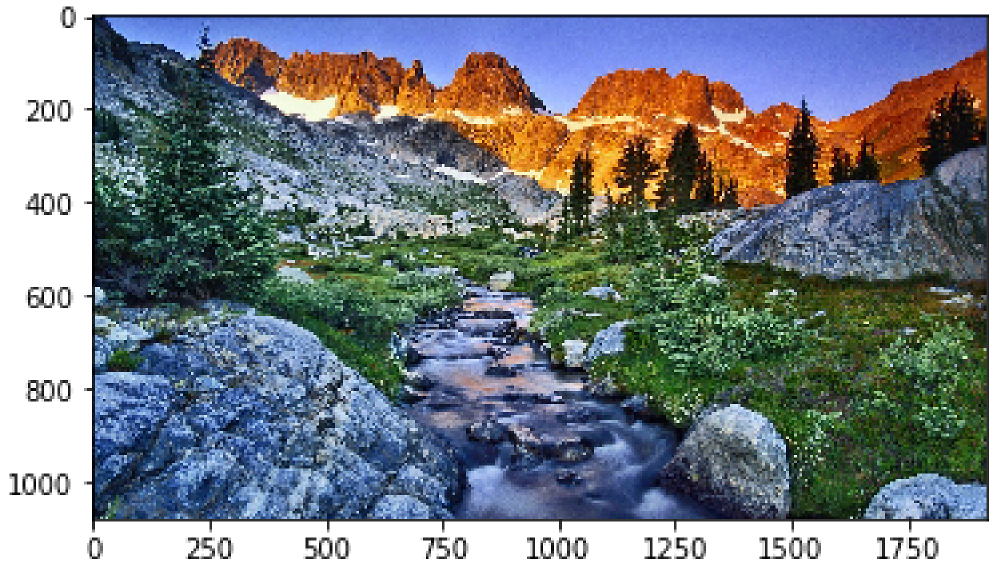
```
------------------------------------------------------------------------------------------

ALARM: Detected Fire
```



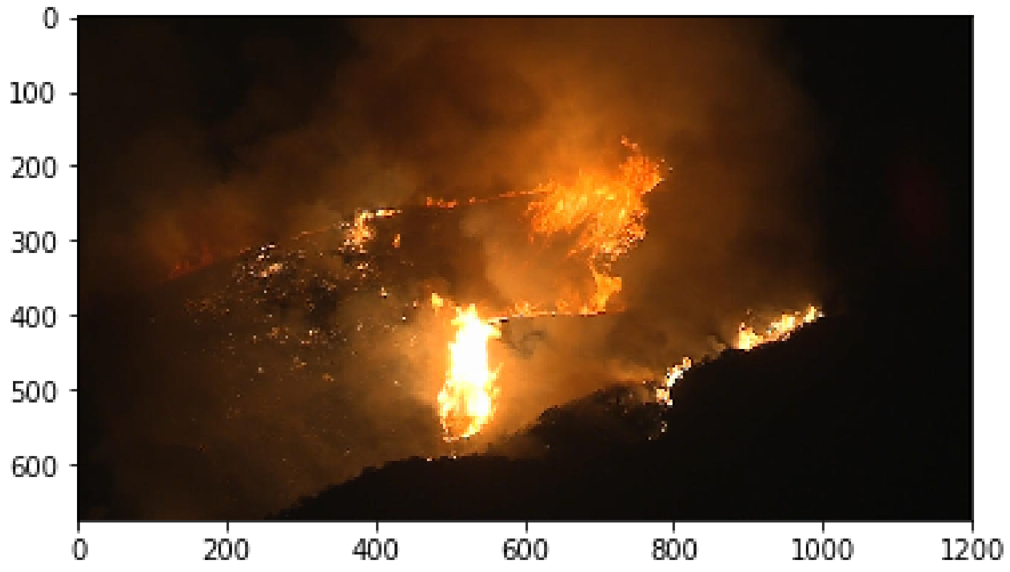
```
------------------------------------------------------------------------------------------

ALARM: Detected Fire
```



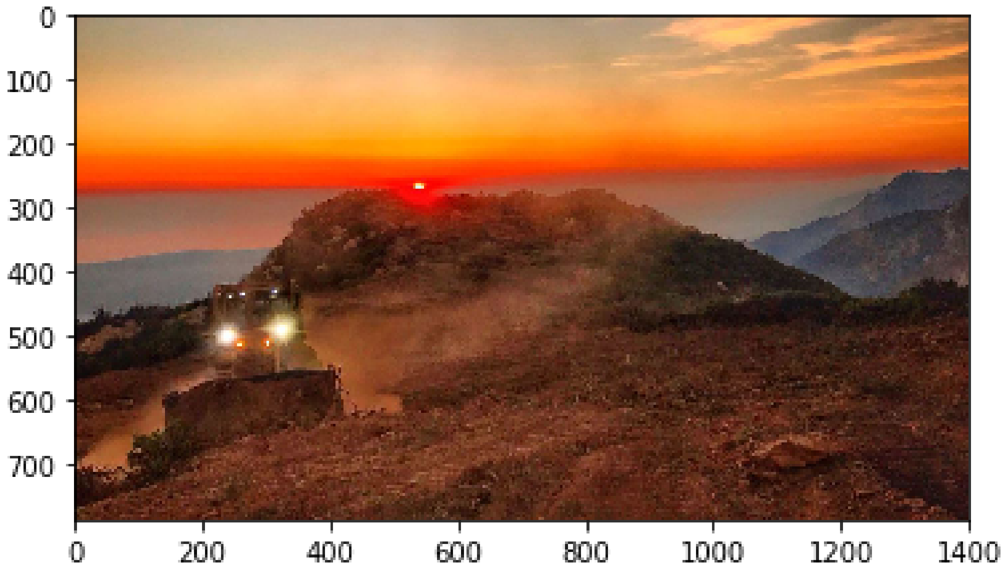
```
----------------------------------------------------------------------------------------------------

Looking good: Safe
```



```
----------------------------------------------------------------------------------------------------

ALARM: Detected Fire
```

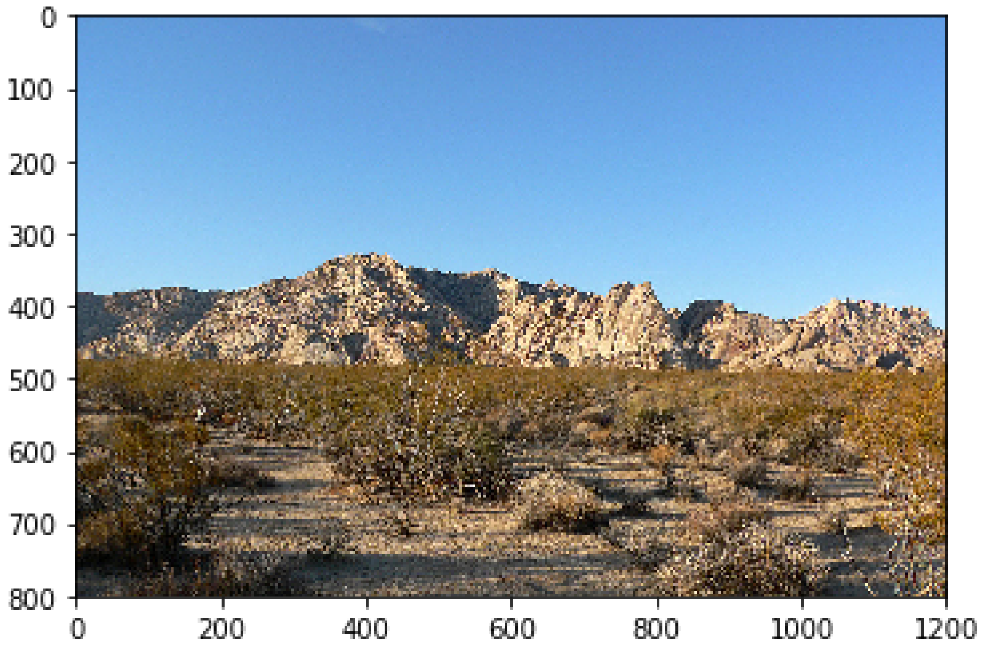


```
--------------------------------------------------------------------------------

ALARM: Detected Fire
```



```
--------------------------------------------------------------------------------

Looking good: Safe
```

```
--------------------------------------------------------------------------------

Looking good: Safe
```



```
--------------------------------------------------------------------------------

ALARM: Detected Fire
```

1 Convert the Keras / TF model to something that Jetson TX2 understands

1.0.1 Freeze the graph

```
In [27]: from keras.models import load_model
         from tensorflow.python.framework import graph_io
         from tensorflow.python.tools import freeze_graph
         from tensorflow.core.protobuf import saver_pb2
         from tensorflow.python.training import saver as saver_lib

In [28]: import tensorflow as tf
         sess = K.get_session()

In [29]: saver = saver_lib.Saver(write_version=saver_pb2.SaverDef.V2)

In [30]: checkpoint_path = saver.save(sess,'fire_detector_saved_ckpt', global_step=0,
                                      latest_filename='fire_detector_checkpoint_state')

In [31]: graph_io.write_graph(sess.graph, '.', 'fire_detector_tmp.pb')

Out[31]: './fire_detector_tmp.pb'

