Import libraries

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

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call d
```

```
!pip3 install pydicom
!pip3 install segmentation_models
```

```
Requirement already satisfied: pydicom in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: segmentation_models in /usr/local/lib/python3.7/
Requirement already satisfied: efficientnet==1.0.0 in /usr/local/lib/python3.7/
Requirement already satisfied: image-classifiers==1.0.0 in /usr/local/lib/pytho
Requirement already satisfied: keras-applications<=1.0.8,>=1.0.7 in /usr/local/
Requirement already satisfied: scikit-image in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: h5py in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: numpy>=1.9.1 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: matplotlib!=3.0.0,>=2.0.0 in /usr/local/lib/pyth
Requirement already satisfied: pillow>=4.3.0 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: scipy>=0.19.0 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: networkx>=2.0 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: PyWavelets>=0.4.0 in /usr/local/lib/python3.7/di
Requirement already satisfied: imageio>=2.3.0 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/di
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr
Requirement already satisfied: decorator>=4.3.0 in /usr/local/lib/python3.7/dis
```

```

import warnings
warnings.filterwarnings('ignore')

import os
import cv2
import csv
import pickle
import pydicom
import numpy as np
import pandas as pd
from glob import glob

os.chdir('/content/drive/MyDrive/Initial-Code')
#os.chdir('./Initial-Code')

```

```
# import the necessary packages
import keras
import tensorflow as tf
from keras import backend as K

from dataset import prepare_data
from metric_loss import my_iou_metric, iou_metric_batch_val, bce_dice_loss
%env SM_FRAMEWORK=tf.keras

from predict import predict_result_val, prepare_test, get_test, get_prediction,
from generator import DataGenerator, label_generator

import seg_models
keras.backend.set_image_data_format('channels_last')

from keras.optimizers import SGD
from keras.callbacks import ModelCheckpoint

import sys
sys.path.insert(0, 'siim-acr-pneumothorax-segmentation')
from mask_functions import rle2mask, mask2rle
#X
## Seeding
seed = 1994
np.random.seed = seed
os.environ['PYTHONHASHSEED'] = str(seed)
tf.seed = seed

import gc   #Gabage collector for cleaning deleted data from memory
```

```
env: SM_FRAMEWORK=tf.keras
```

```
#!pip3 install pydicom
#!pip3 install keras
#!pip3 install tensorflow
#!pip3 install sklearn
#!pip3 install segmentation_models
#!pip3 install generic_utils

#!pip3 install  albumentations
!pip3 install backbone-network
```

Dataset

```
# defining configuration parameters
org_size = 1024 # original image size
img_size = 256 # image resize size
batch_size = 10 # batch size for training unet
```

Load train and validation data from files

```
pkl_file_train = open('process_data/X_train.pkl', 'rb')

X_train = pickle.load(pkl_file_train)
```

```
pkl_file_val = open('process_data/X_val.pkl', 'rb')

X_val = pickle.load(pkl_file_val)
```

```
pkl_file_masks = open('process_data/masks.pkl', 'rb')

masks = pickle.load(pkl_file_masks)
```

Data generation & Augmentations

```
import albumentations as A
```

```
training_augmentation = A.Compose([
    A.HorizontalFlip(p=0.5),
    A.OneOf([
        #A.CLAHE(),
        A.RandomContrast(),
        A.RandomGamma(),
        A.RandomBrightness(),
         ], p=0.3),
    A.OneOf([
        A.ElasticTransform(alpha=120, sigma=120 * 0.05, alpha_affine=120 * 0.03)
        A.GridDistortion(),
        A.OpticalDistortion(distort_limit=2, shift_limit=0.5),
        ], p=0.3),
    A.ShiftScaleRotate(shift_limit=0.2, scale_limit=0.2, rotate_limit=20,
                                        interpolation=cv2.INTER_LINEAR, border_m
    A.RandomSizedCrop(min_max_height=(206,256), height=img_size, width=img_size,
],p=1)
```

```
params_train = {'img_size': img_size,
          'batch_size': batch_size,
          'n_channels': 3,
          'shuffle': True,
           'augmentations':training_augmentation,
           }

```

```
params_val = {'img_size': img_size,
          'batch_size': batch_size,
          'n_channels': 3,
          'shuffle': True,
         }

# Generators
training_generator = DataGenerator(X_train, masks, **params_train)
validation_generator = DataGenerator(X_val, masks, **params_val)
```

