In [1]:
```
%load_ext autoreload
%autoreload 2
```

In [2]:
```
#%autoreload # When utils.py is updated
from utils_unet_resunet import *
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from model.models import Model_3
from model.losses import WBCE
root_path = 'imgs/'
```

In [3]:
```
# Define data type (L8-Landsat8, S2-Sentinel2, S1-Sentinel1)
img_type = 'S2'

if img_type == 'L8':
    # Load images
    ref_2019 = load_tif_image(root_path+'New_Images/References/res_10m/r10m_def_2019
    opt_2018 = load_tif_image(root_path+'New_Images/Landsat8/'+'cut_land8_2018.tif')
    opt_2019 = load_tif_image(root_path+'New_Images/Landsat8/'+'cut_land8_2019.tif')

    # Resize images
    opt_2018 = resize_image(opt_2018.copy(), ref_2019.shape[0], ref_2019.shape[1])
    opt_2019 = resize_image(opt_2019.copy(), ref_2019.shape[0], ref_2019.shape[1])

    # Filter outliers
    opt_2018 = filter_outliers(opt_2018.copy())
    opt_2019 = filter_outliers(opt_2019.copy())

    image_stack = np.concatenate((opt_2018, opt_2019), axis=-1)
    print('landsat_resize:', image_stack.shape)
    del opt_2018, opt_2019

if img_type == 'S2':
    # Load images
    sent2_2018_1 = load_tif_image(root_path+'New_Images/Sentinel2/'+'2018_10m_b2348.
    #sent2_2018_2 = load_tif_image(root_path+'New_Images/Sentinel2/'+'2018_20m_b5678

    # Resize bands of 20m
    #sent2_2018_2 = resize_image(sent2_2018_2.copy(), sent2_2018_1.shape[0], sent2_2
    #sent2_2018 = np.concatenate((sent2_2018_1, sent2_2018_2), axis=-1)
    sent2_2018 = sent2_2018_1.copy()
    del sent2_2018_1#, sent2_2018_2

    sent2_2019_1 = load_tif_image(root_path+'New_Images/Sentinel2/'+'2019_10m_b2348.
    #sent2_2019_2 = load_tif_image(root_path+'New_Images/Sentinel2/'+'2019_20m_b5678

    # Resize bands of 20m
    #sent2_2019_2 = resize_image(sent2_2019_2.copy(), sent2_2019_1.shape[0], sent2_2
    #sent2_2019 = np.concatenate((sent2_2019_1, sent2_2019_2), axis=-1)
    sent2_2019 = sent2_2019_1.copy()
    del sent2_2019_1#, sent2_2019_2

    # Filter outliers
    sent2_2018 = filter_outliers(sent2_2018.copy())
    sent2_2019 = filter_outliers(sent2_2019.copy())

    image_stack = np.concatenate((sent2_2018, sent2_2019), axis=-1)
    print('Image stack:', image_stack.shape)
    del sent2_2018, sent2_2019

if img_type == 'S1':
```

```
    # Load images
    sar_2018_vh = np.expand_dims(load_SAR_image(root_path+'New_Images/Sentinel1/'+'c
    sar_2018_vv = np.expand_dims(load_SAR_image(root_path+'New_Images/Sentinel1/'+'c
    sar_2019_vh = np.expand_dims(load_SAR_image(root_path+'New_Images/Sentinel1/'+'c
    sar_2019_vv = np.expand_dims(load_SAR_image(root_path+'New_Images/Sentinel1/'+'c

    sar_2018 = np.concatenate((sar_2018_vh, sar_2018_vv), axis=-1)
    sar_2019 = np.concatenate((sar_2019_vh, sar_2019_vv), axis=-1)
    del sar_2018_vh, sar_2018_vv, sar_2019_vh, sar_2019_vv

    # Filter outliers
    sar_2018 = filter_outliers(sar_2018.copy())
    sar_2019 = filter_outliers(sar_2019.copy())

    image_stack = np.concatenate((sar_2018, sar_2019), axis=-1)
    print('Image stack:', image_stack.shape)
    del sar_2018, sar_2019

# Load references
# Load current reference
#ref_2019 = load_tif_image(root_path+'New_Images/References/res_10m/r10m_def_2019.ti
# Load past references
#past_ref = np.load(root_path+'New_Images/References/past_ref_and_clouds.npy').astyp
#past_ref1 = load_tif_image(root_path+'New_Images/References/res_10m/r10m_def_1988_2
#past_ref2 = load_tif_image(root_path+'New_Images/References/res_10m/r10m_def_2008_2
#clouds_2018 = load_tif_image(root_path+'New_Images/References/cut_b10_2018.tif').as
#clouds_2018 = resize_image(np.expand_dims(clouds_2018.copy(), axis = -1), ref_2019.
#clouds_2018 = binary_mask_cloud(clouds_2018.copy(), 50)
#clouds_2019 = load_tif_image(root_path+'New_Images/References/cut_b10_2019.tif').as
#clouds_2019 = resize_image(np.expand_dims(clouds_2019.copy(), axis = -1), ref_2019.
#clouds_2019 = binary_mask_cloud(clouds_2019.copy(), 50)
```

```
imgs/New_Images/Sentinel2/2018_10m_b2348.tif
imgs/New_Images/Sentinel2/2019_10m_b2348.tif
Image stack: (17729, 9202, 8)
```

In [4]:

```
# Create label mask
#past_ref = past_ref1 + past_ref2 + clouds_2018 + clouds_2019
#past_ref[past_ref>=1] = 1
#buffer = 2
#final_mask1 = mask_no_considered(ref_2019, buffer, past_ref)
#del past_ref1, past_ref2, clouds_2018, clouds_2019
final_mask1 = np.load(root_path+'New_Images/ref/'+'labels.npy')

lim_x = 10000
lim_y = 7000
image_stack = image_stack[:lim_x, :lim_y, :]
final_mask1 = final_mask1[:lim_x, :lim_y]
#ref_2019 = ref_2019[:lim_x, :lim_y]

h_, w_, channels = image_stack.shape
print('image stack size: ', image_stack.shape)

# Normalization
type_norm = 1
image_array = normalization(image_stack.copy(), type_norm)
print(np.min(image_array), np.max(image_array))
del image_stack

# Print pertengate of each class (whole image)
print('Total no-deforestaion class is {}'.format(len(final_mask1[final_mask1==0])))
print('Total deforestaion class is {}'.format(len(final_mask1[final_mask1==1])))
print('Total past deforestaion class is {}'.format(len(final_mask1[final_mask1==2])))
print('Percentage of deforestaion class is {:.2f}'.format((len(final_mask1[final_mas
```

```
image stack size:  (10000, 7000, 8)
-4.987141 5.626766
Total no-deforestaion class is 36326397
Total deforestaion class is 1048775
Total past deforestaion class is 32624828
Percentage of deforestaion class is 2.89
```