In [32]: freeze_graph.freeze_graph('./fire_detector_tmp.pb', '', False, checkpoint_path, 'dense_
                                   "save/restore_all", "the_labels", 'frozen_fire_detector.pb',
INFO:tensorflow:Restoring parameters from fire_detector_saved_ckpt-0
INFO:tensorflow:Froze 34 variables.
Converted 34 variables to const ops.
127 ops in the final graph.
```

```
In [33]: # read frozen graph and display nodes
         graph = tf.GraphDef()
         with tf.gfile.Open('frozen_fire_detector.pb', 'r') as f:
             data = f.read()
             graph.ParseFromString(data)

         display_nodes(graph.node)
```

```
0 input_5 Placeholder
1 block1_conv1_2/kernel Const
2 block1_conv1_2/kernel/read Identity
 0  block1_conv1_2/kernel
3 block1_conv1_2/bias Const
4 block1_conv1_2/bias/read Identity
 0  block1_conv1_2/bias
5 block1_conv1_2/convolution Conv2D
 0  input_5
 1  block1_conv1_2/kernel/read
6 block1_conv1_2/BiasAdd BiasAdd
 0  block1_conv1_2/convolution
 1  block1_conv1_2/bias/read
7 block1_conv1_2/Relu Relu
 0  block1_conv1_2/BiasAdd
8 block1_conv2_2/kernel Const
9 block1_conv2_2/kernel/read Identity
 0  block1_conv2_2/kernel
10 block1_conv2_2/bias Const
11 block1_conv2_2/bias/read Identity
 0  block1_conv2_2/bias
12 block1_conv2_2/convolution Conv2D
 0  block1_conv1_2/Relu
 1  block1_conv2_2/kernel/read
13 block1_conv2_2/BiasAdd BiasAdd
 0  block1_conv2_2/convolution
 1  block1_conv2_2/bias/read
14 block1_conv2_2/Relu Relu
 0  block1_conv2_2/BiasAdd
15 block1_pool_1/MaxPool MaxPool
 0  block1_conv2_2/Relu
16 block2_conv1_1/kernel Const
17 block2_conv1_1/kernel/read Identity
 0  block2_conv1_1/kernel
```

```
18 block2_conv1_1/bias Const
19 block2_conv1_1/bias/read Identity
 0  block2_conv1_1/bias
20 block2_conv1_1/convolution Conv2D
 0  block1_pool_1/MaxPool
 1  block2_conv1_1/kernel/read
21 block2_conv1_1/BiasAdd BiasAdd
 0  block2_conv1_1/convolution
 1  block2_conv1_1/bias/read
22 block2_conv1_1/Relu Relu
