Instructions

- Some parts of the code are already done for you
- You need to execute all the cells
- You need to add the code where ever you see `"#### Add your code here ####"`
- Marks are mentioned along with the cells

Face detection

Task is to predict the boundaries(mask) around the face in a given image.

Dataset

Faces in images marked with bounding boxes. Have around 500 images with around 1100 faces mar

Mount Google drive if you are using google colab

- We recommend using Google Colab as you can face memory issues and longer runtimes while

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318

Enter your authorization code:
··········
Mounted at /content/drive
```

Change current working directory to project folder (1 mark)

```
import os
import tensorflow as tf
#### Add your code here ####
project_dir = "/content/drive/My Drive/greatlakes/Projects/Advanced_Computer_Vision/Project1/
os.chdir(project_dir)
```

The default version of TensorFlow in Colab will soon switch to TensorFlow 2.x.
We recommend you upgrade now or ensure your notebook will continue to use TensorFlow 1.x via the `%tens` `1.x` magic: more info.

Load the "images.npy" file (2 marks)

- This file contains images with details of bounding boxes

```
import numpy as np
data = np.load('images.npy', allow_pickle=True)
```

Check one sample from the loaded "images.npy" file (2 marks)

Hint - print data[10][1]

```
#### Add your code here ####
print(data[10][1])
```

```
[{'label': ['Face'], 'notes': '', 'points': [{'x': 0.48, 'y': 0.10385756676557864}, {'x'
```

Set image dimensions (1 mark)

- Initialize image height, image width with value: 224

```
IMAGE_WIDTH = 224
IMAGE_HEIGHT = 224
```

Create features and labels

- Here feature is the image
- The label is the mask
- Images will be stored in "X_train" array
- Masks will be stored in "masks" array

```
import cv2
from tensorflow.keras.applications.mobilenet import preprocess_input

masks = np.zeros((int(data.shape[0]), IMAGE_HEIGHT, IMAGE_WIDTH))
X_train = np.zeros((int(data.shape[0]), IMAGE_HEIGHT, IMAGE_WIDTH, 3))
for index in range(data.shape[0]):
    img = data[index][0]
    img = cv2.resize(img, dsize=(IMAGE_HEIGHT, IMAGE_WIDTH), interpolation=cv2.INTER_CUBIC)
    try:
      img = img[:, :, :3]
    except:
      continue
    X_train[index] = preprocess_input(np.array(img, dtype=np.float32))
    for i in data[index][1]:
        x1 = int(i["points"][0]['x'] * IMAGE_WIDTH)
        x2 = int(i["points"][1]['x'] * IMAGE_WIDTH)
```

```
        x2 = int(i["points"][1]['x'] * IMAGE_WIDTH)
        y1 = int(i["points"][0]['y'] * IMAGE_HEIGHT)
        y2 = int(i["points"][1]['y'] * IMAGE_HEIGHT)
        masks[index][y1:y2, x1:x2] = 1
```

Print the shape of X_train and mask array (1 mark)

```
X_train.shape
```

```
(409, 224, 224, 3)
```

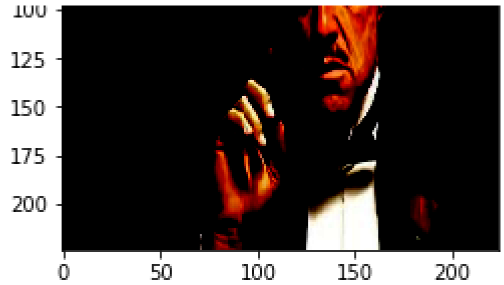
```
masks.shape
```

```
(409, 224, 224)
```

Print a sample image and image array

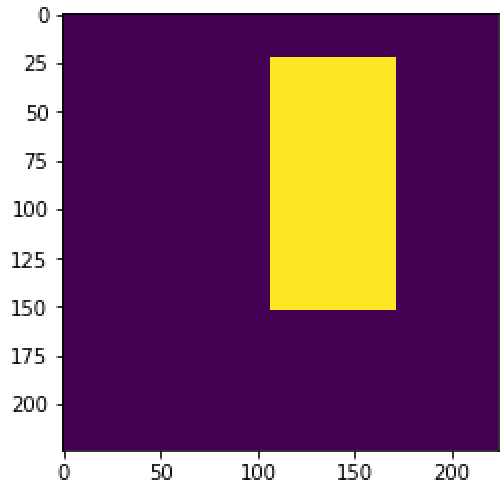
```
from matplotlib import pyplot
n = 10
print(X_train[n])
pyplot.imshow(X_train[n])
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0
[[[-0.98431373 -0.98431373 -0.98431373]
  [-0.98431373 -0.98431373 -0.98431373]
  [-0.98431373 -0.98431373 -0.98431373]
  ...
  [-1.         -1.         -1.        ]
  [-1.         -1.         -1.        ]
  [-1.         -1.         -1.        ]]

 [[-0.98431373 -0.98431373 -0.98431373]
  [-0.98431373 -0.98431373 -0.98431373]
  [-0.98431373 -0.98431373 -0.98431373]
  ...
  [-1.         -1.         -1.        ]
  [-1.         -1.         -1.        ]
  [-1.         -1.         -1.        ]]

 [[-0.98431373 -0.98431373 -0.98431373]
  [-0.98431373 -0.98431373 -0.98431373]
  [-0.98431373 -0.98431373 -0.98431373]
  ...
  [-1.         -1.         -1.        ]
  [-1.         -1.         -1.        ]
  [-1.         -1.         -1.        ]]

 ...

 [[-1.         -1.         -1.        ]
  [-1.         -1.         -1.        ]
  [-1.         -1.         -1.        ]
  ...
  [-0.96862745 -0.96862745 -0.96862745]
  [-0.96078432 -0.96078432 -0.96078432]
  [-0.96078432 -0.96078432 -0.96078432]]

 [[-1.         -1.         -1.        ]
  [-1.         -1.         -1.        ]
  [-1.         -1.         -1.        ]
  ...
  [-0.96862745 -0.96862745 -0.96862745]
  [-0.96078432 -0.96078432 -0.96078432]
  [-0.95294118 -0.95294118 -0.95294118]]

 [[-1.         -1.         -1.        ]
  [-1.         -1.         -1.        ]
  [-1.         -1.         -1.        ]
  ...
  [-0.97647059 -0.97647059 -0.97647059]
  [-0.96862745 -0.96862745 -0.96862745]
  [-0.96078432 -0.96078432 -0.96078432]]]
<matplotlib.image.AxesImage at 0x7f69f3cef5f8>
```



```
pyplot.imshow(masks[n])
```

```
<matplotlib.image.AxesImage at 0x7f69fcf48f60>
```



Create the model (10 marks)

- Add MobileNet as model with below parameter values
  - input_shape: IMAGE_HEIGHT, IMAGE_WIDTH, 3
  - include_top: False
  - alpha: 1.0
  - weights: "imagenet"
- Add UNET architecture layers
  - This is the trickiest part of the project, you need to research and implement it correctly

```
from tensorflow.keras.applications.mobilenet import MobileNet
from tensorflow.keras.layers import Concatenate, UpSampling2D, Conv2D, Reshape
from tensorflow.keras.models import Model

ALPHA = 1.0 # Width hyper parameter for MobileNet (0.25, 0.5, 0.75, 1.0). Higher width means

def create_model(trainable=True):
    model = model = MobileNet(input_shape=(IMAGE_WIDTH, IMAGE_HEIGHT, 3), include_top=False,
    for layer in model.layers:
        layer.trainable = trainable
```

```
    # Add all the UNET layers here

    block1 = model.get_layer("conv_pw_1_relu").output
    block2 = model.get_layer("conv_pw_3_relu").output
    block3 = model.get_layer("conv_pw_5_relu").output
    block4 = model.get_layer("conv_pw_11_relu").output
    block5 = model.get_layer("conv_pw_13_relu").output

    x = Concatenate()([UpSampling2D()(block5), block4])
    x = Concatenate()([UpSampling2D()(x), block3])
    x = Concatenate()([UpSampling2D()(x), block2])
    x = Concatenate()([UpSampling2D()(x), block1])
    x = UpSampling2D()(x)

    x = Conv2D(1, kernel_size=1, activation="sigmoid")(x)
    x = Reshape((IMAGE_WIDTH, IMAGE_HEIGHT))(x)

    return Model(inputs=model.input, outputs=x) #### Add your code here ####
```

Call the create_model function

```
# Give trainable=False as argument, if you want to freeze lower layers for fast training (but
model = create_model()

# Print summary
model.summary()
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_core/python/op
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
Downloading data from https://github.com/fchollet/deep-learning-models/releases/downloac
17227776/17225924 [==============================] - 0s 0us/step
Model: "model"
```

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 224, 224, 3)	0	
conv1_pad (ZeroPadding2D)	(None, 225, 225, 3)	0	input_1[0][0]
conv1 (Conv2D)	(None, 112, 112, 32)	864	conv1_pad[0][0]
conv1_bn (BatchNormalization)	(None, 112, 112, 32)	128	conv1[0][0]
conv1_relu (ReLU)	(None, 112, 112, 32)	0	conv1_bn[0][0]
conv_dw_1 (DepthwiseConv2D)	(None, 112, 112, 32)	288	conv1_relu[0][0]
conv_dw_1_bn (BatchNormalizatio	(None, 112, 112, 32)	128	conv_dw_1[0][0]
conv_dw_1_relu (ReLU)	(None, 112, 112, 32)	0	conv_dw_1_bn[0][0]
conv_pw_1 (Conv2D)	(None, 112, 112, 64)	2048	conv_dw_1_relu[0][0]
conv_pw_1_bn (BatchNormalizatio	(None, 112, 112, 64)	256	conv_pw_1[0][0]
conv_pw_1_relu (ReLU)	(None, 112, 112, 64)	0	conv_pw_1_bn[0][0]
conv_pad_2 (ZeroPadding2D)	(None, 113, 113, 64)	0	conv_pw_1_relu[0][0]
conv_dw_2 (DepthwiseConv2D)	(None, 56, 56, 64)	576	conv_pad_2[0][0]
conv_dw_2_bn (BatchNormalizatio	(None, 56, 56, 64)	256	conv_dw_2[0][0]
conv_dw_2_relu (ReLU)	(None, 56, 56, 64)	0	conv_dw_2_bn[0][0]
conv_pw_2 (Conv2D)	(None, 56, 56, 128)	8192	conv_dw_2_relu[0][0]
conv_pw_2_bn (BatchNormalizatio	(None, 56, 56, 128)	512	conv_pw_2[0][0]
conv_pw_2_relu (ReLU)	(None, 56, 56, 128)	0	conv_pw_2_bn[0][0]
conv_dw_3 (DepthwiseConv2D)	(None, 56, 56, 128)	1152	conv_pw_2_relu[0][0]
conv_dw_3_bn (BatchNormalizatio	(None, 56, 56, 128)	512	conv_dw_3[0][0]
conv_dw_3_relu (ReLU)	(None, 56, 56, 128)	0	conv_dw_3_bn[0][0]
conv_pw_3 (Conv2D)	(None, 56, 56, 128)	16384	conv_dw_3_relu[0][0]
conv_pw_3_bn (BatchNormalizatio	(None, 56, 56, 128)	512	conv_pw_3[0][0]
conv_pw_3_relu (ReLU)	(None, 56, 56, 128)	0	conv_pw_3_bn[0][0]

```
conv_pad_4 (ZeroPadding2D)      (None, 57, 57, 128)  0           conv_pw_3_relu[0][0]
________________________________________________________________________________________
conv_dw_4 (DepthwiseConv2D)     (None, 28, 28, 128)  1152        conv_pad_4[0][0]
________________________________________________________________________________________
conv_dw_4_bn (BatchNormalizatio (None, 28, 28, 128)  512         conv_dw_4[0][0]
________________________________________________________________________________________
conv_dw_4_relu (ReLU)           (None, 28, 28, 128)  0           conv_dw_4_bn[0][0]
________________________________________________________________________________________
conv_pw_4 (Conv2D)              (None, 28, 28, 256)  32768       conv_dw_4_relu[0][0]
________________________________________________________________________________________
conv_pw_4_bn (BatchNormalizatio (None, 28, 28, 256)  1024        conv_pw_4[0][0]
________________________________________________________________________________________
conv_pw_4_relu (ReLU)           (None, 28, 28, 256)  0           conv_pw_4_bn[0][0]
________________________________________________________________________________________
conv_dw_5 (DepthwiseConv2D)     (None, 28, 28, 256)  2304        conv_pw_4_relu[0][0]
________________________________________________________________________________________
conv_dw_5_bn (BatchNormalizatio (None, 28, 28, 256)  1024        conv_dw_5[0][0]
________________________________________________________________________________________
conv_dw_5_relu (ReLU)           (None, 28, 28, 256)  0           conv_dw_5_bn[0][0]
________________________________________________________________________________________
conv_pw_5 (Conv2D)              (None, 28, 28, 256)  65536       conv_dw_5_relu[0][0]
________________________________________________________________________________________
conv_pw_5_bn (BatchNormalizatio (None, 28, 28, 256)  1024        conv_pw_5[0][0]
________________________________________________________________________________________
conv_pw_5_relu (ReLU)           (None, 28, 28, 256)  0           conv_pw_5_bn[0][0]
________________________________________________________________________________________
conv_pad_6 (ZeroPadding2D)      (None, 29, 29, 256)  0           conv_pw_5_relu[0][0]
________________________________________________________________________________________
conv_dw_6 (DepthwiseConv2D)     (None, 14, 14, 256)  2304        conv_pad_6[0][0]
________________________________________________________________________________________
conv_dw_6_bn (BatchNormalizatio (None, 14, 14, 256)  1024        conv_dw_6[0][0]
________________________________________________________________________________________
conv_dw_6_relu (ReLU)           (None, 14, 14, 256)  0           conv_dw_6_bn[0][0]
________________________________________________________________________________________
conv_pw_6 (Conv2D)              (None, 14, 14, 512)  131072      conv_dw_6_relu[0][0]
________________________________________________________________________________________
conv_pw_6_bn (BatchNormalizatio (None, 14, 14, 512)  2048        conv_pw_6[0][0]
________________________________________________________________________________________
conv_pw_6_relu (ReLU)           (None, 14, 14, 512)  0           conv_pw_6_bn[0][0]
________________________________________________________________________________________
conv_dw_7 (DepthwiseConv2D)     (None, 14, 14, 512)  4608        conv_pw_6_relu[0][0]
________________________________________________________________________________________
conv_dw_7_bn (BatchNormalizatio (None, 14, 14, 512)  2048        conv_dw_7[0][0]
________________________________________________________________________________________
conv_dw_7_relu (ReLU)           (None, 14, 14, 512)  0           conv_dw_7_bn[0][0]
________________________________________________________________________________________
conv_pw_7 (Conv2D)              (None, 14, 14, 512)  262144      conv_dw_7_relu[0][0]
________________________________________________________________________________________
conv_pw_7_bn (BatchNormalizatio (None, 14, 14, 512)  2048        conv_pw_7[0][0]
________________________________________________________________________________________
conv_pw_7_relu (ReLU)           (None, 14, 14, 512)  0           conv_pw_7_bn[0][0]
________________________________________________________________________________________
conv_dw_8 (DepthwiseConv2D)     (None, 14, 14, 512)  4608        conv_pw_7_relu[0][0]
________________________________________________________________________________________
conv_dw_8_bn (BatchNormalizatio (None, 14, 14, 512)  2048        conv_dw_8[0][0]
________________________________________________________________________________________
conv_dw_8_relu (ReLU)           (None, 14, 14, 512)  0           conv_dw_8_bn[0][0]
________________________________________________________________________________________
```

```
conv_pw_8 (Conv2D)              (None, 14, 14, 512)  262144      conv_dw_8_relu[0][0]
__________________________________________________________________________________________________
conv_pw_8_bn (BatchNormalizatio (None, 14, 14, 512)  2048        conv_pw_8[0][0]
__________________________________________________________________________________________________
conv_pw_8_relu (ReLU)           (None, 14, 14, 512)  0           conv_pw_8_bn[0][0]
__________________________________________________________________________________________________
conv_dw_9 (DepthwiseConv2D)     (None, 14, 14, 512)  4608        conv_pw_8_relu[0][0]
__________________________________________________________________________________________________
conv_dw_9_bn (BatchNormalizatio (None, 14, 14, 512)  2048        conv_dw_9[0][0]
__________________________________________________________________________________________________
conv_dw_9_relu (ReLU)           (None, 14, 14, 512)  0           conv_dw_9_bn[0][0]
__________________________________________________________________________________________________
conv_pw_9 (Conv2D)              (None, 14, 14, 512)  262144      conv_dw_9_relu[0][0]
__________________________________________________________________________________________________
conv_pw_9_bn (BatchNormalizatio (None, 14, 14, 512)  2048        conv_pw_9[0][0]
__________________________________________________________________________________________________
conv_pw_9_relu (ReLU)           (None, 14, 14, 512)  0           conv_pw_9_bn[0][0]
__________________________________________________________________________________________________
conv_dw_10 (DepthwiseConv2D)    (None, 14, 14, 512)  4608        conv_pw_9_relu[0][0]
__________________________________________________________________________________________________
conv_dw_10_bn (BatchNormalizati (None, 14, 14, 512)  2048        conv_dw_10[0][0]
__________________________________________________________________________________________________
conv_dw_10_relu (ReLU)          (None, 14, 14, 512)  0           conv_dw_10_bn[0][0]
__________________________________________________________________________________________________
conv_pw_10 (Conv2D)             (None, 14, 14, 512)  262144      conv_dw_10_relu[0][0]
__________________________________________________________________________________________________
conv_pw_10_bn (BatchNormalizati (None, 14, 14, 512)  2048        conv_pw_10[0][0]
__________________________________________________________________________________________________
conv_pw_10_relu (ReLU)          (None, 14, 14, 512)  0           conv_pw_10_bn[0][0]
__________________________________________________________________________________________________
conv_dw_11 (DepthwiseConv2D)    (None, 14, 14, 512)  4608        conv_pw_10_relu[0][0]
__________________________________________________________________________________________________
conv_dw_11_bn (BatchNormalizati (None, 14, 14, 512)  2048        conv_dw_11[0][0]
__________________________________________________________________________________________________
conv_dw_11_relu (ReLU)          (None, 14, 14, 512)  0           conv_dw_11_bn[0][0]
__________________________________________________________________________________________________
conv_pw_11 (Conv2D)             (None, 14, 14, 512)  262144      conv_dw_11_relu[0][0]
__________________________________________________________________________________________________
conv_pw_11_bn (BatchNormalizati (None, 14, 14, 512)  2048        conv_pw_11[0][0]
__________________________________________________________________________________________________
conv_pw_11_relu (ReLU)          (None, 14, 14, 512)  0           conv_pw_11_bn[0][0]
__________________________________________________________________________________________________
conv_pad_12 (ZeroPadding2D)     (None, 15, 15, 512)  0           conv_pw_11_relu[0][0]
__________________________________________________________________________________________________
conv_dw_12 (DepthwiseConv2D)    (None, 7, 7, 512)    4608        conv_pad_12[0][0]
__________________________________________________________________________________________________
conv_dw_12_bn (BatchNormalizati (None, 7, 7, 512)    2048        conv_dw_12[0][0]
__________________________________________________________________________________________________
conv_dw_12_relu (ReLU)          (None, 7, 7, 512)    0           conv_dw_12_bn[0][0]
__________________________________________________________________________________________________
conv_pw_12 (Conv2D)             (None, 7, 7, 1024)   524288      conv_dw_12_relu[0][0]
__________________________________________________________________________________________________
conv_pw_12_bn (BatchNormalizati (None, 7, 7, 1024)   4096        conv_pw_12[0][0]
__________________________________________________________________________________________________
conv_pw_12_relu (ReLU)          (None, 7, 7, 1024)   0           conv_pw_12_bn[0][0]
__________________________________________________________________________________________________
conv_dw_13 (DepthwiseConv2D)    (None, 7, 7, 1024)   9216        conv_pw_12_relu[0][0]
```

```
conv_dw_13_bn (BatchNormalizati (None, 7, 7, 1024)   4096        conv_dw_13[0][0]
__________________________________________________________________________________________________
conv_dw_13_relu (ReLU)          (None, 7, 7, 1024)   0           conv_dw_13_bn[0][0]
__________________________________________________________________________________________________
conv_pw_13 (Conv2D)             (None, 7, 7, 1024)   1048576     conv_dw_13_relu[0][0]
__________________________________________________________________________________________________
conv_pw_13_bn (BatchNormalizati (None, 7, 7, 1024)   4096        conv_pw_13[0][0]
__________________________________________________________________________________________________
conv_pw_13_relu (ReLU)          (None, 7, 7, 1024)   0           conv_pw_13_bn[0][0]
__________________________________________________________________________________________________
up_sampling2d (UpSampling2D)    (None, 14, 14, 1024) 0           conv_pw_13_relu[0][0]
__________________________________________________________________________________________________
concatenate (Concatenate)       (None, 14, 14, 1536) 0           up_sampling2d[0][0]
                                                                 conv_pw_11_relu[0][0]
__________________________________________________________________________________________________
up_sampling2d_1 (UpSampling2D)  (None, 28, 28, 1536) 0           concatenate[0][0]
__________________________________________________________________________________________________
concatenate_1 (Concatenate)     (None, 28, 28, 1792) 0           up_sampling2d_1[0][0]
                                                                 conv_pw_5_relu[0][0]
__________________________________________________________________________________________________
up_sampling2d_2 (UpSampling2D)  (None, 56, 56, 1792) 0           concatenate_1[0][0]
__________________________________________________________________________________________________
concatenate_2 (Concatenate)     (None, 56, 56, 1920) 0           up_sampling2d_2[0][0]
                                                                 conv_pw_3_relu[0][0]
__________________________________________________________________________________________________
up_sampling2d_3 (UpSampling2D)  (None, 112, 112, 192 0           concatenate_2[0][0]
__________________________________________________________________________________________________
concatenate_3 (Concatenate)     (None, 112, 112, 198 0           up_sampling2d_3[0][0]
                                                                 conv_pw_1_relu[0][0]
__________________________________________________________________________________________________
up_sampling2d_4 (UpSampling2D)  (None, 224, 224, 198 0           concatenate_3[0][0]
__________________________________________________________________________________________________
conv2d (Conv2D)                 (None, 224, 224, 1)  1985        up_sampling2d_4[0][0]
__________________________________________________________________________________________________
reshape (Reshape)               (None, 224, 224)     0           conv2d[0][0]
==================================================================================================
Total params: 3,230,849
Trainable params: 3,208,961
Non-trainable params: 21,888
```

Define dice coefficient function (5 marks)

- Create a function to calculate dice coefficient

Dice Coefficient (F1 Score) Explanation

The Dice Coefficient is 2 * the Area of Overlap divided by the total number of pixels in both images

```
def dice_coefficient(y_true, y_pred):
    #### Add your code here ####
    numerator = 2 * tf.reduce_sum(y_true * y_pred)
    denominator = tf.reduce_sum(y_true + y_pred)
```

```
    return numerator / (denominator + tf.keras.backend.epsilon())
```

Define loss

```
from tensorflow.keras.losses import binary_crossentropy
from tensorflow.keras.backend import log, epsilon
def loss(y_true, y_pred):
    return binary_crossentropy(y_true, y_pred) - log(dice_coefficient(y_true, y_pred) + epsil
```

Compile the model (2 marks)

- Complie the model using below parameters
    - loss: use the loss function defined above
    - optimizers: use Adam optimizer
    - metrics: use dice_coefficient function defined above

```
#### Add your code here ####
from tensorflow.keras.optimizers import Adam
optimizer = Adam(lr=1e-4, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0, amsgrad=False)
model.compile(loss=loss, optimizer=optimizer, metrics=[dice_coefficient])
```

Define checkpoint and earlystopping

```
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau
checkpoint = ModelCheckpoint("model-{loss:.2f}.h5", monitor="loss", verbose=1, save_best_only
                             save_weights_only=True, mode="min", period=1)
stop = EarlyStopping(monitor="loss", patience=5, mode="min")
reduce_lr = ReduceLROnPlateau(monitor="loss", factor=0.2, patience=5, min_lr=1e-6, verbose=1,
```

```
WARNING:tensorflow:`period` argument is deprecated. Please use `save_freq` to specify th
```

Fit the model (2 marks)

- Fit the model using below parameters
    - epochs: you can decide
    - batch_size: 1
    - callbacks: checkpoint, reduce_lr, stop

```
#### Add your code here ####
EPOCHS = 10
BATCH_SIZE =1
model.fit(X_train, masks, batch_size=BATCH_SIZE, nb_epoch=EPOCHS, callbacks=[checkpoint, redu
```

```
model.fit(X_train, masks, batch_size=BATCH_SIZE, nb_epoch=EPOCHS, callbacks=[checkpoint, redu
                    use_multiprocessing=False)
```

```
WARNING:tensorflow:The `nb_epoch` argument in `fit` has been renamed `epochs`.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_core/python/op
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
Train on 409 samples
Epoch 1/10
407/409 [============================>.] - ETA: 0s - loss: 1.3264 - dice_coefficient: 0.
Epoch 00001: loss improved from inf to 1.32828, saving model to model-1.33.h5
409/409 [==============================] - 28s 68ms/sample - loss: 1.3283 - dice_coeffic
Epoch 2/10
407/409 [============================>.] - ETA: 0s - loss: 0.7881 - dice_coefficient: 0.
Epoch 00002: loss improved from 1.32828 to 0.78870, saving model to model-0.79.h5
409/409 [==============================] - 16s 40ms/sample - loss: 0.7887 - dice_coeffic
Epoch 3/10
407/409 [============================>.] - ETA: 0s - loss: 0.6235 - dice_coefficient: 0.
Epoch 00003: loss improved from 0.78870 to 0.62350, saving model to model-0.62.h5
409/409 [==============================] - 16s 40ms/sample - loss: 0.6235 - dice_coeffic
Epoch 4/10
408/409 [============================>.] - ETA: 0s - loss: 0.5598 - dice_coefficient: 0.
Epoch 00004: loss improved from 0.62350 to 0.55947, saving model to model-0.56.h5
409/409 [==============================] - 16s 40ms/sample - loss: 0.5595 - dice_coeffic
Epoch 5/10
407/409 [============================>.] - ETA: 0s - loss: 0.5170 - dice_coefficient: 0.
Epoch 00005: loss improved from 0.55947 to 0.51682, saving model to model-0.52.h5
409/409 [==============================] - 16s 39ms/sample - loss: 0.5168 - dice_coeffic
Epoch 6/10
407/409 [============================>.] - ETA: 0s - loss: 0.4829 - dice_coefficient: 0.
Epoch 00006: loss improved from 0.51682 to 0.48293, saving model to model-0.48.h5
409/409 [==============================] - 17s 42ms/sample - loss: 0.4829 - dice_coeffic
Epoch 7/10
407/409 [============================>.] - ETA: 0s - loss: 0.4680 - dice_coefficient: 0.
Epoch 00007: loss improved from 0.48293 to 0.46742, saving model to model-0.47.h5
409/409 [==============================] - 16s 40ms/sample - loss: 0.4674 - dice_coeffic
Epoch 8/10
407/409 [============================>.] - ETA: 0s - loss: 0.4481 - dice_coefficient: 0.
Epoch 00008: loss improved from 0.46742 to 0.44828, saving model to model-0.45.h5
409/409 [==============================] - 17s 41ms/sample - loss: 0.4483 - dice_coeffic
Epoch 9/10
407/409 [============================>.] - ETA: 0s - loss: 0.4326 - dice_coefficient: 0.
Epoch 00009: loss improved from 0.44828 to 0.43255, saving model to model-0.43.h5
409/409 [==============================] - 16s 39ms/sample - loss: 0.4325 - dice_coeffic
Epoch 10/10
407/409 [============================>.] - ETA: 0s - loss: 0.4217 - dice_coefficient: 0.
Epoch 00010: loss improved from 0.43255 to 0.42149, saving model to model-0.42.h5
409/409 [==============================] - 16s 40ms/sample - loss: 0.4215 - dice_coeffic
<tensorflow.python.keras.callbacks.History at 0x7f69e00464a8>
```

Get the predicted mask for a sample image (3 marks)

```
n = 10
sample_image = X_train[n]
```

```
sample_image = X_train[n]
#### Add your code here ####
print(sample_image.shape)
sample_image_reshaped = np.reshape(sample_image,(1,sample_image.shape[0],sample_image.shape[1
print(sample_image_reshaped.shape)
predicted_mask = model.predict(sample_image_reshaped)
```

```
(224, 224, 3)
(1, 224, 224, 3)
```

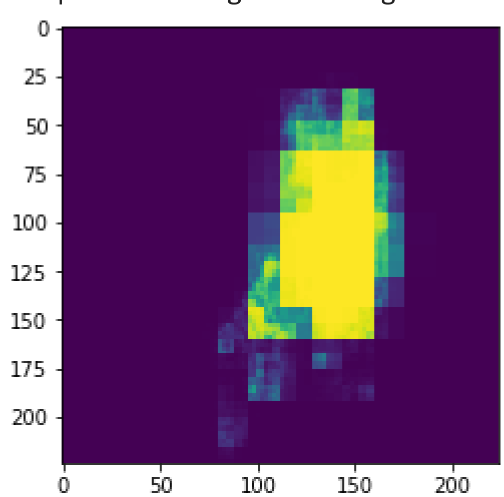
```
print(predicted_mask.shape)
print(masks[n].shape)
predicted_mask_reshaped = np.reshape(predicted_mask,masks[n].shape)
print(predicted_mask_reshaped.shape)
```

```
(1, 224, 224)
(224, 224)
(224, 224)
```

Impose the mask on the image (3 marks)

```
#### Add your code here ####
#pyplot.imshow(masks[n])
pyplot.imshow(predicted_mask_reshaped)
```

```
<matplotlib.image.AxesImage at 0x7f677d94e2b0>
```




```
pyplot.imshow(masks[n])
```

```
<matplotlib.image.AxesImage at 0x7f677d926f28>
```