In [5]:

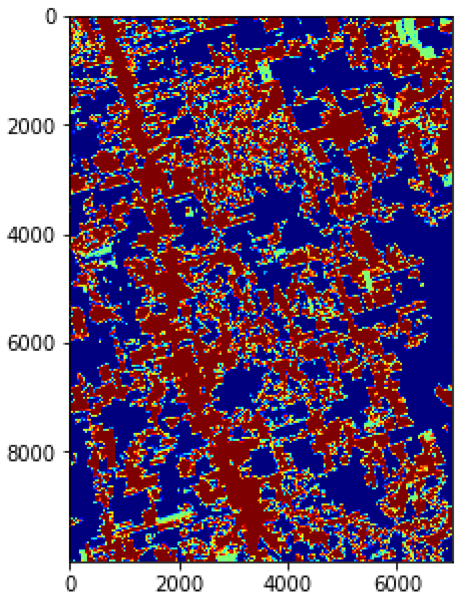
```
# Create tile mask
mask_tiles = create_mask(final_mask1.shape[0], final_mask1.shape[1], grid_size=(5, 4
image_array = image_array[:mask_tiles.shape[0], :mask_tiles.shape[1],:]
final_mask1 = final_mask1[:mask_tiles.shape[0], :mask_tiles.shape[1]]

print('mask: ',mask_tiles.shape)
print('image stack: ', image_array.shape)
print('ref :', final_mask1.shape)
#plt.imshow(mask_tiles)
```

```
Tiles size:  2000 1750
Mask size:  (10000, 7000)
mask:  (10000, 7000)
image stack:  (10000, 7000, 8)
ref : (10000, 7000)
```

In [6]:

```
plt.figure(figsize=(10,5))
plt.imshow(final_mask1, cmap = 'jet')
```

Out[6]: <matplotlib.image.AxesImage at 0x1a71e9c71c0>



In [7]:

```
# Define tiles for training, validation, and test sets
tiles_tr = [1,3,5,8,11,13,14,20]
tiles_val = [6,19]
tiles_ts = (list(set(np.arange(20)+1)-set(tiles_tr)-set(tiles_val)))

mask_tr_val = np.zeros((mask_tiles.shape)).astype('float32')
# Training and validation mask
for tr_ in tiles_tr:
    mask_tr_val[mask_tiles == tr_] = 1

for val_ in tiles_val:
    mask_tr_val[mask_tiles == val_] = 2

mask_amazon_ts = np.zeros((mask_tiles.shape)).astype('float32')
```

```
for ts_ in tiles_ts:
    mask_amazon_ts[mask_tiles == ts_] = 1
```

In [8]:

```
# Create ixd image to extract patches
overlap = 0.7
patch_size = 128
batch_size = 32
im_idx = create_idx_image(final_mask1)
patches_idx = extract_patches(im_idx, patch_size=(patch_size, patch_size), overlap=o
patches_mask = extract_patches(mask_tr_val, patch_size=(patch_size, patch_size), ove
del im_idx
```

In [9]:

```
# Selecting index trn val and test patches idx
idx_trn = np.squeeze(np.where(patches_mask.sum(axis=(1, 2))==patch_size**2))
idx_val = np.squeeze(np.where(patches_mask.sum(axis=(1, 2))==2*patch_size**2))
del patches_mask

patches_idx_trn = patches_idx[idx_trn]
patches_idx_val = patches_idx[idx_val]
del idx_trn, idx_val

print('Number of training patches:  ', len(patches_idx_trn), 'Number of validation p
```

```
Number of training patches:   17110 Number of validation patches 4116
```

In [10]:

```
# Extract patches with at least 2% of deforestation class
X_train = retrieve_idx_percentage(final_mask1, patches_idx_trn, patch_size, pertenta
X_valid = retrieve_idx_percentage(final_mask1, patches_idx_val, patch_size, pertenta
print(X_train.shape, X_valid.shape)
del patches_idx_trn, patches_idx_val
```

```
(1158, 128, 128) (341, 128, 128)
```

In [11]:

```
def batch_generator(batches, image, reference, target_size, number_class):
    """Take as input a Keras ImageGen (Iterator) and generate random
    crops from the image batches generated by the original iterator.
    """
    image = image.reshape(-1, image.shape[-1])
    reference = reference.reshape(final_mask1.shape[0]*final_mask1.shape[1])
    while True:
        batch_x, batch_y = next(batches)
        batch_x = np.squeeze(batch_x.astype('int64'))
        #print(batch_x.shape)
        batch_img = np.zeros((batch_x.shape[0], target_size, target_size, image.shap
        batch_ref = np.zeros((batch_x.shape[0], target_size, target_size, number_cla

        for i in range(batch_x.shape[0]):
            if np.random.rand()>0.5:
                batch_x[i] = np.rot90(batch_x[i], 1)
            batch_img[i] = image[batch_x[i]]
            batch_ref[i] = tf.keras.utils.to_categorical(reference[batch_x[i]] , num

        yield (batch_img, batch_ref)

train_datagen = ImageDataGenerator(horizontal_flip = True,
                                   vertical_flip = True)
valid_datagen = ImageDataGenerator(horizontal_flip = True,
                                   vertical_flip = True)

y_train = np.zeros((len(X_train)))
y_valid = np.zeros((len(X_valid)))
```

```
train_gen = train_datagen.flow(np.expand_dims(X_train, axis = -1), y_train,
                               batch_size=batch_size,
                               shuffle=True)

valid_gen = valid_datagen.flow(np.expand_dims(X_valid, axis = -1), y_valid,
                               batch_size=batch_size,
                               shuffle=False)

number_class = 3
train_gen_crops = batch_generator(train_gen, image_array, final_mask1, patch_size, n
valid_gen_crops = batch_generator(valid_gen, image_array, final_mask1, patch_size, n
```