 0  block2_conv1_1/BiasAdd
23 block2_conv2_1/kernel Const
24 block2_conv2_1/kernel/read Identity
 0  block2_conv2_1/kernel
25 block2_conv2_1/bias Const
26 block2_conv2_1/bias/read Identity
 0  block2_conv2_1/bias
27 block2_conv2_1/convolution Conv2D
 0  block2_conv1_1/Relu
 1  block2_conv2_1/kernel/read
28 block2_conv2_1/BiasAdd BiasAdd
 0  block2_conv2_1/convolution
 1  block2_conv2_1/bias/read
29 block2_conv2_1/Relu Relu
 0  block2_conv2_1/BiasAdd
30 block2_pool_2/MaxPool MaxPool
 0  block2_conv2_1/Relu
31 block3_conv1_1/kernel Const
32 block3_conv1_1/kernel/read Identity
 0  block3_conv1_1/kernel
33 block3_conv1_1/bias Const
34 block3_conv1_1/bias/read Identity
 0  block3_conv1_1/bias
35 block3_conv1_1/convolution Conv2D
 0  block2_pool_2/MaxPool
 1  block3_conv1_1/kernel/read
36 block3_conv1_1/BiasAdd BiasAdd
 0  block3_conv1_1/convolution
 1  block3_conv1_1/bias/read
37 block3_conv1_1/Relu Relu
 0  block3_conv1_1/BiasAdd
38 block3_conv2_1/kernel Const
39 block3_conv2_1/kernel/read Identity
 0  block3_conv2_1/kernel
40 block3_conv2_1/bias Const
41 block3_conv2_1/bias/read Identity
 0  block3_conv2_1/bias
42 block3_conv2_1/convolution Conv2D
```

```
 0  block3_conv1_1/Relu
 1  block3_conv2_1/kernel/read
43 block3_conv2_1/BiasAdd BiasAdd
 0  block3_conv2_1/convolution
 1  block3_conv2_1/bias/read
44 block3_conv2_1/Relu Relu
 0  block3_conv2_1/BiasAdd
45 block3_conv3_1/kernel Const
46 block3_conv3_1/kernel/read Identity
 0  block3_conv3_1/kernel
47 block3_conv3_1/bias Const
48 block3_conv3_1/bias/read Identity
 0  block3_conv3_1/bias
49 block3_conv3_1/convolution Conv2D
 0  block3_conv2_1/Relu
 1  block3_conv3_1/kernel/read
50 block3_conv3_1/BiasAdd BiasAdd
 0  block3_conv3_1/convolution
 1  block3_conv3_1/bias/read
51 block3_conv3_1/Relu Relu
 0  block3_conv3_1/BiasAdd
52 block3_conv4/kernel Const
53 block3_conv4/kernel/read Identity
 0  block3_conv4/kernel
54 block3_conv4/bias Const
55 block3_conv4/bias/read Identity