```
x, y = training_generator.__getitem__(0)
print(x.shape, y.shape)
```

```
(10, 256, 256, 3) (10, 256, 256, 1)
```

Segmentation model

```
K.clear_session()
```

```
BACKBONE = 'resnet50'
model = seg_models.Unet(backbone_name=BACKBONE, encoder_weights='imagenet') #, c
model.summary()
```

```
                                                                 add_11[0][0]
______________________________________________________________________________
stage4_unit1_bn1 (BatchNormaliz (None, None, None, 1 4096        add_12[0][0]
______________________________________________________________________________
stage4_unit1_relu1 (Activation) (None, None, None, 1 0           stage4_unit1_
______________________________________________________________________________
stage4_unit1_conv1 (Conv2D)     (None, None, None, 5 524288      stage4_unit1_
______________________________________________________________________________
stage4_unit1_bn2 (BatchNormaliz (None, None, None, 5 2048        stage4_unit1_
______________________________________________________________________________
stage4_unit1_relu2 (Activation) (None, None, None, 5 0           stage4_unit1_
______________________________________________________________________________
zero_padding2d_15 (ZeroPadding2 (None, None, None, 5 0           stage4_unit1_
______________________________________________________________________________
stage4_unit1_conv2 (Conv2D)     (None, None, None, 5 2359296     zero_padding2
______________________________________________________________________________
stage4_unit1_bn3 (BatchNormaliz (None, None, None, 5 2048        stage4_unit1_
______________________________________________________________________________
stage4_unit1_relu3 (Activation) (None, None, None, 5 0           stage4_unit1_
______________________________________________________________________________
stage4_unit1_conv3 (Conv2D)     (None, None, None, 2 1048576     stage4_unit1_
______________________________________________________________________________
stage4_unit1_sc (Conv2D)        (None, None, None, 2 2097152     stage4_unit1_
______________________________________________________________________________
add_13 (Add)                    (None, None, None, 2 0           stage4_unit1_
                                                                 stage4_unit1_
______________________________________________________________________________
stage4_unit2_bn1 (BatchNormaliz (None, None, None, 2 8192        add_13[0][0]
stage4_unit2_bn1 (BatchNormaliz (None, None, None, 2 8192        add_13[0][0]
__________________________________________________________________________________
stage4_unit2_relu1 (Activation) (None, None, None, 2 0           stage4_unit2_
__________________________________________________________________________________
stage4_unit2_conv1 (Conv2D)     (None, None, None, 5 1048576     stage4_unit2_
__________________________________________________________________________________
stage4_unit2_bn2 (BatchNormaliz (None, None, None, 5 2048        stage4_unit2_
__________________________________________________________________________________
stage4_unit2_relu2 (Activation) (None, None, None, 5 0           stage4_unit2_
__________________________________________________________________________________
zero_padding2d_16 (ZeroPadding2 (None, None, None, 5 0           stage4_unit2_
__________________________________________________________________________________
stage4_unit2_conv2 (Conv2D)     (None, None, None, 5 2359296     zero_padding2
__________________________________________________________________________________
stage4_unit2_bn3 (BatchNormaliz (None, None, None, 5 2048        stage4_unit2_
__________________________________________________________________________________
stage4_unit2_relu3 (Activation) (None, None, None, 5 0           stage4_unit2_
__________________________________________________________________________________

stage4_unit2_conv3 (Conv2D)     (None, None, None, 2 1048576     stage4_unit2_
__________________________________________________________________________________
add_14 (Add)                    (None, None, None, 2 0           stage4_unit2_
                                                                 add_13[0][0]
__________________________________________________________________________________
stage4_unit3_bn1 (BatchNormaliz (None, None, None, 2 8192        add_14[0][0]
__________________________________________________________________________________
stage4_unit3_relu1 (Activation) (None, None, None, 2 0           stage4_unit3_
__________________________________________________________________________________
stage4_unit3_conv1 (Conv2D)     (None, None, None, 5 1048576     stage4_unit3_
__________________________________________________________________________________
stage4_unit3_bn2 (BatchNormaliz (None, None, None, 5 2048        stage4_unit3_
__________________________________________________________________________________
stage4_unit3_relu2 (Activation) (None, None, None, 5 0           stage4_unit3_
```