In [26]:

```
exp = 1
path_exp = root_path+'experiments/exp'+str(exp)
path_models = path_exp+'/models'
path_maps = path_exp+'/pred_maps'

if not os.path.exists(path_exp):
    os.makedirs(path_exp)
if not os.path.exists(path_models):
    os.makedirs(path_models)
if not os.path.exists(path_maps):
    os.makedirs(path_maps)
```

In [41]:

```
# Define model
input_shape = (patch_size, patch_size, channels)
nb_filters = [32, 64, 128]

method = 'unet'
if method == 'unet':
   model = build_unet(input_shape, nb_filters, number_class)

if method == 'resunet':
   model = build_resunet(input_shape, nb_filters, number_class)

model = Model_3(nb_filters, number_class)
model.build((None, 128,128,8))
```

In [42]:

```
# Parameters of the model
weights = [0.2, 0.8, 0]
adam = Adam(lr = 1e-3 , beta_1=0.9)

loss = weighted_categorical_crossentropy(weights)
#loss = WBCE(weights)
```

In [43]:

```
time_tr = []
times = 5
for tm in range(0,times):
    print('time: ', tm)
    model = Model_3(nb_filters, number_class)
    model.build((None, 128,128,8))
    model.compile(optimizer=adam, loss=loss, metrics=['accuracy'])
    model.summary()

    earlystop = EarlyStopping(monitor='val_loss', min_delta=0.0001, patience=10, ver
    checkpoint = ModelCheckpoint(path_models+ '/' + method +'_'+str(tm)+'.h5', save_
    #checkpoint = ModelCheckpoint(path_models+ '/' + method +'_'+str(tm)+'.h5', moni
    lr_reduce = ReduceLROnPlateau(factor=0.9, min_delta=0.0001, patience=5, verbose=
```

```
    callbacks_list = [earlystop, checkpoint]
    # train the model
    start_training = time.time()
    history = model.fit_generator(train_gen_crops,
                                  steps_per_epoch=len(X_train)*3//train_gen.batch_size,
                                  validation_data=valid_gen_crops,
                                  validation_steps=len(X_valid)*3//valid_gen.batch_size,
                                  epochs=100,
                                  callbacks=callbacks_list)
    end_training = time.time() - start_training
    time_tr.append(end_training)
time_tr_array = np.asarray(time_tr)
# Save training time
np.save(path_exp+'/metrics_tr.npy', time_tr_array)
```

```
time:  0
Model: "model_3_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv1 (Conv2D)               multiple                  2336      
_________________________________________________________________
conv2 (Conv2D)               multiple                  18496     
_________________________________________________________________
conv3 (Conv2D)               multiple                  73856     
_________________________________________________________________
maxPool1 (MaxPooling2D)      multiple                  0         
_________________________________________________________________
maxPool2 (MaxPooling2D)      multiple                  0         
_________________________________________________________________
maxPool3 (MaxPooling2D)      multiple                  0         
_________________________________________________________________
conv4 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv5 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv6 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv7 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv8 (Conv2D)               multiple                  147520    
_________________________________________________________________
conv9 (Conv2D)               multiple                  36896     
_________________________________________________________________
upSamp1 (UpSampling2D)       multiple                  0         
_________________________________________________________________
upSamp2 (UpSampling2D)       multiple                  0         
_________________________________________________________________
upSamp3 (UpSampling2D)       multiple                  0         
_________________________________________________________________
conv2d_12 (Conv2D)           multiple                  195       
=================================================================
Total params: 869,635
Trainable params: 869,635
Non-trainable params: 0
_________________________________________________________________
C:\Users\felferrari\AppData\Roaming\Python\Python38\site-packages\tensorflow\python
\keras\engine\training.py:1844: UserWarning: `Model.fit_generator` is deprecated and
will be removed in a future version. Please use `Model.fit`, which supports generato
rs.
  warnings.warn('`Model.fit_generator` is deprecated and '
Epoch 1/100
108/108 [==============================] - 18s 163ms/step - loss: 0.0929 - accuracy:
0.7678 - val_loss: 0.1035 - val_accuracy: 0.7718
```

```
Epoch 00001: val_loss improved from inf to 0.10347, saving model to imgs/experiment
s/exp1/models\unet_0.h5
Epoch 2/100
108/108 [==============================] - 17s 159ms/step - loss: 0.0642 - accuracy:
0.8275 - val_loss: 0.0977 - val_accuracy: 0.7896

Epoch 00002: val_loss improved from 0.10347 to 0.09775, saving model to imgs/experim
ents/exp1/models\unet_0.h5
Epoch 3/100
108/108 [==============================] - 17s 163ms/step - loss: 0.0594 - accuracy:
0.8390 - val_loss: 0.1186 - val_accuracy: 0.7669

Epoch 00003: val_loss did not improve from 0.09775
Epoch 4/100
108/108 [==============================] - 17s 160ms/step - loss: 0.0586 - accuracy:
0.8396 - val_loss: 0.1251 - val_accuracy: 0.7853

Epoch 00004: val_loss did not improve from 0.09775
Epoch 5/100
108/108 [==============================] - 17s 160ms/step - loss: 0.0550 - accuracy:
0.8463 - val_loss: 0.1375 - val_accuracy: 0.7911

Epoch 00005: val_loss did not improve from 0.09775
Epoch 6/100
108/108 [==============================] - 17s 157ms/step - loss: 0.0527 - accuracy:
0.8520 - val_loss: 0.1224 - val_accuracy: 0.7862

Epoch 00006: val_loss did not improve from 0.09775
Epoch 7/100
108/108 [==============================] - 17s 157ms/step - loss: 0.0477 - accuracy:
0.8606 - val_loss: 0.1757 - val_accuracy: 0.7914

Epoch 00007: val_loss did not improve from 0.09775
Epoch 8/100
108/108 [==============================] - 17s 160ms/step - loss: 0.0433 - accuracy:
0.8682 - val_loss: 0.1680 - val_accuracy: 0.7907

Epoch 00008: val_loss did not improve from 0.09775
Epoch 9/100
108/108 [==============================] - 17s 159ms/step - loss: 0.0399 - accuracy:
0.8748 - val_loss: 0.1935 - val_accuracy: 0.7830

Epoch 00009: val_loss did not improve from 0.09775
Epoch 10/100
108/108 [==============================] - 17s 158ms/step - loss: 0.0380 - accuracy:
0.8777 - val_loss: 0.1846 - val_accuracy: 0.7922

Epoch 00010: val_loss did not improve from 0.09775
Epoch 11/100
108/108 [==============================] - 17s 159ms/step - loss: 0.0347 - accuracy:
0.8824 - val_loss: 0.1681 - val_accuracy: 0.7829

Epoch 00011: val_loss did not improve from 0.09775
Epoch 12/100
108/108 [==============================] - 17s 162ms/step - loss: 0.0322 - accuracy:
0.8895 - val_loss: 0.1657 - val_accuracy: 0.7883

Epoch 00012: val_loss did not improve from 0.09775
Epoch 00012: early stopping
time:  1
Model: "model_3_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
```

```
conv1 (Conv2D)               multiple                  2336      
_________________________________________________________________
conv2 (Conv2D)               multiple                  18496     
_________________________________________________________________
conv3 (Conv2D)               multiple                  73856     
_________________________________________________________________
maxPool1 (MaxPooling2D)      multiple                  0         
_________________________________________________________________
maxPool2 (MaxPooling2D)      multiple                  0         
_________________________________________________________________
maxPool3 (MaxPooling2D)      multiple                  0         
_________________________________________________________________
conv4 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv5 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv6 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv7 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv8 (Conv2D)               multiple                  147520    
_________________________________________________________________
conv9 (Conv2D)               multiple                  36896     
_________________________________________________________________
upSamp1 (UpSampling2D)       multiple                  0         
_________________________________________________________________
upSamp2 (UpSampling2D)       multiple                  0         
_________________________________________________________________
upSamp3 (UpSampling2D)       multiple                  0         
_________________________________________________________________
conv2d_13 (Conv2D)           multiple                  195       
=================================================================
Total params: 869,635
Trainable params: 869,635
Non-trainable params: 0
_________________________________________________________________
Epoch 1/100
108/108 [==============================] - 18s 158ms/step - loss: 0.0986 - accuracy: 
0.7458 - val_loss: 0.0961 - val_accuracy: 0.7815

Epoch 00001: val_loss improved from inf to 0.09608, saving model to imgs/experiment
s/exp1/models\unet_1.h5
Epoch 2/100
108/108 [==============================] - 17s 159ms/step - loss: 0.0661 - accuracy: 
0.8222 - val_loss: 0.1043 - val_accuracy: 0.7750

Epoch 00002: val_loss did not improve from 0.09608
Epoch 3/100
108/108 [==============================] - 17s 159ms/step - loss: 0.0631 - accuracy: 
0.8292 - val_loss: 0.1040 - val_accuracy: 0.7826

Epoch 00003: val_loss did not improve from 0.09608
Epoch 4/100
108/108 [==============================] - 17s 156ms/step - loss: 0.0601 - accuracy: 
0.8345 - val_loss: 0.0978 - val_accuracy: 0.7951

Epoch 00004: val_loss did not improve from 0.09608
Epoch 5/100
108/108 [==============================] - 17s 158ms/step - loss: 0.0577 - accuracy: 
0.8399 - val_loss: 0.1082 - val_accuracy: 0.7911

Epoch 00005: val_loss did not improve from 0.09608
Epoch 6/100
108/108 [==============================] - 18s 165ms/step - loss: 0.0558 - accuracy: 
0.8443 - val_loss: 0.1106 - val_accuracy: 0.7779
```

```
Epoch 00006: val_loss did not improve from 0.09608
Epoch 7/100
108/108 [==============================] - 17s 160ms/step - loss: 0.0527 - accuracy:
0.8506 - val_loss: 0.1036 - val_accuracy: 0.7944

Epoch 00007: val_loss did not improve from 0.09608
Epoch 8/100
108/108 [==============================] - 17s 160ms/step - loss: 0.0500 - accuracy:
0.8552 - val_loss: 0.1105 - val_accuracy: 0.7943

Epoch 00008: val_loss did not improve from 0.09608
Epoch 9/100
108/108 [==============================] - 17s 157ms/step - loss: 0.0479 - accuracy:
0.8593 - val_loss: 0.0977 - val_accuracy: 0.7970

Epoch 00009: val_loss did not improve from 0.09608
Epoch 10/100
108/108 [==============================] - 17s 159ms/step - loss: 0.0464 - accuracy:
0.8603 - val_loss: 0.1091 - val_accuracy: 0.8004

Epoch 00010: val_loss did not improve from 0.09608
Epoch 11/100
108/108 [==============================] - 17s 157ms/step - loss: 0.0432 - accuracy:
0.8670 - val_loss: 0.1056 - val_accuracy: 0.7898

Epoch 00011: val_loss did not improve from 0.09608
Epoch 00011: early stopping
time:  2
Model: "model_3_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv1 (Conv2D)               multiple                  2336      
_________________________________________________________________
conv2 (Conv2D)               multiple                  18496     
_________________________________________________________________
conv3 (Conv2D)               multiple                  73856     
_________________________________________________________________
maxPool1 (MaxPooling2D)      multiple                  0         
_________________________________________________________________
maxPool2 (MaxPooling2D)      multiple                  0         
_________________________________________________________________
maxPool3 (MaxPooling2D)      multiple                  0         
_________________________________________________________________
conv4 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv5 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv6 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv7 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv8 (Conv2D)               multiple                  147520    
_________________________________________________________________
conv9 (Conv2D)               multiple                  36896     
_________________________________________________________________
upSamp1 (UpSampling2D)       multiple                  0         
_________________________________________________________________
upSamp2 (UpSampling2D)       multiple                  0         
_________________________________________________________________
upSamp3 (UpSampling2D)       multiple                  0         
_________________________________________________________________
conv2d_14 (Conv2D)           multiple                  195       
=================================================================
```

```
Total params: 869,635
Trainable params: 869,635
Non-trainable params: 0
_________________________________________________________________
Epoch 1/100