 0  block3_conv4/bias
56 block3_conv4/convolution Conv2D
 0  block3_conv3_1/Relu
 1  block3_conv4/kernel/read
57 block3_conv4/BiasAdd BiasAdd
 0  block3_conv4/convolution
 1  block3_conv4/bias/read
58 block3_conv4/Relu Relu
 0  block3_conv4/BiasAdd
59 block3_pool_2/MaxPool MaxPool
 0  block3_conv4/Relu
60 block4_conv1_1/kernel Const
61 block4_conv1_1/kernel/read Identity
 0  block4_conv1_1/kernel
62 block4_conv1_1/bias Const
63 block4_conv1_1/bias/read Identity
 0  block4_conv1_1/bias
64 block4_conv1_1/convolution Conv2D
 0  block3_pool_2/MaxPool
 1  block4_conv1_1/kernel/read
65 block4_conv1_1/BiasAdd BiasAdd
 0  block4_conv1_1/convolution
 1  block4_conv1_1/bias/read
66 block4_conv1_1/Relu Relu
 0  block4_conv1_1/BiasAdd
67 block4_conv2_1/kernel Const
68 block4_conv2_1/kernel/read Identity
 0  block4_conv2_1/kernel
69 block4_conv2_1/bias Const
70 block4_conv2_1/bias/read Identity
 0  block4_conv2_1/bias
71 block4_conv2_1/convolution Conv2D
 0  block4_conv1_1/Relu
 1  block4_conv2_1/kernel/read
72 block4_conv2_1/BiasAdd BiasAdd
 0  block4_conv2_1/convolution
 1  block4_conv2_1/bias/read
73 block4_conv2_1/Relu Relu
 0  block4_conv2_1/BiasAdd
74 block4_conv3_1/kernel Const
75 block4_conv3_1/kernel/read Identity
 0  block4_conv3_1/kernel
76 block4_conv3_1/bias Const
77 block4_conv3_1/bias/read Identity
 0  block4_conv3_1/bias
78 block4_conv3_1/convolution Conv2D
 0  block4_conv2_1/Relu
 1  block4_conv3_1/kernel/read
79 block4_conv3_1/BiasAdd BiasAdd
 0  block4_conv3_1/convolution
 1  block4_conv3_1/bias/read
80 block4_conv3_1/Relu Relu
 0  block4_conv3_1/BiasAdd
81 block4_conv4/kernel Const
82 block4_conv4/kernel/read Identity
 0  block4_conv4/kernel
83 block4_conv4/bias Const
84 block4_conv4/bias/read Identity
 0  block4_conv4/bias
85 block4_conv4/convolution Conv2D
 0  block4_conv3_1/Relu
 1  block4_conv4/kernel/read
86 block4_conv4/BiasAdd BiasAdd
 0  block4_conv4/convolution
 1  block4_conv4/bias/read
87 block4_conv4/Relu Relu