```python
# From: https://github.com/jocicmarko/ultrasound-nerve-segmentation/blob/master/
def dice_coef(y_true, y_pred):
    y_true_f = tf.keras.layers.flatten(y_true)
    y_pred_f = tf.keras.layers.flatten(y_pred)
    intersection = keras.sum(y_true_f * y_pred_f)
    return (2. * intersection + 1) / (keras.sum(y_true_f) + keras.sum(y_pred_f)

def dice_coef_loss(y_true, y_pred):
    return -dice_coef(y_true, y_pred)

def unet(input_size=(256,256,1)):

    inputs = Input(input_size)

    conv1 = Conv2D(32, (3, 3), activation='relu', padding='same')(inputs)
    conv1 = Conv2D(32, (3, 3), activation='relu', padding='same')(conv1)
    pool1 = MaxPooling2D(pool_size=(2, 2))(conv1)

    conv2 = Conv2D(64, (3, 3), activation='relu', padding='same')(pool1)
    conv2 = Conv2D(64, (3, 3), activation='relu', padding='same')(conv2)
    pool2 = MaxPooling2D(pool_size=(2, 2))(conv2)

    conv3 = Conv2D(128, (3, 3), activation='relu', padding='same')(pool2)
    conv3 = Conv2D(128, (3, 3), activation='relu', padding='same')(conv3)
    pool3 = MaxPooling2D(pool_size=(2, 2))(conv3)

    conv4 = Conv2D(256, (3, 3), activation='relu', padding='same')(pool3)
    conv4 = Conv2D(256, (3, 3), activation='relu', padding='same')(conv4)
    pool4 = MaxPooling2D(pool_size=(2, 2))(conv4)

    conv5 = Conv2D(512, (3, 3), activation='relu', padding='same')(pool4)
    conv5 = Conv2D(512, (3, 3), activation='relu', padding='same')(conv5)

    up6 = concatenate([Conv2DTranspose(256, (2, 2), strides=(2, 2), padding='sam
    conv6 = Conv2D(256, (3, 3), activation='relu', padding='same')(up6)
    conv6 = Conv2D(256, (3, 3), activation='relu', padding='same')(conv6)

    up7 = concatenate([Conv2DTranspose(128, (2, 2), strides=(2, 2), padding='sam
    conv7 = Conv2D(128, (3, 3), activation='relu', padding='same')(up7)
    conv7 = Conv2D(128, (3, 3), activation='relu', padding='same')(conv7)

    up8 = concatenate([Conv2DTranspose(64, (2, 2), strides=(2, 2), padding='same
    conv8 = Conv2D(64, (3, 3), activation='relu', padding='same')(up8)
    conv8 = Conv2D(64, (3, 3), activation='relu', padding='same')(conv8)

    up9 = concatenate([Conv2DTranspose(32, (2, 2), strides=(2, 2), padding='same
    conv9 = Conv2D(32, (3, 3), activation='relu', padding='same')(up9)
    conv9 = Conv2D(32, (3, 3), activation='relu', padding='same')(conv9)

    conv10 = Conv2D(1, (1, 1), activation='sigmoid')(conv9)

    return Model(inputs=[inputs], outputs=[conv10])


```

```
opt = SGD(momentum=0.9)
```

```
model.compile(optimizer=opt, loss=bce_dice_loss, metrics=[my_iou_metric])
```

```
from swa import SWA
from cosine_schedule import CosineAnnealingScheduler
```

```
epochs = 60
swa = SWA('model_output/512_resnet50_swa.model',55)

callbacks = [
    ModelCheckpoint("model_output/512_resnet50.model",monitor='val_loss',
                            mode = 'min', save_best_only=True,
                            verbose=1),
    swa,
    CosineAnnealingScheduler(T_max=epochs, eta_max=1e-3, eta_min=1e-5, verbose=1
]
```

```
history = model.fit_generator(generator=training_generator,
                              validation_data=validation_generator,
                             epochs=epochs, verbose=1,
                              callbacks=callbacks)
```

```
Epoch 39/60

Epoch 00039: CosineAnnealingScheduler setting learning rate to 0.0003036653616
511/511 [==============================] - 203s 396ms/step - loss: 0.6643 - my

Epoch 00039: val_loss did not improve from 0.73848
Epoch 40/60

Epoch 00040: CosineAnnealingScheduler setting learning rate to 0.0002802747026
511/511 [==============================] - 203s 397ms/step - loss: 0.6695 - my

Epoch 00040: val_loss improved from 0.73848 to 0.73527, saving model to model_
INFO:tensorflow:Assets written to: model_output/512_resnet50.model/assets
Epoch 41/60

Epoch 00041: CosineAnnealingScheduler setting learning rate to 0.0002575000000
511/511 [==============================] - 205s 400ms/step - loss: 0.6612 - my