108/108 [==============================] - 23s 207ms/step - loss: 0.1219 - accuracy: 0.7283 - val_loss: 0.0905 - val_accuracy: 0.7596

Epoch 00001: val_loss improved from inf to 0.09046, saving model to imgs/experiments/exp1/models\unet_2.h5
Epoch 2/100
108/108 [==============================] - 18s 167ms/step - loss: 0.0672 - accuracy: 0.8191 - val_loss: 0.1032 - val_accuracy: 0.7693

Epoch 00002: val_loss did not improve from 0.09046
Epoch 3/100
108/108 [==============================] - 18s 167ms/step - loss: 0.0635 - accuracy: 0.8261 - val_loss: 0.1003 - val_accuracy: 0.7776

Epoch 00003: val_loss did not improve from 0.09046
Epoch 4/100
108/108 [==============================] - 18s 164ms/step - loss: 0.0602 - accuracy: 0.8335 - val_loss: 0.1023 - val_accuracy: 0.7698

Epoch 00004: val_loss did not improve from 0.09046
Epoch 5/100
108/108 [==============================] - 18s 166ms/step - loss: 0.0583 - accuracy: 0.8374 - val_loss: 0.1016 - val_accuracy: 0.7716

Epoch 00005: val_loss did not improve from 0.09046
Epoch 6/100
108/108 [==============================] - 18s 170ms/step - loss: 0.0576 - accuracy: 0.8379 - val_loss: 0.1132 - val_accuracy: 0.7664

Epoch 00006: val_loss did not improve from 0.09046
Epoch 7/100
108/108 [==============================] - 18s 168ms/step - loss: 0.0557 - accuracy: 0.8415 - val_loss: 0.1043 - val_accuracy: 0.7933

Epoch 00007: val_loss did not improve from 0.09046
Epoch 8/100
108/108 [==============================] - 18s 169ms/step - loss: 0.0540 - accuracy: 0.8443 - val_loss: 0.1085 - val_accuracy: 0.7830

Epoch 00008: val_loss did not improve from 0.09046
Epoch 9/100
108/108 [==============================] - 18s 165ms/step - loss: 0.0512 - accuracy: 0.8490 - val_loss: 0.1081 - val_accuracy: 0.7866

Epoch 00009: val_loss did not improve from 0.09046
Epoch 10/100
108/108 [==============================] - 18s 166ms/step - loss: 0.0509 - accuracy: 0.8512 - val_loss: 0.1267 - val_accuracy: 0.7976

Epoch 00010: val_loss did not improve from 0.09046
Epoch 11/100
108/108 [==============================] - 18s 166ms/step - loss: 0.0487 - accuracy: 0.8544 - val_loss: 0.1024 - val_accuracy: 0.7869

Epoch 00011: val_loss did not improve from 0.09046
Epoch 00011: early stopping
time:  3
Model: "model_3_5"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
```

```
=================================================================
conv1 (Conv2D)               multiple                  2336      
_________________________________________________________________
conv2 (Conv2D)               multiple                  18496     
_________________________________________________________________
conv3 (Conv2D)               multiple                  73856     
_________________________________________________________________
maxPool1 (MaxPooling2D)      multiple                  0         
_________________________________________________________________
maxPool2 (MaxPooling2D)      multiple                  0         
_________________________________________________________________
maxPool3 (MaxPooling2D)      multiple                  0         
_________________________________________________________________
conv4 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv5 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv6 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv7 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv8 (Conv2D)               multiple                  147520    
_________________________________________________________________
conv9 (Conv2D)               multiple                  36896     
_________________________________________________________________
upSamp1 (UpSampling2D)       multiple                  0         
_________________________________________________________________
upSamp2 (UpSampling2D)       multiple                  0         
_________________________________________________________________
upSamp3 (UpSampling2D)       multiple                  0         
_________________________________________________________________
conv2d_15 (Conv2D)           multiple                  195       
=================================================================
Total params: 869,635
Trainable params: 869,635
Non-trainable params: 0
_________________________________________________________________
Epoch 1/100
108/108 [==============================] - 19s 168ms/step - loss: 0.1198 - accuracy: 0.7419 - val_loss: 0.1020 - val_accuracy: 0.7645

Epoch 00001: val_loss improved from inf to 0.10196, saving model to imgs/experiments/exp1/models\unet_3.h5
Epoch 2/100
108/108 [==============================] - 18s 166ms/step - loss: 0.0716 - accuracy: 0.8129 - val_loss: 0.1063 - val_accuracy: 0.7652

Epoch 00002: val_loss did not improve from 0.10196
Epoch 3/100
108/108 [==============================] - 18s 166ms/step - loss: 0.0673 - accuracy: 0.8224 - val_loss: 0.1160 - val_accuracy: 0.7472

Epoch 00003: val_loss did not improve from 0.10196
Epoch 4/100
108/108 [==============================] - 18s 167ms/step - loss: 0.0638 - accuracy: 0.8276 - val_loss: 0.1125 - val_accuracy: 0.7743

Epoch 00004: val_loss did not improve from 0.10196
Epoch 5/100
108/108 [==============================] - 18s 170ms/step - loss: 0.0624 - accuracy: 0.8280 - val_loss: 0.1079 - val_accuracy: 0.7795

Epoch 00005: val_loss did not improve from 0.10196
Epoch 6/100
108/108 [==============================] - 18s 171ms/step - loss: 0.0591 - accuracy:
```

```
0.8359 - val_loss: 0.0948 - val_accuracy: 0.7862

Epoch 00006: val_loss improved from 0.10196 to 0.09477, saving model to imgs/experim
ents/exp1/models\unet_3.h5
Epoch 7/100
108/108 [==============================] - 18s 167ms/step - loss: 0.0592 - accuracy:
0.8347 - val_loss: 0.0890 - val_accuracy: 0.7874

Epoch 00007: val_loss improved from 0.09477 to 0.08896, saving model to imgs/experim
ents/exp1/models\unet_3.h5
Epoch 8/100
108/108 [==============================] - 19s 178ms/step - loss: 0.0567 - accuracy:
0.8403 - val_loss: 0.1221 - val_accuracy: 0.7707

Epoch 00008: val_loss did not improve from 0.08896
Epoch 9/100
108/108 [==============================] - 19s 175ms/step - loss: 0.0571 - accuracy:
0.8413 - val_loss: 0.1106 - val_accuracy: 0.7790

Epoch 00009: val_loss did not improve from 0.08896
Epoch 10/100
108/108 [==============================] - 19s 176ms/step - loss: 0.0542 - accuracy:
0.8456 - val_loss: 0.0928 - val_accuracy: 0.7916

Epoch 00010: val_loss did not improve from 0.08896
Epoch 11/100
108/108 [==============================] - 18s 172ms/step - loss: 0.0535 - accuracy:
0.8478 - val_loss: 0.1024 - val_accuracy: 0.7916