 0  block4_conv4/BiasAdd
88 block4_pool_2/MaxPool MaxPool
 0  block4_conv4/Relu
89 block5_conv1_1/kernel Const
```

```
90 block5_conv1_1/kernel/read Identity
 0  block5_conv1_1/kernel
91 block5_conv1_1/bias Const
92 block5_conv1_1/bias/read Identity
 0  block5_conv1_1/bias
93 block5_conv1_1/convolution Conv2D
 0  block4_pool_2/MaxPool
 1  block5_conv1_1/kernel/read
94 block5_conv1_1/BiasAdd BiasAdd
 0  block5_conv1_1/convolution
 1  block5_conv1_1/bias/read
95 block5_conv1_1/Relu Relu
 0  block5_conv1_1/BiasAdd
96 block5_conv2_1/kernel Const
97 block5_conv2_1/kernel/read Identity
 0  block5_conv2_1/kernel
98 block5_conv2_1/bias Const
99 block5_conv2_1/bias/read Identity
 0  block5_conv2_1/bias
100 block5_conv2_1/convolution Conv2D
 0  block5_conv1_1/Relu
 1  block5_conv2_1/kernel/read
101 block5_conv2_1/BiasAdd BiasAdd
 0  block5_conv2_1/convolution
 1  block5_conv2_1/bias/read
102 block5_conv2_1/Relu Relu
 0  block5_conv2_1/BiasAdd
103 block5_conv3_1/kernel Const
104 block5_conv3_1/kernel/read Identity
 0  block5_conv3_1/kernel
105 block5_conv3_1/bias Const
106 block5_conv3_1/bias/read Identity
 0  block5_conv3_1/bias
107 block5_conv3_1/convolution Conv2D
 0  block5_conv2_1/Relu
 1  block5_conv3_1/kernel/read
108 block5_conv3_1/BiasAdd BiasAdd
 0  block5_conv3_1/convolution
 1  block5_conv3_1/bias/read
109 block5_conv3_1/Relu Relu
 0  block5_conv3_1/BiasAdd
110 block5_conv4/kernel Const
111 block5_conv4/kernel/read Identity
 0  block5_conv4/kernel
112 block5_conv4/bias Const
113 block5_conv4/bias/read Identity
 0  block5_conv4/bias
114 block5_conv4/convolution Conv2D
```

```
 0  block5_conv3_1/Relu
 1  block5_conv4/kernel/read
115 block5_conv4/BiasAdd BiasAdd
 0  block5_conv4/convolution
 1  block5_conv4/bias/read
116 block5_conv4/Relu Relu
 0  block5_conv4/BiasAdd
117 block5_pool_1/MaxPool MaxPool
 0  block5_conv4/Relu
118 global_average_pooling2d_1/Mean/reduction_indices Const
119 global_average_pooling2d_1/Mean Mean