Epoch 00041: val_loss improved from 0.73527 to 0.72180, saving model to model_
INFO:tensorflow:Assets written to: model_output/512_resnet50.model/assets
Epoch 42/60

Epoch 00042: CosineAnnealingScheduler setting learning rate to 0.0002354036776
511/511 [==============================] - 206s 403ms/step - loss: 0.6366 - my

Epoch 00042: val_loss improved from 0.72180 to 0.71721, saving model to model_
INFO:tensorflow:Assets written to: model_output/512_resnet50.model/assets
Epoch 43/60

Epoch 00043: CosineAnnealingScheduler setting learning rate to 0.0002140463001
511/511 [==============================] - 206s 402ms/step - loss: 0.6370 - my

Epoch 00043: val_loss did not improve from 0.71721

Epoch 44/60

Epoch 00044: CosineAnnealingScheduler setting learning rate to 0.0001934864064
511/511 [==============================] - 205s 402ms/step - loss: 0.6312 - my

Epoch 00044: val_loss did not improve from 0.71721
Epoch 45/60

Epoch 00045: CosineAnnealingScheduler setting learning rate to 0.0001737803498
511/511 [==============================] - 203s 396ms/step - loss: 0.6303 - my

Epoch 00045: val_loss did not improve from 0.71721
Epoch 46/60

Epoch 00046: CosineAnnealingScheduler setting learning rate to 0.0001549821433
511/511 [==============================] - 203s 398ms/step - loss: 0.6425 - my
```

```
511/511 [==============================] - 203s 398ms/step - loss: 0.6435 - my

Epoch 00046: val_loss did not improve from 0.71721
Epoch 47/60

Epoch 00047: CosineAnnealingScheduler setting learning rate to 0.000137143311
511/511 [==============================] - 205s 400ms/step - loss: 0.6304 - my

Epoch 00047: val_loss did not improve from 0.71721
Epoch 48/60
```

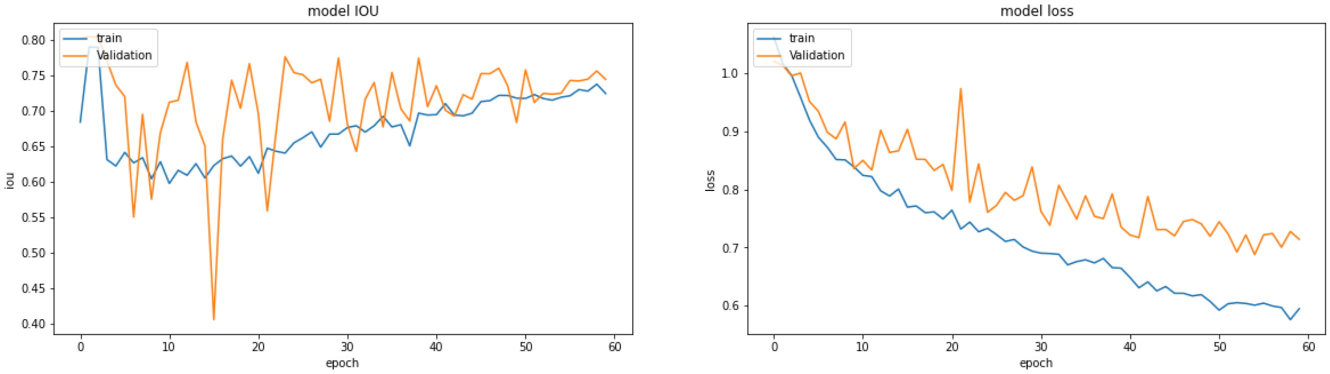
```
# list all data in history
import matplotlib.pyplot as plt

print(history.history.keys())

# summarize history for iou
plt.figure(figsize=(20,5))
plt.subplot(1,2,1)
plt.plot(history.history['my_iou_metric'])
plt.plot(history.history['val_my_iou_metric'])
plt.title('model IOU')
plt.ylabel('iou')
plt.xlabel('epoch')
plt.legend(['train', 'Validation'], loc='upper left')

# summarize history for loss
plt.subplot(1,2,2)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'Validation'], loc='upper left')
```

```
dict_keys(['loss', 'my_iou_metric', 'val_loss', 'val_my_iou_metric', 'lr'])
<matplotlib.legend.Legend at 0x7fecb0a5e990>
```



```
# Load best model or swa model

model.load_weights('model_output/512_resnet50_swa.model')