Epoch 00011: val_loss did not improve from 0.08896
Epoch 12/100
108/108 [==============================] - 18s 169ms/step - loss: 0.0507 - accuracy:
0.8516 - val_loss: 0.0952 - val_accuracy: 0.7918

Epoch 00012: val_loss did not improve from 0.08896
Epoch 13/100
108/108 [==============================] - 18s 167ms/step - loss: 0.0501 - accuracy:
0.8546 - val_loss: 0.0978 - val_accuracy: 0.7993

Epoch 00013: val_loss did not improve from 0.08896
Epoch 14/100
108/108 [==============================] - 18s 169ms/step - loss: 0.0490 - accuracy:
0.8533 - val_loss: 0.1275 - val_accuracy: 0.7869

Epoch 00014: val_loss did not improve from 0.08896
Epoch 15/100
108/108 [==============================] - 18s 172ms/step - loss: 0.0471 - accuracy:
0.8592 - val_loss: 0.1350 - val_accuracy: 0.7934

Epoch 00015: val_loss did not improve from 0.08896
Epoch 16/100
108/108 [==============================] - 19s 173ms/step - loss: 0.0454 - accuracy:
0.8624 - val_loss: 0.1133 - val_accuracy: 0.7916

Epoch 00016: val_loss did not improve from 0.08896
Epoch 17/100
108/108 [==============================] - 18s 172ms/step - loss: 0.0448 - accuracy:
0.8623 - val_loss: 0.1378 - val_accuracy: 0.7951

Epoch 00017: val_loss did not improve from 0.08896
Epoch 00017: early stopping
time:  4
Model: "model_3_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
```

```
=================================================================
conv1 (Conv2D)               multiple                  2336      
_________________________________________________________________
conv2 (Conv2D)               multiple                  18496     
_________________________________________________________________
conv3 (Conv2D)               multiple                  73856     
_________________________________________________________________
maxPool1 (MaxPooling2D)      multiple                  0         
_________________________________________________________________
maxPool2 (MaxPooling2D)      multiple                  0         
_________________________________________________________________
maxPool3 (MaxPooling2D)      multiple                  0         
_________________________________________________________________
conv4 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv5 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv6 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv7 (Conv2D)               multiple                  147584    
_________________________________________________________________
conv8 (Conv2D)               multiple                  147520    
_________________________________________________________________
conv9 (Conv2D)               multiple                  36896     
_________________________________________________________________
upSamp1 (UpSampling2D)       multiple                  0         
_________________________________________________________________
upSamp2 (UpSampling2D)       multiple                  0         
_________________________________________________________________
upSamp3 (UpSampling2D)       multiple                  0         
_________________________________________________________________
conv2d_16 (Conv2D)           multiple                  195       
=================================================================
Total params: 869,635
Trainable params: 869,635
Non-trainable params: 0
_________________________________________________________________
Epoch 1/100
108/108 [==============================] - 19s 172ms/step - loss: 0.1157 - accuracy: 
0.7222 - val_loss: 0.1044 - val_accuracy: 0.7686

Epoch 00001: val_loss improved from inf to 0.10442, saving model to imgs/experiment
s/exp1/models\unet_4.h5
Epoch 2/100
108/108 [==============================] - 18s 171ms/step - loss: 0.0694 - accuracy: 
0.8182 - val_loss: 0.1037 - val_accuracy: 0.7665

Epoch 00002: val_loss improved from 0.10442 to 0.10369, saving model to imgs/experim
ents/exp1/models\unet_4.h5
Epoch 3/100
108/108 [==============================] - 18s 169ms/step - loss: 0.0665 - accuracy: 
0.8257 - val_loss: 0.1035 - val_accuracy: 0.7729

Epoch 00003: val_loss improved from 0.10369 to 0.10349, saving model to imgs/experim
ents/exp1/models\unet_4.h5
Epoch 4/100
108/108 [==============================] - 18s 170ms/step - loss: 0.0619 - accuracy: 
0.8336 - val_loss: 0.1064 - val_accuracy: 0.7839

Epoch 00004: val_loss did not improve from 0.10349
Epoch 5/100
108/108 [==============================] - 18s 168ms/step - loss: 0.0596 - accuracy: 
0.8372 - val_loss: 0.0893 - val_accuracy: 0.7767

Epoch 00005: val_loss improved from 0.10349 to 0.08931, saving model to imgs/experim
```

```
ents/exp1/models\unet_4.h5
Epoch 6/100
108/108 [==============================] - 19s 174ms/step - loss: 0.0578 - accuracy:
0.8384 - val_loss: 0.0991 - val_accuracy: 0.7837

Epoch 00006: val_loss did not improve from 0.08931
Epoch 7/100
108/108 [==============================] - 19s 175ms/step - loss: 0.0550 - accuracy:
0.8450 - val_loss: 0.1142 - val_accuracy: 0.7639

Epoch 00007: val_loss did not improve from 0.08931
Epoch 8/100
108/108 [==============================] - 19s 173ms/step - loss: 0.0544 - accuracy:
0.8446 - val_loss: 0.1147 - val_accuracy: 0.7855

Epoch 00008: val_loss did not improve from 0.08931
Epoch 9/100
108/108 [==============================] - 18s 170ms/step - loss: 0.0512 - accuracy:
0.8528 - val_loss: 0.1339 - val_accuracy: 0.7773

Epoch 00009: val_loss did not improve from 0.08931
Epoch 10/100
108/108 [==============================] - 19s 174ms/step - loss: 0.0500 - accuracy:
0.8553 - val_loss: 0.1154 - val_accuracy: 0.7760

Epoch 00010: val_loss did not improve from 0.08931
Epoch 11/100
108/108 [==============================] - 18s 170ms/step - loss: 0.0493 - accuracy:
0.8559 - val_loss: 0.1419 - val_accuracy: 0.7919

Epoch 00011: val_loss did not improve from 0.08931
Epoch 12/100
108/108 [==============================] - 19s 174ms/step - loss: 0.0483 - accuracy:
0.8572 - val_loss: 0.1410 - val_accuracy: 0.7829

Epoch 00012: val_loss did not improve from 0.08931
Epoch 13/100
108/108 [==============================] - 18s 172ms/step - loss: 0.0466 - accuracy:
0.8619 - val_loss: 0.1522 - val_accuracy: 0.7706

Epoch 00013: val_loss did not improve from 0.08931
Epoch 14/100
108/108 [==============================] - 18s 172ms/step - loss: 0.0441 - accuracy:
0.8655 - val_loss: 0.1422 - val_accuracy: 0.7754

Epoch 00014: val_loss did not improve from 0.08931
Epoch 15/100
108/108 [==============================] - 18s 168ms/step - loss: 0.0442 - accuracy:
0.8660 - val_loss: 0.1507 - val_accuracy: 0.7925

Epoch 00015: val_loss did not improve from 0.08931
Epoch 00015: early stopping
```