 0  block5_pool_1/MaxPool
 1  global_average_pooling2d_1/Mean/reduction_indices
120 dense_1/kernel Const
121 dense_1/kernel/read Identity
 0  dense_1/kernel
122 dense_1/bias Const
123 dense_1/bias/read Identity
 0  dense_1/bias
124 dense_1/MatMul MatMul
 0  global_average_pooling2d_1/Mean
 1  dense_1/kernel/read
125 dense_1/BiasAdd BiasAdd
 0  dense_1/MatMul
 1  dense_1/bias/read
126 dense_1/Softmax Softmax
 0  dense_1/BiasAdd
```

1.0.2 Convert to UFF

```
In [34]: !convert-to-uff tensorflow --input-file frozen_fire_detector.pb -l
```

```
Loading frozen_fire_detector.pb
1 Placeholder: "input_5"
2 Const: "block1_conv1_2/kernel"
3 Identity: "block1_conv1_2/kernel/read"
4 Const: "block1_conv1_2/bias"
5 Identity: "block1_conv1_2/bias/read"
6 Conv2D: "block1_conv1_2/convolution"
7 BiasAdd: "block1_conv1_2/BiasAdd"
8 Relu: "block1_conv1_2/Relu"
9 Const: "block1_conv2_2/kernel"
10 Identity: "block1_conv2_2/kernel/read"
11 Const: "block1_conv2_2/bias"
12 Identity: "block1_conv2_2/bias/read"
13 Conv2D: "block1_conv2_2/convolution"
14 BiasAdd: "block1_conv2_2/BiasAdd"
```

```
15 Relu: "block1_conv2_2/Relu"
16 MaxPool: "block1_pool_1/MaxPool"
17 Const: "block2_conv1_1/kernel"
18 Identity: "block2_conv1_1/kernel/read"
19 Const: "block2_conv1_1/bias"
20 Identity: "block2_conv1_1/bias/read"
21 Conv2D: "block2_conv1_1/convolution"
22 BiasAdd: "block2_conv1_1/BiasAdd"
23 Relu: "block2_conv1_1/Relu"
24 Const: "block2_conv2_1/kernel"
25 Identity: "block2_conv2_1/kernel/read"
26 Const: "block2_conv2_1/bias"
27 Identity: "block2_conv2_1/bias/read"
28 Conv2D: "block2_conv2_1/convolution"
29 BiasAdd: "block2_conv2_1/BiasAdd"
30 Relu: "block2_conv2_1/Relu"
31 MaxPool: "block2_pool_2/MaxPool"
32 Const: "block3_conv1_1/kernel"
33 Identity: "block3_conv1_1/kernel/read"
34 Const: "block3_conv1_1/bias"
35 Identity: "block3_conv1_1/bias/read"
36 Conv2D: "block3_conv1_1/convolution"
37 BiasAdd: "block3_conv1_1/BiasAdd"
38 Relu: "block3_conv1_1/Relu"
39 Const: "block3_conv2_1/kernel"
40 Identity: "block3_conv2_1/kernel/read"
41 Const: "block3_conv2_1/bias"
42 Identity: "block3_conv2_1/bias/read"
43 Conv2D: "block3_conv2_1/convolution"
44 BiasAdd: "block3_conv2_1/BiasAdd"
45 Relu: "block3_conv2_1/Relu"
46 Const: "block3_conv3_1/kernel"
47 Identity: "block3_conv3_1/kernel/read"
48 Const: "block3_conv3_1/bias"
49 Identity: "block3_conv3_1/bias/read"
50 Conv2D: "block3_conv3_1/convolution"
51 BiasAdd: "block3_conv3_1/BiasAdd"
52 Relu: "block3_conv3_1/Relu"
53 Const: "block3_conv4/kernel"
54 Identity: "block3_conv4/kernel/read"
55 Const: "block3_conv4/bias"
56 Identity: "block3_conv4/bias/read"
57 Conv2D: "block3_conv4/convolution"
58 BiasAdd: "block3_conv4/BiasAdd"
59 Relu: "block3_conv4/Relu"
60 MaxPool: "block3_pool_2/MaxPool"
61 Const: "block4_conv1_1/kernel"
62 Identity: "block4_conv1_1/kernel/read"
```

```
63 Const: "block4_conv1_1/bias"
64 Identity: "block4_conv1_1/bias/read"
65 Conv2D: "block4_conv1_1/convolution"
66 BiasAdd: "block4_conv1_1/BiasAdd"
67 Relu: "block4_conv1_1/Relu"
68 Const: "block4_conv2_1/kernel"
69 Identity: "block4_conv2_1/kernel/read"
70 Const: "block4_conv2_1/bias"
71 Identity: "block4_conv2_1/bias/read"
72 Conv2D: "block4_conv2_1/convolution"
73 BiasAdd: "block4_conv2_1/BiasAdd"
74 Relu: "block4_conv2_1/Relu"
75 Const: "block4_conv3_1/kernel"
76 Identity: "block4_conv3_1/kernel/read"
77 Const: "block4_conv3_1/bias"
78 Identity: "block4_conv3_1/bias/read"
79 Conv2D: "block4_conv3_1/convolution"
80 BiasAdd: "block4_conv3_1/BiasAdd"
81 Relu: "block4_conv3_1/Relu"
82 Const: "block4_conv4/kernel"
83 Identity: "block4_conv4/kernel/read"
84 Const: "block4_conv4/bias"
85 Identity: "block4_conv4/bias/read"
86 Conv2D: "block4_conv4/convolution"
87 BiasAdd: "block4_conv4/BiasAdd"
88 Relu: "block4_conv4/Relu"
89 MaxPool: "block4_pool_2/MaxPool"
90 Const: "block5_conv1_1/kernel"
91 Identity: "block5_conv1_1/kernel/read"