#print('using best weight model')
#model.load_weights('stage2_model_output/256_resnet34.model')
```

```
<tensorflow.python.training.tracking.util.CheckpointLoadStatus at 0x7feca9bde2d
```

```
# defining configuration parameters
org_size = 1024 # original image size
img_size = 512# image resize size
batch_size = 10 # batch size for training unet
```

```
pkl_file_val = open('process_data/X_val.pkl', 'rb')

X_val = pickle.load(pkl_file_val)
```

```
pkl_file_val = open('process_data/X_val.pkl', 'rb')

X_val = pickle.load(pkl_file_val)
```

```
pkl_file_masks = open('process_data/masks.pkl', 'rb')

masks = pickle.load(pkl_file_masks)
```

```
import albumentations as A
```

```
training_augmentation = A.Compose([
    A.HorizontalFlip(p=0.5),
    A.OneOf([
        #A.CLAHE(),
        A.RandomContrast(),
        A.RandomGamma(),
        A.RandomBrightness(),
         ], p=0.3),
    A.OneOf([
        A.ElasticTransform(alpha=120, sigma=120 * 0.05, alpha_affine=120 * 0.03)
        A.GridDistortion(),
        A.OpticalDistortion(distort_limit=2, shift_limit=0.5),
        ], p=0.3),
    A.ShiftScaleRotate(shift_limit=0.2, scale_limit=0.2, rotate_limit=20,
                                        interpolation=cv2.INTER_LINEAR, border_m
    A.RandomSizedCrop(min_max_height=(412, 512), height=img_size, width=img_size
],p=1)
```

```
params_train = {'img_size': img_size,
params_train = {'img_size': img_size,
          'batch_size': batch_size,
          'n_channels': 3,
          'shuffle': True,
           'augmentations':training_augmentation,
           }

params_val = {'img_size': img_size,
          'batch_size': batch_size,
          'n_channels': 3,
          'shuffle': True,
         }

# Generators
training_generator = DataGenerator(X_train, masks, **params_train)
validation_generator = DataGenerator(X_val, masks, **params_val)
```

```
x, y = training_generator.__getitem__(0)
print(x.shape, y.shape)
```

```
(10, 512, 512, 3) (10, 512, 512, 1)
```

```
from swa import SWA
from cosine_schedule import CosineAnnealingScheduler
```

```
epochs = 60
swa = SWA('model_output/512_resnet50_swa_stage2.model',55)

callbacks = [
    ModelCheckpoint("model_output/512_resnet50_stage2.model",monitor='val_loss',
                            mode = 'min', save_best_only=True,
                            verbose=1),
    swa,
    CosineAnnealingScheduler(T_max=epochs, eta_max=1e-3, eta_min=1e-5, verbose=1
]
```

```
history = model.fit_generator(generator=training_generator,
                            validation_data=validation_generator,
                           epochs=epochs, verbose=1,
                            callbacks=callbacks)
```

```
Stochastic weight averaging selected for last 5 epochs.
Epoch 1/60

Epoch 00001: CosineAnnealingScheduler setting learning rate to 0.001.
511/511 [==============================] - 322s 630ms/step - loss: 0.7588 - my_

Epoch 00001: val_loss improved from inf to 0.86037, saving model to model_outpu
INFO:tensorflow:Assets written to: model_output/512_resnet50_stage2.model/assets
Epoch 2/60
Epoch 00002: CosineAnnealingScheduler setting learning rate to 0.00099932161970
511/511 [==============================] - 323s 631ms/step - loss: 0.7320 - my_

Epoch 00002: val_loss improved from 0.86037 to 0.76955, saving model to model_ou
INFO:tensorflow:Assets written to: model_output/512_resnet50_stage2.model/asset
Epoch 3/60

Epoch 00003: CosineAnnealingScheduler setting learning rate to 0.00099728833820
511/511 [==============================] - 324s 632ms/step - loss: 0.7111 - my_

Epoch 00003: val_loss did not improve from 0.76955
Epoch 4/60

Epoch 00004: CosineAnnealingScheduler setting learning rate to 0.00099390572859
511/511 [==============================] - 321s 627ms/step - loss: 0.7146 - my_

Epoch 00004: val_loss did not improve from 0.76955
Epoch 5/60

Epoch 00005: CosineAnnealingScheduler setting learning rate to 0.00098918306236
511/511 [==============================] - 316s 618ms/step - loss: 0.6904 - my_

Epoch 00005: val_loss improved from 0.76955 to 0.75707, saving model to model_ou
INFO:tensorflow:Assets written to: model_output/512_resnet50_stage2.model/asset
Epoch 6/60

Epoch 00006: CosineAnnealingScheduler setting learning rate to 0.00098313328401
511/511 [==============================] - 323s 630ms/step - loss: 0.6844 - my_

Epoch 00006: val_loss did not improve from 0.75707
Epoch 7/60

Epoch 00007: CosineAnnealingScheduler setting learning rate to 0.00097577297555
359/511 [===================>..........] - ETA: 1:27 - loss: 0.6858 - my_iou_me
```

```python
# list all data in history
import matplotlib.pyplot as plt

print(history.history.keys())

# summarize history for iou
plt.figure(figsize=(20,5))
plt.subplot(1,2,1)
plt.plot(history.history['my_iou_metric'])
plt.plot(history.history['val_my_iou_metric'])
plt.title('model IOU')
plt.ylabel('iou')
plt.xlabel('epoch')
plt.legend(['train', 'Validation'], loc='upper left')

# summarize history for loss
plt.subplot(1,2,2)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'Validation'], loc='upper left')
```

Evaluation validation data

```
params_val = {'img_size': img_size,
          'batch_size': 5,
          'n_channels': 3,
          'shuffle': False,
         }

# Generators
validation_generator = DataGenerator(X_val, masks, **params_val)
```

```
AUGMENTATIONS_TEST_FLIPPED = A.Compose([
    A.HorizontalFlip(p=1),
],p=1)

params_val_flip = {'img_size': img_size,
          'batch_size': 5,
          'n_channels': 3,
          'shuffle': False,
        'augmentations':AUGMENTATIONS_TEST_FLIPPED,
         }

validation_generator_flipped = DataGenerator(X_val, masks, **params_val_flip)
```

```
preds_valid_orig = predict_result(model,validation_generator,img_size)
preds_valid_flipped = predict_result(model,validation_generator_flipped,img_size
preds_valid_flipped = np.array([np.fliplr(x) for x in preds_valid_flipped])
preds_valid = 0.5*preds_valid_orig + 0.5*preds_valid_flipped
```

```
np.savez_compressed('process_data/val_pre/preds_valid_resnet50', array1= preds_v
```

```
y_truth_val = label_generator(X_val, masks, len(preds_valid), img_size, 3)

np.savez_compressed('process_data/val_pre/y_truth_val', array1= y_truth_val)
```

```
decompressed_array= np.load("process_data/val_pre/y_truth_val.npz")
y_truth_val = decompressed_array['array1']
```

```
## Scoring for last model
score = 0.0
mask_area = 0
best_th = 0

thresholds = np.arange(0.2, 0.9, 0.01)
areas = [1024, 2048, 3072, 4096]
for threshold in tqdm(thresholds):
    for area in tqdm(areas):
        iou = iou_metric_batch_val(y_truth_val, np.int32(preds_valid > threshold
        if iou > score:
            score = iou
            mask_area = area
            best_th = threshold
            print("Threshold {}\tMask area {}\tIoU {}".format(best_th, mask_area
    print()
```

Test Prediction

```
test_file = 'stage2_siim_data/stage_2_images/*.dcm'
test_metadata_df = prepare_test(test_file, rle_file)
```

```
test_data = get_test(3205, test_metadata_df, img_size=img_size, channels=3) #0,
print(test_data.shape)
```

```
resnet50_512_pred_test = get_prediction(model, test_data, batch_size=batch_size)
```

```
np.savez_compressed('process_data/test_pre/resnet50_512_pred_test', array1= resn
```

```
decompressed_array= np.load("process_data/test_pre/resnet50_512_pred_test.npz")
resnet50_512_pred_test = decompressed_array['array1']
```

```
rles = get_rles(preds_test, b_th = 0.73, r_th = 2048)
```

```
test_fn = sorted(glob('stage2_siim_data/stage_2_images/*.dcm'))
test_IDs = [o.split('/')[-1][:-4] for o in test_fn]
```

```
sub_df = pd.DataFrame({'ImageId': test_IDs, 'EncodedPixels': rles})
sub_df.loc[sub_df.EncodedPixels=='', 'EncodedPixels'] = '-1'
sub_df.head()
```

```
sub_df.to_csv('model_submission/resnet50_submission.csv', index=False)
```

```
sub_df['EncodedPixels'].value_counts(normalize=True) * 100
```