In [44]:

```
# Test Loop
time_ts = []
n_pool = 3
n_rows = 5
n_cols = 4
rows, cols = image_array.shape[:2]
pad_rows = rows - np.ceil(rows/(n_rows*2**n_pool))*n_rows*2**n_pool
pad_cols = cols - np.ceil(cols/(n_cols*2**n_pool))*n_cols*2**n_pool
print(pad_rows, pad_cols)

npad = ((0, int(abs(pad_rows))), (0, int(abs(pad_cols))), (0, 0))
```

```
image1_pad = np.pad(image_array, pad_width=npad, mode='reflect')

h, w, c = image1_pad.shape
patch_size_rows = h//n_rows
patch_size_cols = w//n_cols
num_patches_x = int(h/patch_size_rows)
num_patches_y = int(w/patch_size_cols)

input_shape=(patch_size_rows,patch_size_cols, c)

if method == 'unet':
   new_model = build_unet(input_shape, nb_filters, number_class)

if method == 'resunet':
   new_model = build_resunet(input_shape, nb_filters, number_class)

new_model = Model_3(nb_filters, number_class)
new_model.build((None, 128,128,8))
new_model.compile(optimizer=adam, loss=loss, metrics=['accuracy'])

for tm in range(0,times):
    print('time: ', tm)
    #model = load_model(path_models+ '/' + method +'_'+str(tm)+'.h5', compile=False)

    #for l in range(1, len(model.layers)):
    #    new_model.layers[l].set_weights(model.layers[l].get_weights())
    new_model.load_weights(path_models+ '/' + method +'_'+str(tm)+'.h5')

    start_test = time.time()
    patch_t = []

    for i in range(0,num_patches_y):
        for j in range(0,num_patches_x):
            patch = image1_pad[patch_size_rows*j:patch_size_rows*(j+1), patch_size_c
            predictions_ = new_model.predict(np.expand_dims(patch, axis=0))
            del patch
            patch_t.append(predictions_[:,:,:,1])
            del predictions_
    end_test =  time.time() - start_test
    patches_pred = np.asarray(patch_t).astype(np.float32)

    prob_recontructed = pred_reconctruct(h, w, num_patches_x, num_patches_y, patch_s
    np.save(path_maps+'/'+'prob_'+str(tm)+'.npy',prob_recontructed)

    time_ts.append(end_test)
    del prob_recontructed, patches_pred
time_ts_array = np.asarray(time_ts)
del new_model
# Save test time
np.save(path_exp+'/metrics_ts.npy', time_ts_array)
```

```
0.0 -8.0
time:  0
time:  1
time:  2
time:  3
time:  4
```