92 Const: "block5_conv1_1/bias"
93 Identity: "block5_conv1_1/bias/read"
94 Conv2D: "block5_conv1_1/convolution"
95 BiasAdd: "block5_conv1_1/BiasAdd"
96 Relu: "block5_conv1_1/Relu"
97 Const: "block5_conv2_1/kernel"
98 Identity: "block5_conv2_1/kernel/read"
99 Const: "block5_conv2_1/bias"
100 Identity: "block5_conv2_1/bias/read"
101 Conv2D: "block5_conv2_1/convolution"
102 BiasAdd: "block5_conv2_1/BiasAdd"
103 Relu: "block5_conv2_1/Relu"
104 Const: "block5_conv3_1/kernel"
105 Identity: "block5_conv3_1/kernel/read"
106 Const: "block5_conv3_1/bias"
107 Identity: "block5_conv3_1/bias/read"
108 Conv2D: "block5_conv3_1/convolution"
109 BiasAdd: "block5_conv3_1/BiasAdd"
110 Relu: "block5_conv3_1/Relu"
```

```
111 Const: "block5_conv4/kernel"
112 Identity: "block5_conv4/kernel/read"
113 Const: "block5_conv4/bias"
114 Identity: "block5_conv4/bias/read"
115 Conv2D: "block5_conv4/convolution"
116 BiasAdd: "block5_conv4/BiasAdd"
117 Relu: "block5_conv4/Relu"
118 MaxPool: "block5_pool_1/MaxPool"
119 Const: "global_average_pooling2d_1/Mean/reduction_indices"
120 Mean: "global_average_pooling2d_1/Mean"
121 Const: "dense_1/kernel"
122 Identity: "dense_1/kernel/read"
123 Const: "dense_1/bias"
124 Identity: "dense_1/bias/read"
125 MatMul: "dense_1/MatMul"
126 BiasAdd: "dense_1/BiasAdd"
127 Softmax: "dense_1/Softmax"
```

1.0.3 Convert to uff

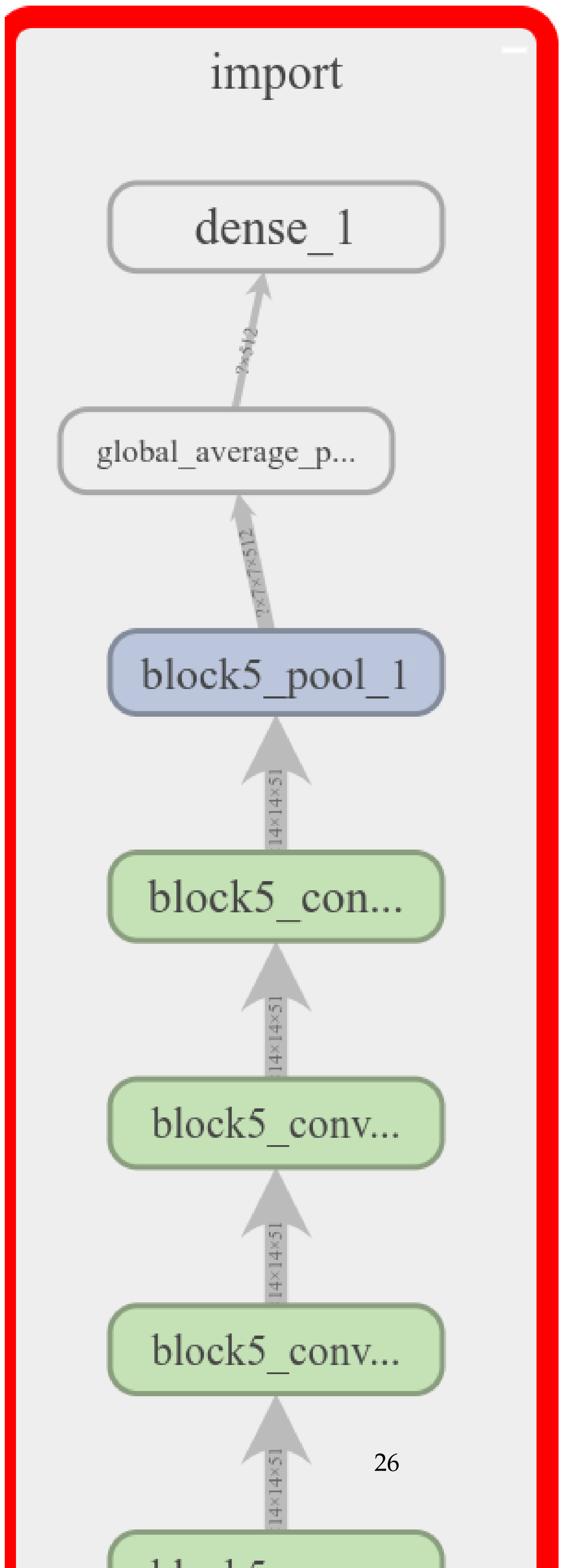
```
In [35]: !convert-to-uff tensorflow -o fire_detector.uff --input-file frozen_fire_detector.pb -O
```

```
Loading frozen_fire_detector.pb
Using output node dense_1/Softmax
Converting to UFF graph
No. nodes: 93
UFF Output written to fire_detector.uff
```

```
In [36]: !ls -lh fire_detector.uff
```

```
-rw-rw-r-- 1 ngeorgis ngeorgis 77M Feb 16 14:55 fire_detector.uff
```

1.0.4 Visualize model using tensorboard


import
dense_1
?×512
global_average_p...
?×7×7×512
block5_pool_1
14×14×51
block5_con...
14×14×51
block5_conv...
14×14×51
block5_conv...
14×14×51