In [45]:

```
# Compute mean of the tm predictions maps
prob_rec = np.zeros((image1_pad.shape[0],image1_pad.shape[1], times))

for tm in range (0, times):
    print(tm)
    prob_rec[:,:,tm] = np.load(path_maps+'/'+'prob_'+str(tm)+'.npy').astype(np.float
```

```
mean_prob = np.mean(prob_rec, axis = -1)
np.save(path_maps+'/prob_mean.npy', mean_prob)
```

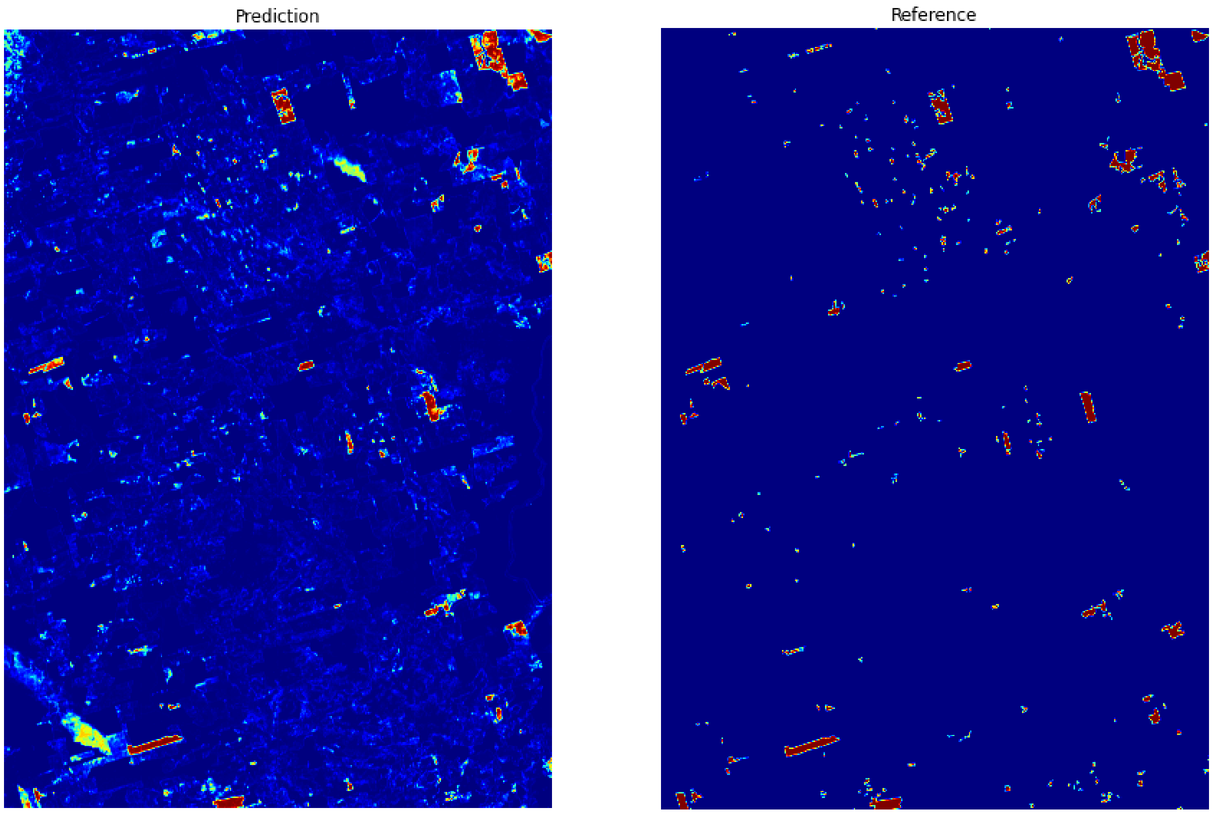
```
0
1
2
3
4
```

In [46]:

```
ref = final_mask1
ref[ref==0]=0
ref[ref==2]=0
# Plot mean map and reference
fig = plt.figure(figsize=(15,10))
ax1 = fig.add_subplot(121)
plt.title('Prediction')
ax1.imshow(mean_prob, cmap ='jet')
ax1.axis('off')

ax2 = fig.add_subplot(122)
plt.title('Reference')
ax2.imshow(ref, cmap ='jet')
ax2.axis('off')
```

Out[46]:

```
(-0.5, 6999.5, 9999.5, -0.5)
```



In [47]:

```
# Computing metrics
mean_prob = mean_prob[:final_mask1.shape[0], :final_mask1.shape[1]]
ref1 = np.ones_like(final_mask1).astype(np.float32)

ref1 [final_mask1 == 2] = 0
TileMask = mask_amazon_ts * ref1
GTTruePositives = final_mask1==1

Npoints = 50
Pmax = np.max(mean_prob[GTTruePositives * TileMask ==1])
```

```
ProbList = np.linspace(Pmax,0,Npoints)

metrics_ = matrics_AA_recall(ProbList, mean_prob, final_mask1, mask_amazon_ts, 625)
np.save(path_exp+'/acc_metrics.npy',metrics_)
```

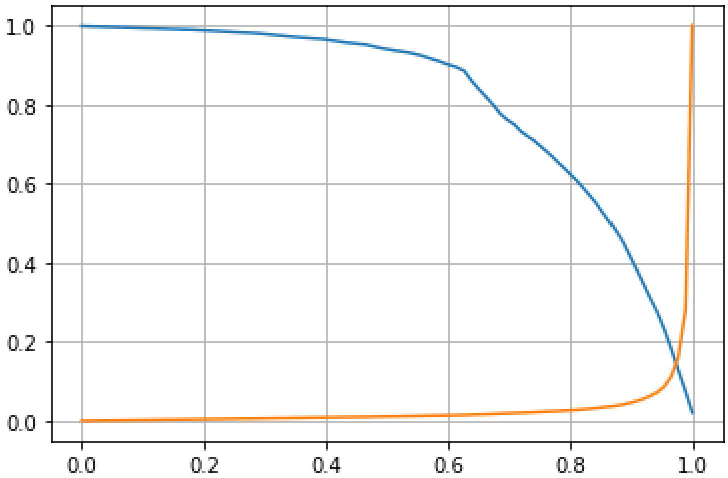
```
0.996409285068512
D:\Ferrari\proj_1\projeto\utils_unet_resunet.py:200: RuntimeWarning: invalid value e
ncountered in longlong_scalars
  precision_ = TP/(TP+FP)
0.9760744016997669
0.9557395183310218
0.9354046349622765
0.9150697515935314
0.8947348682247863
0.8743999848560411
0.854065101487296
0.8337302181185509
0.8133953347498057
0.7930604513810606
0.7727255680123154
0.7523906846435703
0.7320558012748252
0.7117209179060799
0.6913860345373348
0.6710511511685897
0.6507162677998446
0.6303813844310995
0.6100465010623543
0.5897116176936091
0.569376734324864
0.5490418509561188
0.5287069675873737
0.5083720842186286
0.48803720084988345
0.46770231748113833
0.4473674341123931
0.427032550743648
0.40669766737490287
0.38636278400615764
0.36602790063741253
0.3456930172686674
0.3253581338999223
0.3050232505311772
0.28468836716243195
0.26435348379368684
0.24401860042494172
0.2236837170561965
0.20334883368745138
0.18301395031870626
0.16267906694996115
0.14234418358121603
0.1220093002124708
0.10167441684372569
0.08133953347498057
0.06100465010623535
0.04066976673749023
0.020334883368745116
0.0
```

In [48]:

```
# Complete NaN values
metrics_copy = metrics_.copy()
metrics_copy = complete_nan_values(metrics_copy)
```

In [49]:
```
# Comput Mean Average Precision (mAP) score
Recall = metrics_copy[:,0]
Precision = metrics_copy[:,1]
AA = metrics_copy[:,2]

DeltaR = Recall[1:]-Recall[:-1]
AP = np.sum(Precision[:-1]*DeltaR)
print('mAP', AP)

# Plot Recall vs. Precision curve
plt.close('all')
plt.plot(metrics_copy[:,0],metrics_copy[:,1])
plt.plot(metrics_copy[:,0],metrics_copy[:,2])
plt.grid()
```

```
mAP 0.8192160883357321
```



In [ ]: