Generic-Semantic-Melanoma-Segment-Unet++ (G)

March 23, 2023

1 Segment using Unet++

[86]:

```
#Libraries-----------------
import os
import random
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import warnings
import gc
from pathlib import Path
from tqdm.notebook import trange, tqdm
from itertools import chain
from skimage.io import imread, imshow, concatenate_images
from skimage.transform import resize
from skimage.morphology import label
from sklearn.model_selection import train_test_split
import glob
import cv2
from PIL import Image
import glob2
from tensorflow.keras.models import load_model
import tensorflow
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator,␣
↪array_to_img, img_to_array, load_img
from tensorflow.keras.layers import Conv2D, Input, MaxPooling2D, Dropout,␣
↪concatenate, UpSampling2D
from tensorflow.keras.models import load_model, Model
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint,␣
↪ReduceLROnPlateau, TensorBoard
from tensorflow.keras import backend as K
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import (BatchNormalization, Conv2DTranspose,
                                     SeparableConv2D, MaxPooling2D, Activation,␣
↪Flatten, Dropout, Dense)
```

```python
from tensorflow.keras.preprocessing.image import load_img, array_to_img,␣
 ↪img_to_array
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Conv2D, LeakyReLU, BatchNormalization,␣
 ↪MaxPool2D,Conv2DTranspose, concatenate,Input
from tensorflow.keras.callbacks import CSVLogger
from tensorflow.keras.utils import plot_model
import pickle
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import sklearn
from sklearn.cluster import KMeans
from tensorflow.keras.layers import *
from tensorflow.keras import models
from tensorflow.keras.callbacks import *
from tensorflow.keras.applications import ResNet50
from sklearn.utils import shuffle
import matplotlib.pyplot as plt
from tensorflow.keras.metrics import MeanIoU

K.clear_session()
warnings.filterwarnings('ignore')
plt.style.use("ggplot")
get_ipython().run_line_magic('matplotlib', 'inline')
```

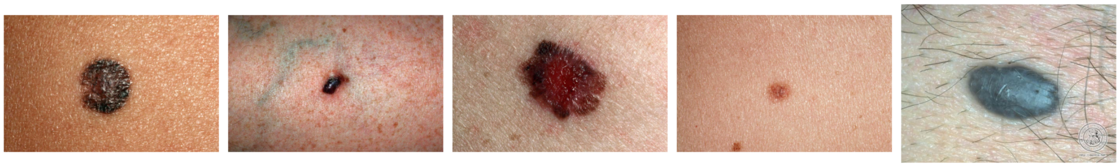
2 Reding and Preprocessing images

[87]:
```python
#Load image data-------------------
H,W,CH=[128,128,3]
def cv_load_img(path):
    img= cv2.imread(path)
    img= cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    img=cv2.resize(img,(W,H))
    return img
```

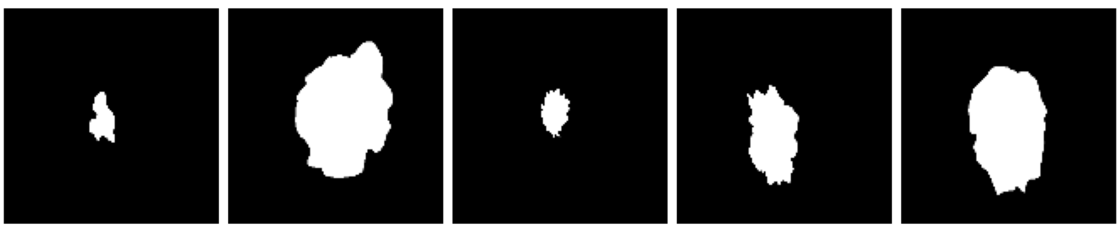
[88]:
```python
#Load data---------------------
BASE_DIR="Melanoma/train/"
img_path= os.listdir(BASE_DIR+'images')
mask_path= os.listdir(BASE_DIR+'masks')
```

[89]:
```python
#plot sample images--------------
fig, ax= plt.subplots(1,5, figsize=(20, 10))
for i in range(5):
    path= BASE_DIR + 'images/'
    ax[i].imshow(load_img(path + img_path[i]))
    ax[i].set_xticks([]); ax[i].set_yticks([])
```

```
fig.tight_layout()
plt.show()
```



[90]:
```
#plot sample masks---------------
fig, ax= plt.subplots(1,5, figsize=(20, 10))
for i in range(5):
    path= BASE_DIR + 'masks/'
    ax[i].imshow(cv_load_img(path + mask_path[i])[:, :, 0], 'gray')
    ax[i].set_xticks([]); ax[i].set_yticks([])

fig.tight_layout()
plt.show()
```



[91]:
```
#plot sample images--with blended mask ------------
fig, ax= plt.subplots(1,5, figsize=(20, 10))
for i in range(5):
    path1= BASE_DIR + 'images/'
    ax[i].imshow((cv_load_img(path1 + img_path[i])/255) * (cv_load_img(path +␣
 ↪mask_path[i])/255))
    ax[i].set_xticks([]); ax[i].set_yticks([])

fig.tight_layout()
plt.show()
```

3 Augmented Images and Masks

[92]:
```
#data  preparation
X_train, X_test, y_train, y_test = train_test_split(img_path, mask_path,␣
↪test_size=0.2, random_state=22)
len(X_train), len(X_test)
```

[92]: (164, 42)

[93]:
```
#batch generation-----------------------
def load_data(path_list, gray=False):
    data=[]
    for path in tqdm(path_list):
        img= cv_load_img(path)
        if gray:
            img= img[:, :, 0:1]
        img= cv2.resize(img, (W, H))
        data.append(img)
    return np.array(data)
```

[94]:
```
#train data generation---------------------
X_train= load_data([BASE_DIR + 'images/' + x for x in X_train])/255.0
X_test= load_data([BASE_DIR + 'images/' + x for x in X_test])/255.0
```

```
0%|          | 0/164 [00:00<?, ?it/s]

0%|          | 0/42 [00:00<?, ?it/s]
```

[95]:
```
##test data generation---------------------
Y_train= load_data([BASE_DIR + 'masks/' + x for x in y_train], gray=True)/255.0
Y_test= load_data([BASE_DIR + 'masks/' + x for x in y_test], gray=True)/255.0
Y_train= Y_train.reshape(-1, W, H, 1)
Y_test= Y_test.reshape(-1, W, H, 1)

Y_train.shape, Y_test.shape
```

```
0%|          | 0/164 [00:00<?, ?it/s]

0%|          | 0/42 [00:00<?, ?it/s]
```

[95]: ((164, 128, 128, 1), (42, 128, 128, 1))

Unet Model

[96]:

```
def Conv2dBlock(inputTensor, numFilters, kernelSize = 3, doBatchNorm = True):
    #first Conv
    x = tf.keras.layers.Conv2D(filters = numFilters, kernel_size = (kernelSize,␣
↪kernelSize),
                              kernel_initializer = 'he_normal', padding =␣
↪'same') (inputTensor)

    if doBatchNorm:
        x = tf.keras.layers.BatchNormalization()(x)

    x =tf.keras.layers.Activation('relu')(x)

    #Second Conv
    x = tf.keras.layers.Conv2D(filters = numFilters, kernel_size = (kernelSize,␣
↪kernelSize),
                              kernel_initializer = 'he_normal', padding =␣
↪'same') (x)
    if doBatchNorm:
        x = tf.keras.layers.BatchNormalization()(x)

    x = tf.keras.layers.Activation('relu')(x)

    return x


# Now defining Unet
def GiveMeUnet(inputImage, numFilters = 16, droupouts = 0.1, doBatchNorm =␣
↪True):
    # defining encoder Path
    c1 = Conv2dBlock(inputImage, numFilters * 1, kernelSize = 3, doBatchNorm =␣
↪doBatchNorm)
    p1 = tf.keras.layers.MaxPooling2D((2,2))(c1)
    p1 = tf.keras.layers.Dropout(droupouts)(p1)

    c2 = Conv2dBlock(p1, numFilters * 2, kernelSize = 3, doBatchNorm =␣
↪doBatchNorm)
    p2 = tf.keras.layers.MaxPooling2D((2,2))(c2)
    p2 = tf.keras.layers.Dropout(droupouts)(p2)

    c3 = Conv2dBlock(p2, numFilters * 4, kernelSize = 3, doBatchNorm =␣
↪doBatchNorm)
    p3 = tf.keras.layers.MaxPooling2D((2,2))(c3)
    p3 = tf.keras.layers.Dropout(droupouts)(p3)

    c4 = Conv2dBlock(p3, numFilters * 8, kernelSize = 3, doBatchNorm =␣
↪doBatchNorm)
```

```
    p4 = tf.keras.layers.MaxPooling2D((2,2))(c4)
    p4 = tf.keras.layers.Dropout(droupouts)(p4)

    c5 = Conv2dBlock(p4, numFilters * 16, kernelSize = 3, doBatchNorm =␣
↪doBatchNorm)

    # defining decoder path
    u6 = tf.keras.layers.Conv2DTranspose(numFilters*8, (3, 3), strides = (2,␣
↪2), padding = 'same')(c5)
    u6 = tf.keras.layers.concatenate([u6, c4])
    u6 = tf.keras.layers.Dropout(droupouts)(u6)
    c6 = Conv2dBlock(u6, numFilters * 8, kernelSize = 3, doBatchNorm =␣
↪doBatchNorm)

    u7 = tf.keras.layers.Conv2DTranspose(numFilters*4, (3, 3), strides = (2,␣
↪2), padding = 'same')(c6)

    u7 = tf.keras.layers.concatenate([u7, c3])
    u7 = tf.keras.layers.Dropout(droupouts)(u7)
    c7 = Conv2dBlock(u7, numFilters * 4, kernelSize = 3, doBatchNorm =␣
↪doBatchNorm)

    u8 = tf.keras.layers.Conv2DTranspose(numFilters*2, (3, 3), strides = (2,␣
↪2), padding = 'same')(c7)
    u8 = tf.keras.layers.concatenate([u8, c2])
    u8 = tf.keras.layers.Dropout(droupouts)(u8)
    c8 = Conv2dBlock(u8, numFilters * 2, kernelSize = 3, doBatchNorm =␣
↪doBatchNorm)

    u9 = tf.keras.layers.Conv2DTranspose(numFilters*1, (3, 3), strides = (2,␣
↪2), padding = 'same')(c8)
    u9 = tf.keras.layers.concatenate([u9, c1])
    u9 = tf.keras.layers.Dropout(droupouts)(u9)
    c9 = Conv2dBlock(u9, numFilters * 1, kernelSize = 3, doBatchNorm =␣
↪doBatchNorm)

    output = tf.keras.layers.Conv2D(1, (1, 1), activation = 'sigmoid')(c9)
    model = tf.keras.Model(inputs = [inputImage], outputs = [output])

    return model
```

[97]:
```
def dice_loss(y_true, y_pred):
    numerator = tf.reduce_sum(y_true * y_pred)
    denominator = tf.reduce_sum(y_true * y_true) + tf.reduce_sum(y_pred *␣
↪y_pred) - tf.reduce_sum(y_true * y_pred)
```

```
    return 1 - numerator / denominator
```

[112]:
```
smooth =100
def iou(y_true, y_pred):
    intersection = K.sum(y_true * y_pred)
    sum_ = K.sum(y_true + y_pred)
    jac = (intersection + smooth) / (sum_ - intersection + smooth)
    return jac
```

[98]:
```
def jacard_coef(y_true, y_pred):
    y_true_f = K.flatten(y_true)
    y_pred_f = K.flatten(y_pred)
    intersection = K.sum(y_true_f * y_pred_f)
    return (intersection + 1.0) / (K.sum(y_true_f) + K.sum(y_pred_f) -␣
 ↪intersection + 1.0)
```

[99]:
```
metrics=['accuracy', jacard_coef, iou]

inputs = tf.keras.layers.Input((H, W, CH))
model = GiveMeUnet(inputs, droupouts= 0.07)
model.compile(optimizer = 'Adam', loss =  dice_loss, metrics =metrics)#loss =␣
 ↪binary_crossentropy "binary_accuracy",
```

4 Model Summary

[100]:
```
#Summary of model------------------
model.summary()


#Plot of model------------------
dot_img_file = 'model.png'
plot_model(model, to_file=dot_img_file, show_shapes=True)
```

```
Model: "model"

________________________________________________________________________________
__________________
 Layer (type)                   Output Shape         Param #     Connected to
================================================================================
==================
 input_1 (InputLayer)           [(None, 128, 128, 3  0           []
                                )]

 conv2d (Conv2D)                (None, 128, 128, 16  448
['input_1[0][0]']
                                )
```

```
 batch_normalization (BatchNorm  (None, 128, 128, 16  64
['conv2d[0][0]']
 alization)                     )

 activation (Activation)        (None, 128, 128, 16  0
['batch_normalization[0][0]']
                                )

 conv2d_1 (Conv2D)              (None, 128, 128, 16  2320
['activation[0][0]']
                                )

 batch_normalization_1 (BatchNo  (None, 128, 128, 16  64
['conv2d_1[0][0]']
 rmalization)                   )

 activation_1 (Activation)      (None, 128, 128, 16  0
['batch_normalization_1[0][0]']
                                )

 max_pooling2d (MaxPooling2D)   (None, 64, 64, 16)   0
['activation_1[0][0]']

 dropout (Dropout)              (None, 64, 64, 16)   0
['max_pooling2d[0][0]']

 conv2d_2 (Conv2D)              (None, 64, 64, 32)   4640
['dropout[0][0]']

 batch_normalization_2 (BatchNo  (None, 64, 64, 32)  128
['conv2d_2[0][0]']
 rmalization)

 activation_2 (Activation)      (None, 64, 64, 32)   0
['batch_normalization_2[0][0]']

 conv2d_3 (Conv2D)              (None, 64, 64, 32)   9248
['activation_2[0][0]']

 batch_normalization_3 (BatchNo  (None, 64, 64, 32)  128
['conv2d_3[0][0]']
 rmalization)

 activation_3 (Activation)      (None, 64, 64, 32)   0
['batch_normalization_3[0][0]']

 max_pooling2d_1 (MaxPooling2D)  (None, 32, 32, 32)  0
['activation_3[0][0]']
```

```
 dropout_1 (Dropout)            (None, 32, 32, 32)   0
['max_pooling2d_1[0][0]']

 conv2d_4 (Conv2D)              (None, 32, 32, 64)   18496
['dropout_1[0][0]']

 batch_normalization_4 (BatchNo  (None, 32, 32, 64)  256
['conv2d_4[0][0]']
 rmalization)

 activation_4 (Activation)      (None, 32, 32, 64)   0
['batch_normalization_4[0][0]']

 conv2d_5 (Conv2D)              (None, 32, 32, 64)   36928
['activation_4[0][0]']

 batch_normalization_5 (BatchNo  (None, 32, 32, 64)  256
['conv2d_5[0][0]']
 rmalization)

 activation_5 (Activation)      (None, 32, 32, 64)   0
['batch_normalization_5[0][0]']

 max_pooling2d_2 (MaxPooling2D)  (None, 16, 16, 64)  0
['activation_5[0][0]']

 dropout_2 (Dropout)            (None, 16, 16, 64)   0
['max_pooling2d_2[0][0]']

 conv2d_6 (Conv2D)              (None, 16, 16, 128)  73856
['dropout_2[0][0]']

 batch_normalization_6 (BatchNo  (None, 16, 16, 128)  512
['conv2d_6[0][0]']
 rmalization)

 activation_6 (Activation)      (None, 16, 16, 128)  0
['batch_normalization_6[0][0]']

 conv2d_7 (Conv2D)              (None, 16, 16, 128)  147584
['activation_6[0][0]']

 batch_normalization_7 (BatchNo  (None, 16, 16, 128)  512
['conv2d_7[0][0]']
 rmalization)

 activation_7 (Activation)      (None, 16, 16, 128)  0
```

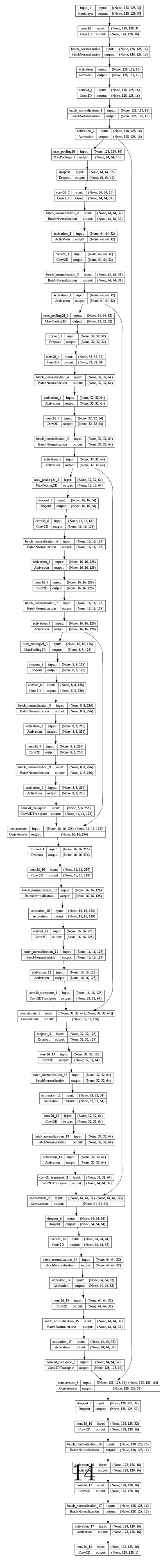
```
['batch_normalization_7[0][0]']

 max_pooling2d_3 (MaxPooling2D)  (None, 8, 8, 128)   0
['activation_7[0][0]']

 dropout_3 (Dropout)            (None, 8, 8, 128)    0
['max_pooling2d_3[0][0]']

 conv2d_8 (Conv2D)              (None, 8, 8, 256)    295168
['dropout_3[0][0]']

 batch_normalization_8 (BatchNo  (None, 8, 8, 256)   1024
['conv2d_8[0][0]']
 rmalization)

 activation_8 (Activation)      (None, 8, 8, 256)    0
['batch_normalization_8[0][0]']

 conv2d_9 (Conv2D)              (None, 8, 8, 256)    590080
['activation_8[0][0]']

 batch_normalization_9 (BatchNo  (None, 8, 8, 256)   1024
['conv2d_9[0][0]']
 rmalization)

 activation_9 (Activation)      (None, 8, 8, 256)    0
['batch_normalization_9[0][0]']

 conv2d_transpose (Conv2DTransp  (None, 16, 16, 128)  295040
['activation_9[0][0]']
 ose)

 concatenate (Concatenate)      (None, 16, 16, 256)  0
['conv2d_transpose[0][0]',
'activation_7[0][0]']

 dropout_4 (Dropout)            (None, 16, 16, 256)  0
['concatenate[0][0]']

 conv2d_10 (Conv2D)             (None, 16, 16, 128)  295040
['dropout_4[0][0]']

 batch_normalization_10 (BatchN  (None, 16, 16, 128)  512
['conv2d_10[0][0]']
 ormalization)

 activation_10 (Activation)     (None, 16, 16, 128)  0
['batch_normalization_10[0][0]']
```

```
 conv2d_11 (Conv2D)             (None, 16, 16, 128)  147584
['activation_10[0][0]']

 batch_normalization_11 (BatchN  (None, 16, 16, 128)  512
['conv2d_11[0][0]']
 ormalization)

 activation_11 (Activation)     (None, 16, 16, 128)  0
['batch_normalization_11[0][0]']

 conv2d_transpose_1 (Conv2DTran  (None, 32, 32, 64)  73792
['activation_11[0][0]']
 spose)

 concatenate_1 (Concatenate)    (None, 32, 32, 128)  0
['conv2d_transpose_1[0][0]',
'activation_5[0][0]']

 dropout_5 (Dropout)            (None, 32, 32, 128)  0
['concatenate_1[0][0]']

 conv2d_12 (Conv2D)             (None, 32, 32, 64)   73792
['dropout_5[0][0]']

 batch_normalization_12 (BatchN  (None, 32, 32, 64)  256
['conv2d_12[0][0]']
 ormalization)

 activation_12 (Activation)     (None, 32, 32, 64)   0
['batch_normalization_12[0][0]']

 conv2d_13 (Conv2D)             (None, 32, 32, 64)   36928
['activation_12[0][0]']

 batch_normalization_13 (BatchN  (None, 32, 32, 64)  256
['conv2d_13[0][0]']
 ormalization)

 activation_13 (Activation)     (None, 32, 32, 64)   0
['batch_normalization_13[0][0]']

 conv2d_transpose_2 (Conv2DTran  (None, 64, 64, 32)  18464
['activation_13[0][0]']
 spose)

 concatenate_2 (Concatenate)    (None, 64, 64, 64)   0
['conv2d_transpose_2[0][0]',
```

```
'activation_3[0][0]']

 dropout_6 (Dropout)            (None, 64, 64, 64)   0
['concatenate_2[0][0]']

 conv2d_14 (Conv2D)             (None, 64, 64, 32)   18464
['dropout_6[0][0]']

 batch_normalization_14 (BatchN  (None, 64, 64, 32)  128
['conv2d_14[0][0]']
 ormalization)

 activation_14 (Activation)     (None, 64, 64, 32)   0
['batch_normalization_14[0][0]']

 conv2d_15 (Conv2D)             (None, 64, 64, 32)   9248
['activation_14[0][0]']

 batch_normalization_15 (BatchN  (None, 64, 64, 32)  128
['conv2d_15[0][0]']
 ormalization)

 activation_15 (Activation)     (None, 64, 64, 32)   0
['batch_normalization_15[0][0]']

 conv2d_transpose_3 (Conv2DTran  (None, 128, 128, 16  4624
['activation_15[0][0]']
 spose)                         )

 concatenate_3 (Concatenate)    (None, 128, 128, 32  0
['conv2d_transpose_3[0][0]',
                                )
'activation_1[0][0]']

 dropout_7 (Dropout)            (None, 128, 128, 32  0
['concatenate_3[0][0]']
                                )

 conv2d_16 (Conv2D)             (None, 128, 128, 16  4624
['dropout_7[0][0]']
                                )

 batch_normalization_16 (BatchN  (None, 128, 128, 16  64
['conv2d_16[0][0]']
 ormalization)                  )

 activation_16 (Activation)     (None, 128, 128, 16  0
['batch_normalization_16[0][0]']
```

```
                                )

 conv2d_17 (Conv2D)             (None, 128, 128, 16  2320
['activation_16[0][0]']
                                )

 batch_normalization_17 (BatchN  (None, 128, 128, 16  64
['conv2d_17[0][0]']
 ormalization)                  )

 activation_17 (Activation)     (None, 128, 128, 16  0
['batch_normalization_17[0][0]']
                                )

 conv2d_18 (Conv2D)             (None, 128, 128, 1)  17
['activation_17[0][0]']

================================================================================
==================
Total params: 2,164,593
Trainable params: 2,161,649
Non-trainable params: 2,944
--------------------------------------------------------------------------------
------------------
```

[100]:



[101]:

```python
#Train model-----------------------------
nbatch_size=128
nepochs=200
history = model.fit(X_train,Y_train,batch_size=nbatch_size,
                    epochs=nepochs,validation_split=0.2,shuffle=True,
                    max_queue_size=32,workers=4,use_multiprocessing=True,
                   )
```

```
Epoch 1/200
2/2 [==============================] - 22s 1s/step - loss: 0.8720 - accuracy:
0.3560 - jacard_coef: 0.0937 - val_loss: 0.8787 - val_accuracy: 0.1291 -
val_jacard_coef: 0.0720
Epoch 2/200
2/2 [==============================] - 17s 1s/step - loss: 0.8465 - accuracy:
0.4166 - jacard_coef: 0.0838 - val_loss: 0.9006 - val_accuracy: 0.0789 -
val_jacard_coef: 0.0788
Epoch 3/200
2/2 [==============================] - 16s 1s/step - loss: 0.8261 - accuracy:
0.4976 - jacard_coef: 0.0785 - val_loss: 0.9140 - val_accuracy: 0.0749 -
val_jacard_coef: 0.0777
Epoch 4/200
2/2 [==============================] - 16s 959ms/step - loss: 0.8076 - accuracy:
0.5952 - jacard_coef: 0.1011 - val_loss: 0.9098 - val_accuracy: 0.0751 -
val_jacard_coef: 0.0762
Epoch 5/200
2/2 [==============================] - 18s 1s/step - loss: 0.7864 - accuracy:
0.6928 - jacard_coef: 0.1392 - val_loss: 0.9156 - val_accuracy: 0.0753 -
val_jacard_coef: 0.0769
Epoch 6/200
2/2 [==============================] - 18s 1s/step - loss: 0.7700 - accuracy:
0.7597 - jacard_coef: 0.1559 - val_loss: 0.9163 - val_accuracy: 0.0762 -
val_jacard_coef: 0.0785
Epoch 7/200
2/2 [==============================] - 19s 1s/step - loss: 0.7723 - accuracy:
0.7702 - jacard_coef: 0.1627 - val_loss: 0.9153 - val_accuracy: 0.0778 -
val_jacard_coef: 0.0805
Epoch 8/200
2/2 [==============================] - 16s 820ms/step - loss: 0.7773 - accuracy:
0.7679 - jacard_coef: 0.1221 - val_loss: 0.9200 - val_accuracy: 0.0782 -
val_jacard_coef: 0.0781
Epoch 9/200
2/2 [==============================] - 16s 886ms/step - loss: 0.7647 - accuracy:
0.7809 - jacard_coef: 0.1383 - val_loss: 0.9182 - val_accuracy: 0.0835 -
val_jacard_coef: 0.0790
Epoch 10/200
```

```
2/2 [==============================] - 17s 1s/step - loss: 0.7376 - accuracy:
0.7891 - jacard_coef: 0.1568 - val_loss: 0.9121 - val_accuracy: 0.1073 -
val_jacard_coef: 0.0763
Epoch 11/200
2/2 [==============================] - 17s 1s/step - loss: 0.7269 - accuracy:
0.7891 - jacard_coef: 0.1879 - val_loss: 0.9026 - val_accuracy: 0.1521 -
val_jacard_coef: 0.0644
Epoch 12/200
2/2 [==============================] - 17s 866ms/step - loss: 0.7442 - accuracy:
0.7797 - jacard_coef: 0.1033 - val_loss: 0.8930 - val_accuracy: 0.1880 -
val_jacard_coef: 0.0670
Epoch 13/200
2/2 [==============================] - 17s 1s/step - loss: 0.7436 - accuracy:
0.7771 - jacard_coef: 0.0942 - val_loss: 0.8998 - val_accuracy: 0.1636 -
val_jacard_coef: 0.0821
Epoch 14/200
2/2 [==============================] - 19s 1s/step - loss: 0.7140 - accuracy:
0.8003 - jacard_coef: 0.1663 - val_loss: 0.9198 - val_accuracy: 0.1029 -
val_jacard_coef: 0.0784
Epoch 15/200
2/2 [==============================] - 17s 1s/step - loss: 0.7129 - accuracy:
0.8193 - jacard_coef: 0.1041 - val_loss: 0.9237 - val_accuracy: 0.0835 -
val_jacard_coef: 0.0760
Epoch 16/200
2/2 [==============================] - 18s 980ms/step - loss: 0.7195 - accuracy:
0.8307 - jacard_coef: 0.1230 - val_loss: 0.9229 - val_accuracy: 0.0939 -
val_jacard_coef: 0.0765
Epoch 17/200
2/2 [==============================] - 20s 1s/step - loss: 0.7096 - accuracy:
0.8429 - jacard_coef: 0.1475 - val_loss: 0.9184 - val_accuracy: 0.1409 -
val_jacard_coef: 0.0795
Epoch 18/200
2/2 [==============================] - 20s 1s/step - loss: 0.6902 - accuracy:
0.8549 - jacard_coef: 0.1417 - val_loss: 0.8994 - val_accuracy: 0.2829 -
val_jacard_coef: 0.0909
Epoch 19/200
2/2 [==============================] - 19s 1s/step - loss: 0.6845 - accuracy:
0.8562 - jacard_coef: 0.1817 - val_loss: 0.8845 - val_accuracy: 0.3986 -
val_jacard_coef: 0.0988
Epoch 20/200
2/2 [==============================] - 19s 1s/step - loss: 0.6849 - accuracy:
0.8531 - jacard_coef: 0.1532 - val_loss: 0.8944 - val_accuracy: 0.3495 -
val_jacard_coef: 0.0944
Epoch 21/200
2/2 [==============================] - 18s 931ms/step - loss: 0.6737 - accuracy:
0.8609 - jacard_coef: 0.1239 - val_loss: 0.9016 - val_accuracy: 0.3003 -
val_jacard_coef: 0.0908
Epoch 22/200
```

```
2/2 [==============================] - 19s 1s/step - loss: 0.6679 - accuracy:
0.8638 - jacard_coef: 0.1580 - val_loss: 0.9051 - val_accuracy: 0.2743 -
val_jacard_coef: 0.0889
Epoch 23/200
2/2 [==============================] - 19s 1s/step - loss: 0.6660 - accuracy:
0.8687 - jacard_coef: 0.2130 - val_loss: 0.9078 - val_accuracy: 0.2512 -
val_jacard_coef: 0.0871
Epoch 24/200
2/2 [==============================] - 19s 1s/step - loss: 0.6765 - accuracy:
0.8679 - jacard_coef: 0.1577 - val_loss: 0.9029 - val_accuracy: 0.2985 -
val_jacard_coef: 0.0906
Epoch 25/200
2/2 [==============================] - 17s 977ms/step - loss: 0.6823 - accuracy:
0.8659 - jacard_coef: 0.1194 - val_loss: 0.8927 - val_accuracy: 0.3785 -
val_jacard_coef: 0.0981
Epoch 26/200
2/2 [==============================] - 18s 1s/step - loss: 0.6684 - accuracy:
0.8715 - jacard_coef: 0.1426 - val_loss: 0.8945 - val_accuracy: 0.3744 -
val_jacard_coef: 0.0969
Epoch 27/200
2/2 [==============================] - 18s 1s/step - loss: 0.6510 - accuracy:
0.8779 - jacard_coef: 0.1419 - val_loss: 0.9040 - val_accuracy: 0.3040 -
val_jacard_coef: 0.0901
Epoch 28/200
2/2 [==============================] - 17s 966ms/step - loss: 0.6412 - accuracy:
0.8822 - jacard_coef: 0.2184 - val_loss: 0.9095 - val_accuracy: 0.2503 -
val_jacard_coef: 0.0862
Epoch 29/200
2/2 [==============================] - 18s 961ms/step - loss: 0.6362 - accuracy:
0.8851 - jacard_coef: 0.2239 - val_loss: 0.9191 - val_accuracy: 0.1405 -
val_jacard_coef: 0.0795
Epoch 30/200
2/2 [==============================] - 20s 1s/step - loss: 0.6270 - accuracy:
0.8877 - jacard_coef: 0.2136 - val_loss: 0.9238 - val_accuracy: 0.0841 -
val_jacard_coef: 0.0759
Epoch 31/200
2/2 [==============================] - 21s 1s/step - loss: 0.6356 - accuracy:
0.8848 - jacard_coef: 0.1317 - val_loss: 0.9233 - val_accuracy: 0.0902 -
val_jacard_coef: 0.0763
Epoch 32/200
2/2 [==============================] - 17s 971ms/step - loss: 0.6280 - accuracy:
0.8840 - jacard_coef: 0.1515 - val_loss: 0.9140 - val_accuracy: 0.1899 -
val_jacard_coef: 0.0835
Epoch 33/200
2/2 [==============================] - 19s 1s/step - loss: 0.6189 - accuracy:
0.8838 - jacard_coef: 0.2213 - val_loss: 0.8854 - val_accuracy: 0.4163 -
val_jacard_coef: 0.1049
Epoch 34/200
```

```
2/2 [==============================] - 19s 1s/step - loss: 0.6275 - accuracy:
0.8843 - jacard_coef: 0.2335 - val_loss: 0.8779 - val_accuracy: 0.4585 -
val_jacard_coef: 0.1099
Epoch 35/200
2/2 [==============================] - 19s 1s/step - loss: 0.6209 - accuracy:
0.8902 - jacard_coef: 0.1427 - val_loss: 0.8812 - val_accuracy: 0.4411 -
val_jacard_coef: 0.1073
Epoch 36/200
2/2 [==============================] - 19s 1s/step - loss: 0.6061 - accuracy:
0.8992 - jacard_coef: 0.1917 - val_loss: 0.8898 - val_accuracy: 0.3908 -
val_jacard_coef: 0.1013
Epoch 37/200
2/2 [==============================] - 21s 1s/step - loss: 0.5949 - accuracy:
0.9096 - jacard_coef: 0.2535 - val_loss: 0.9100 - val_accuracy: 0.2320 -
val_jacard_coef: 0.0866
Epoch 38/200
2/2 [==============================] - 22s 1s/step - loss: 0.5921 - accuracy:
0.9136 - jacard_coef: 0.1594 - val_loss: 0.9196 - val_accuracy: 0.1343 -
val_jacard_coef: 0.0793
Epoch 39/200
2/2 [==============================] - 22s 2s/step - loss: 0.5982 - accuracy:
0.9100 - jacard_coef: 0.1687 - val_loss: 0.9204 - val_accuracy: 0.1238 -
val_jacard_coef: 0.0786
Epoch 40/200
2/2 [==============================] - 20s 1s/step - loss: 0.5956 - accuracy:
0.9073 - jacard_coef: 0.2076 - val_loss: 0.9207 - val_accuracy: 0.1198 -
val_jacard_coef: 0.0784
Epoch 41/200
2/2 [==============================] - 20s 1s/step - loss: 0.5965 - accuracy:
0.9054 - jacard_coef: 0.2225 - val_loss: 0.9218 - val_accuracy: 0.1070 -
val_jacard_coef: 0.0775
Epoch 42/200
2/2 [==============================] - 21s 1s/step - loss: 0.5987 - accuracy:
0.9039 - jacard_coef: 0.1805 - val_loss: 0.9220 - val_accuracy: 0.1039 -
val_jacard_coef: 0.0774
Epoch 43/200
2/2 [==============================] - 22s 1s/step - loss: 0.6020 - accuracy:
0.9017 - jacard_coef: 0.2183 - val_loss: 0.9209 - val_accuracy: 0.1163 -
val_jacard_coef: 0.0782
Epoch 44/200
2/2 [==============================] - 23s 1s/step - loss: 0.5945 - accuracy:
0.9038 - jacard_coef: 0.2013 - val_loss: 0.9182 - val_accuracy: 0.1442 -
val_jacard_coef: 0.0802
Epoch 45/200
2/2 [==============================] - 24s 2s/step - loss: 0.5802 - accuracy:
0.9078 - jacard_coef: 0.2330 - val_loss: 0.9108 - val_accuracy: 0.2143 -
val_jacard_coef: 0.0857
Epoch 46/200
```

```
2/2 [==============================] - 23s 1s/step - loss: 0.5853 - accuracy:
0.9068 - jacard_coef: 0.1527 - val_loss: 0.8908 - val_accuracy: 0.3740 -
val_jacard_coef: 0.1000
Epoch 47/200
2/2 [==============================] - 20s 1s/step - loss: 0.6004 - accuracy:
0.9052 - jacard_coef: 0.1764 - val_loss: 0.8904 - val_accuracy: 0.3783 -
val_jacard_coef: 0.1004
Epoch 48/200
2/2 [==============================] - 21s 1s/step - loss: 0.5793 - accuracy:
0.9135 - jacard_coef: 0.1786 - val_loss: 0.9026 - val_accuracy: 0.2884 -
val_jacard_coef: 0.0915
Epoch 49/200
2/2 [==============================] - 21s 2s/step - loss: 0.5703 - accuracy:
0.9170 - jacard_coef: 0.1990 - val_loss: 0.9129 - val_accuracy: 0.1992 -
val_jacard_coef: 0.0841
Epoch 50/200
2/2 [==============================] - 21s 1s/step - loss: 0.5724 - accuracy:
0.9166 - jacard_coef: 0.1987 - val_loss: 0.9164 - val_accuracy: 0.1666 -
val_jacard_coef: 0.0817
Epoch 51/200
2/2 [==============================] - 18s 874ms/step - loss: 0.5665 - accuracy:
0.9181 - jacard_coef: 0.2404 - val_loss: 0.9163 - val_accuracy: 0.1692 -
val_jacard_coef: 0.0819
Epoch 52/200
2/2 [==============================] - 16s 1s/step - loss: 0.5648 - accuracy:
0.9182 - jacard_coef: 0.1801 - val_loss: 0.9151 - val_accuracy: 0.1826 -
val_jacard_coef: 0.0828
Epoch 53/200
2/2 [==============================] - 20s 1s/step - loss: 0.5703 - accuracy:
0.9157 - jacard_coef: 0.2046 - val_loss: 0.9119 - val_accuracy: 0.2142 -
val_jacard_coef: 0.0852
Epoch 54/200
2/2 [==============================] - 22s 1s/step - loss: 0.5589 - accuracy:
0.9189 - jacard_coef: 0.2471 - val_loss: 0.9120 - val_accuracy: 0.2131 -
val_jacard_coef: 0.0850
Epoch 55/200
2/2 [==============================] - 24s 1s/step - loss: 0.5478 - accuracy:
0.9217 - jacard_coef: 0.2245 - val_loss: 0.9188 - val_accuracy: 0.1391 -
val_jacard_coef: 0.0796
Epoch 56/200
2/2 [==============================] - 24s 1s/step - loss: 0.5494 - accuracy:
0.9204 - jacard_coef: 0.2055 - val_loss: 0.9213 - val_accuracy: 0.1092 -
val_jacard_coef: 0.0777
Epoch 57/200
2/2 [==============================] - 29s 2s/step - loss: 0.5517 - accuracy:
0.9186 - jacard_coef: 0.2064 - val_loss: 0.9211 - val_accuracy: 0.1108 -
val_jacard_coef: 0.0778
Epoch 58/200
```

```
2/2 [==============================] - 22s 1s/step - loss: 0.5510 - accuracy:
0.9180 - jacard_coef: 0.2507 - val_loss: 0.9189 - val_accuracy: 0.1338 -
val_jacard_coef: 0.0795
Epoch 59/200
2/2 [==============================] - 25s 1s/step - loss: 0.5552 - accuracy:
0.9163 - jacard_coef: 0.2900 - val_loss: 0.9148 - val_accuracy: 0.1753 -
val_jacard_coef: 0.0825
Epoch 60/200
2/2 [==============================] - 23s 1s/step - loss: 0.5735 - accuracy:
0.9088 - jacard_coef: 0.2161 - val_loss: 0.9078 - val_accuracy: 0.2395 -
val_jacard_coef: 0.0876
Epoch 61/200
2/2 [==============================] - 25s 2s/step - loss: 0.5791 - accuracy:
0.9039 - jacard_coef: 0.1988 - val_loss: 0.9035 - val_accuracy: 0.2771 -
val_jacard_coef: 0.0908
Epoch 62/200
2/2 [==============================] - 24s 1s/step - loss: 0.5682 - accuracy:
0.9055 - jacard_coef: 0.2474 - val_loss: 0.9018 - val_accuracy: 0.2929 -
val_jacard_coef: 0.0921
Epoch 63/200
2/2 [==============================] - 24s 2s/step - loss: 0.5493 - accuracy:
0.9096 - jacard_coef: 0.2006 - val_loss: 0.8986 - val_accuracy: 0.3202 -
val_jacard_coef: 0.0944
Epoch 64/200
2/2 [==============================] - 25s 1s/step - loss: 0.5418 - accuracy:
0.9129 - jacard_coef: 0.2916 - val_loss: 0.8974 - val_accuracy: 0.3296 -
val_jacard_coef: 0.0952
Epoch 65/200
2/2 [==============================] - 21s 1s/step - loss: 0.5359 - accuracy:
0.9163 - jacard_coef: 0.2174 - val_loss: 0.8953 - val_accuracy: 0.3427 -
val_jacard_coef: 0.0966
Epoch 66/200
2/2 [==============================] - 23s 2s/step - loss: 0.5270 - accuracy:
0.9199 - jacard_coef: 0.2969 - val_loss: 0.8925 - val_accuracy: 0.3606 -
val_jacard_coef: 0.0986
Epoch 67/200
2/2 [==============================] - 24s 2s/step - loss: 0.5224 - accuracy:
0.9215 - jacard_coef: 0.1965 - val_loss: 0.8909 - val_accuracy: 0.3709 -
val_jacard_coef: 0.0997
Epoch 68/200
2/2 [==============================] - 25s 1s/step - loss: 0.5165 - accuracy:
0.9232 - jacard_coef: 0.2594 - val_loss: 0.8789 - val_accuracy: 0.4421 -
val_jacard_coef: 0.1078
Epoch 69/200
2/2 [==============================] - 22s 961ms/step - loss: 0.5204 - accuracy:
0.9233 - jacard_coef: 0.2327 - val_loss: 0.8540 - val_accuracy: 0.5535 -
val_jacard_coef: 0.1239
Epoch 70/200
```

```
2/2 [==============================] - 20s 1s/step - loss: 0.5368 - accuracy:
0.9207 - jacard_coef: 0.1522 - val_loss: 0.8274 - val_accuracy: 0.6428 -
val_jacard_coef: 0.1388
Epoch 71/200
2/2 [==============================] - 22s 1s/step - loss: 0.5439 - accuracy:
0.9196 - jacard_coef: 0.1907 - val_loss: 0.8107 - val_accuracy: 0.6836 -
val_jacard_coef: 0.1482
Epoch 72/200
2/2 [==============================] - 24s 1s/step - loss: 0.5360 - accuracy:
0.9224 - jacard_coef: 0.1562 - val_loss: 0.8067 - val_accuracy: 0.6900 -
val_jacard_coef: 0.1508
Epoch 73/200
2/2 [==============================] - 19s 938ms/step - loss: 0.5166 - accuracy:
0.9270 - jacard_coef: 0.2329 - val_loss: 0.8098 - val_accuracy: 0.6819 -
val_jacard_coef: 0.1494
Epoch 74/200
2/2 [==============================] - 20s 1s/step - loss: 0.5146 - accuracy:
0.9279 - jacard_coef: 0.2020 - val_loss: 0.8169 - val_accuracy: 0.6669 -
val_jacard_coef: 0.1449
Epoch 75/200
2/2 [==============================] - 20s 1s/step - loss: 0.5242 - accuracy:
0.9265 - jacard_coef: 0.1686 - val_loss: 0.8250 - val_accuracy: 0.6511 -
val_jacard_coef: 0.1391
Epoch 76/200
2/2 [==============================] - 18s 1s/step - loss: 0.5291 - accuracy:
0.9259 - jacard_coef: 0.2121 - val_loss: 0.8367 - val_accuracy: 0.6210 -
val_jacard_coef: 0.1321
Epoch 77/200
2/2 [==============================] - 22s 1s/step - loss: 0.5350 - accuracy:
0.9248 - jacard_coef: 0.2397 - val_loss: 0.8453 - val_accuracy: 0.5912 -
val_jacard_coef: 0.1283
Epoch 78/200
2/2 [==============================] - 20s 2s/step - loss: 0.5254 - accuracy:
0.9262 - jacard_coef: 0.2817 - val_loss: 0.8453 - val_accuracy: 0.5863 -
val_jacard_coef: 0.1297
Epoch 79/200
2/2 [==============================] - 19s 936ms/step - loss: 0.5048 - accuracy:
0.9284 - jacard_coef: 0.2685 - val_loss: 0.8397 - val_accuracy: 0.6001 -
val_jacard_coef: 0.1348
Epoch 80/200
2/2 [==============================] - 17s 1s/step - loss: 0.5053 - accuracy:
0.9252 - jacard_coef: 0.2667 - val_loss: 0.8400 - val_accuracy: 0.5985 -
val_jacard_coef: 0.1356
Epoch 81/200
2/2 [==============================] - 19s 983ms/step - loss: 0.5072 - accuracy:
0.9229 - jacard_coef: 0.2789 - val_loss: 0.8481 - val_accuracy: 0.5737 -
val_jacard_coef: 0.1307
Epoch 82/200
```

```
2/2 [==============================] - 19s 957ms/step - loss: 0.5006 - accuracy:
0.9231 - jacard_coef: 0.3120 - val_loss: 0.8564 - val_accuracy: 0.5436 -
val_jacard_coef: 0.1255
Epoch 83/200
2/2 [==============================] - 17s 1s/step - loss: 0.4976 - accuracy:
0.9227 - jacard_coef: 0.2278 - val_loss: 0.8820 - val_accuracy: 0.4252 -
val_jacard_coef: 0.1078
Epoch 84/200
2/2 [==============================] - 21s 1s/step - loss: 0.4986 - accuracy:
0.9215 - jacard_coef: 0.3187 - val_loss: 0.9110 - val_accuracy: 0.2146 -
val_jacard_coef: 0.0860
Epoch 85/200
2/2 [==============================] - 21s 1s/step - loss: 0.5270 - accuracy:
0.9147 - jacard_coef: 0.3250 - val_loss: 0.9174 - val_accuracy: 0.1524 -
val_jacard_coef: 0.0809
Epoch 86/200
2/2 [==============================] - 28s 2s/step - loss: 0.5402 - accuracy:
0.9111 - jacard_coef: 0.2980 - val_loss: 0.9139 - val_accuracy: 0.1875 -
val_jacard_coef: 0.0837
Epoch 87/200
2/2 [==============================] - 20s 973ms/step - loss: 0.5290 - accuracy:
0.9124 - jacard_coef: 0.2703 - val_loss: 0.9012 - val_accuracy: 0.2967 -
val_jacard_coef: 0.0937
Epoch 88/200
2/2 [==============================] - 23s 2s/step - loss: 0.5243 - accuracy:
0.9147 - jacard_coef: 0.2229 - val_loss: 0.8868 - val_accuracy: 0.3957 -
val_jacard_coef: 0.1041
Epoch 89/200
2/2 [==============================] - 24s 1s/step - loss: 0.5239 - accuracy:
0.9174 - jacard_coef: 0.3106 - val_loss: 0.8705 - val_accuracy: 0.4841 -
val_jacard_coef: 0.1152
Epoch 90/200
2/2 [==============================] - 19s 1s/step - loss: 0.5150 - accuracy:
0.9211 - jacard_coef: 0.2827 - val_loss: 0.8550 - val_accuracy: 0.5471 -
val_jacard_coef: 0.1250
Epoch 91/200
2/2 [==============================] - 19s 1s/step - loss: 0.5097 - accuracy:
0.9235 - jacard_coef: 0.2817 - val_loss: 0.8419 - val_accuracy: 0.5905 -
val_jacard_coef: 0.1329
Epoch 92/200
2/2 [==============================] - 21s 1s/step - loss: 0.5044 - accuracy:
0.9262 - jacard_coef: 0.3212 - val_loss: 0.8388 - val_accuracy: 0.5998 -
val_jacard_coef: 0.1350
Epoch 93/200
2/2 [==============================] - 20s 1s/step - loss: 0.5001 - accuracy:
0.9266 - jacard_coef: 0.3034 - val_loss: 0.8490 - val_accuracy: 0.5651 -
val_jacard_coef: 0.1293
Epoch 94/200
```

```
2/2 [==============================] - 24s 2s/step - loss: 0.4931 - accuracy:
0.9252 - jacard_coef: 0.3097 - val_loss: 0.8605 - val_accuracy: 0.5178 -
val_jacard_coef: 0.1223
Epoch 95/200
2/2 [==============================] - 22s 1s/step - loss: 0.4918 - accuracy:
0.9249 - jacard_coef: 0.2023 - val_loss: 0.8685 - val_accuracy: 0.4807 -
val_jacard_coef: 0.1170
Epoch 96/200
2/2 [==============================] - 19s 974ms/step - loss: 0.4968 - accuracy:
0.9234 - jacard_coef: 0.2875 - val_loss: 0.8710 - val_accuracy: 0.4685 -
val_jacard_coef: 0.1155
Epoch 97/200
2/2 [==============================] - 19s 1s/step - loss: 0.4974 - accuracy:
0.9232 - jacard_coef: 0.2217 - val_loss: 0.8734 - val_accuracy: 0.4589 -
val_jacard_coef: 0.1144
Epoch 98/200
2/2 [==============================] - 22s 1s/step - loss: 0.4926 - accuracy:
0.9238 - jacard_coef: 0.2619 - val_loss: 0.8774 - val_accuracy: 0.4414 -
val_jacard_coef: 0.1116
Epoch 99/200
2/2 [==============================] - 23s 1s/step - loss: 0.5086 - accuracy:
0.9211 - jacard_coef: 0.3137 - val_loss: 0.8811 - val_accuracy: 0.4218 -
val_jacard_coef: 0.1085
Epoch 100/200
2/2 [==============================] - 21s 1s/step - loss: 0.5036 - accuracy:
0.9234 - jacard_coef: 0.3024 - val_loss: 0.8798 - val_accuracy: 0.4270 -
val_jacard_coef: 0.1088
Epoch 101/200
2/2 [==============================] - 22s 1s/step - loss: 0.4772 - accuracy:
0.9284 - jacard_coef: 0.2489 - val_loss: 0.8633 - val_accuracy: 0.5051 -
val_jacard_coef: 0.1201
Epoch 102/200
2/2 [==============================] - 21s 1s/step - loss: 0.4683 - accuracy:
0.9300 - jacard_coef: 0.2119 - val_loss: 0.8357 - val_accuracy: 0.6011 -
val_jacard_coef: 0.1392
Epoch 103/200
2/2 [==============================] - 20s 1s/step - loss: 0.4676 - accuracy:
0.9302 - jacard_coef: 0.2713 - val_loss: 0.8155 - val_accuracy: 0.6542 -
val_jacard_coef: 0.1518
Epoch 104/200
2/2 [==============================] - 22s 989ms/step - loss: 0.4664 - accuracy:
0.9311 - jacard_coef: 0.3346 - val_loss: 0.8110 - val_accuracy: 0.6636 -
val_jacard_coef: 0.1542
Epoch 105/200
2/2 [==============================] - 19s 1s/step - loss: 0.4653 - accuracy:
0.9323 - jacard_coef: 0.2663 - val_loss: 0.8143 - val_accuracy: 0.6559 -
val_jacard_coef: 0.1521
Epoch 106/200
```

```
2/2 [==============================] - 21s 1s/step - loss: 0.4655 - accuracy:
0.9319 - jacard_coef: 0.3352 - val_loss: 0.8283 - val_accuracy: 0.6224 -
val_jacard_coef: 0.1428
Epoch 107/200
2/2 [==============================] - 19s 1s/step - loss: 0.4731 - accuracy:
0.9293 - jacard_coef: 0.3521 - val_loss: 0.8441 - val_accuracy: 0.5768 -
val_jacard_coef: 0.1326
Epoch 108/200
2/2 [==============================] - 20s 1s/step - loss: 0.4924 - accuracy:
0.9222 - jacard_coef: 0.2858 - val_loss: 0.8555 - val_accuracy: 0.5360 -
val_jacard_coef: 0.1256
Epoch 109/200
2/2 [==============================] - 21s 1s/step - loss: 0.5054 - accuracy:
0.9147 - jacard_coef: 0.2085 - val_loss: 0.8631 - val_accuracy: 0.5052 -
val_jacard_coef: 0.1207
Epoch 110/200
2/2 [==============================] - 18s 1s/step - loss: 0.4994 - accuracy:
0.9141 - jacard_coef: 0.2736 - val_loss: 0.8628 - val_accuracy: 0.5060 -
val_jacard_coef: 0.1217
Epoch 111/200
2/2 [==============================] - 18s 1s/step - loss: 0.4804 - accuracy:
0.9185 - jacard_coef: 0.2380 - val_loss: 0.8540 - val_accuracy: 0.5405 -
val_jacard_coef: 0.1289
Epoch 112/200
2/2 [==============================] - 18s 1s/step - loss: 0.4709 - accuracy:
0.9215 - jacard_coef: 0.3077 - val_loss: 0.8416 - val_accuracy: 0.5826 -
val_jacard_coef: 0.1384
Epoch 113/200
2/2 [==============================] - 19s 1s/step - loss: 0.4640 - accuracy:
0.9246 - jacard_coef: 0.3507 - val_loss: 0.8429 - val_accuracy: 0.5805 -
val_jacard_coef: 0.1375
Epoch 114/200
2/2 [==============================] - 18s 1s/step - loss: 0.4600 - accuracy:
0.9279 - jacard_coef: 0.2874 - val_loss: 0.8598 - val_accuracy: 0.5203 -
val_jacard_coef: 0.1250
Epoch 115/200
2/2 [==============================] - 17s 1s/step - loss: 0.4794 - accuracy:
0.9255 - jacard_coef: 0.3268 - val_loss: 0.8795 - val_accuracy: 0.4229 -
val_jacard_coef: 0.1104
Epoch 116/200
2/2 [==============================] - 18s 976ms/step - loss: 0.4794 - accuracy:
0.9259 - jacard_coef: 0.3313 - val_loss: 0.8945 - val_accuracy: 0.3329 -
val_jacard_coef: 0.0990
Epoch 117/200
2/2 [==============================] - 20s 986ms/step - loss: 0.4720 - accuracy:
0.9263 - jacard_coef: 0.3363 - val_loss: 0.9016 - val_accuracy: 0.2840 -
val_jacard_coef: 0.0935
Epoch 118/200
```

```
2/2 [==============================] - 19s 1s/step - loss: 0.4659 - accuracy:
0.9264 - jacard_coef: 0.3079 - val_loss: 0.9039 - val_accuracy: 0.2658 -
val_jacard_coef: 0.0916
Epoch 119/200
2/2 [==============================] - 22s 1s/step - loss: 0.4552 - accuracy:
0.9287 - jacard_coef: 0.2971 - val_loss: 0.9019 - val_accuracy: 0.2816 -
val_jacard_coef: 0.0931
Epoch 120/200
2/2 [==============================] - 18s 1s/step - loss: 0.4467 - accuracy:
0.9307 - jacard_coef: 0.2771 - val_loss: 0.8998 - val_accuracy: 0.2976 -
val_jacard_coef: 0.0947
Epoch 121/200
2/2 [==============================] - 20s 1s/step - loss: 0.4469 - accuracy:
0.9316 - jacard_coef: 0.3475 - val_loss: 0.9005 - val_accuracy: 0.2923 -
val_jacard_coef: 0.0941
Epoch 122/200
2/2 [==============================] - 21s 1s/step - loss: 0.4565 - accuracy:
0.9301 - jacard_coef: 0.2895 - val_loss: 0.9001 - val_accuracy: 0.2928 -
val_jacard_coef: 0.0943
Epoch 123/200
2/2 [==============================] - 20s 1s/step - loss: 0.4650 - accuracy:
0.9292 - jacard_coef: 0.3293 - val_loss: 0.8994 - val_accuracy: 0.2944 -
val_jacard_coef: 0.0946
Epoch 124/200
2/2 [==============================] - 19s 1s/step - loss: 0.4696 - accuracy:
0.9303 - jacard_coef: 0.2770 - val_loss: 0.8948 - val_accuracy: 0.3241 -
val_jacard_coef: 0.0982
Epoch 125/200
2/2 [==============================] - 22s 1s/step - loss: 0.4743 - accuracy:
0.9308 - jacard_coef: 0.3263 - val_loss: 0.8715 - val_accuracy: 0.4534 -
val_jacard_coef: 0.1157
Epoch 126/200
2/2 [==============================] - 22s 1s/step - loss: 0.4657 - accuracy:
0.9317 - jacard_coef: 0.2402 - val_loss: 0.8090 - val_accuracy: 0.6596 -
val_jacard_coef: 0.1598
Epoch 127/200
2/2 [==============================] - 20s 1s/step - loss: 0.4629 - accuracy:
0.9321 - jacard_coef: 0.2765 - val_loss: 0.7370 - val_accuracy: 0.7842 -
val_jacard_coef: 0.2043
Epoch 128/200
2/2 [==============================] - 20s 998ms/step - loss: 0.4839 - accuracy:
0.9291 - jacard_coef: 0.2535 - val_loss: 0.7060 - val_accuracy: 0.8191 -
val_jacard_coef: 0.2204
Epoch 129/200
2/2 [==============================] - 19s 1s/step - loss: 0.4871 - accuracy:
0.9285 - jacard_coef: 0.2498 - val_loss: 0.7204 - val_accuracy: 0.8027 -
val_jacard_coef: 0.2122
Epoch 130/200
```

```
2/2 [==============================] - 18s 1s/step - loss: 0.4787 - accuracy:
0.9299 - jacard_coef: 0.2897 - val_loss: 0.7628 - val_accuracy: 0.7518 -
val_jacard_coef: 0.1873
Epoch 131/200
2/2 [==============================] - 20s 1s/step - loss: 0.4764 - accuracy:
0.9301 - jacard_coef: 0.2582 - val_loss: 0.8025 - val_accuracy: 0.6905 -
val_jacard_coef: 0.1616
Epoch 132/200
2/2 [==============================] - 18s 1s/step - loss: 0.4702 - accuracy:
0.9308 - jacard_coef: 0.3106 - val_loss: 0.8265 - val_accuracy: 0.6418 -
val_jacard_coef: 0.1455
Epoch 133/200
2/2 [==============================] - 20s 1s/step - loss: 0.4696 - accuracy:
0.9314 - jacard_coef: 0.2810 - val_loss: 0.8396 - val_accuracy: 0.6087 -
val_jacard_coef: 0.1367
Epoch 134/200
2/2 [==============================] - 20s 1s/step - loss: 0.4716 - accuracy:
0.9307 - jacard_coef: 0.2953 - val_loss: 0.8397 - val_accuracy: 0.6133 -
val_jacard_coef: 0.1365
Epoch 135/200
2/2 [==============================] - 18s 1s/step - loss: 0.4735 - accuracy:
0.9303 - jacard_coef: 0.3317 - val_loss: 0.8270 - val_accuracy: 0.6450 -
val_jacard_coef: 0.1457
Epoch 136/200
2/2 [==============================] - 19s 1s/step - loss: 0.4634 - accuracy:
0.9321 - jacard_coef: 0.3345 - val_loss: 0.8061 - val_accuracy: 0.6799 -
val_jacard_coef: 0.1606
Epoch 137/200
2/2 [==============================] - 21s 1s/step - loss: 0.4579 - accuracy:
0.9322 - jacard_coef: 0.2807 - val_loss: 0.7923 - val_accuracy: 0.6988 -
val_jacard_coef: 0.1701
Epoch 138/200
2/2 [==============================] - 18s 980ms/step - loss: 0.4590 - accuracy:
0.9303 - jacard_coef: 0.2924 - val_loss: 0.7914 - val_accuracy: 0.7003 -
val_jacard_coef: 0.1707
Epoch 139/200
2/2 [==============================] - 18s 1s/step - loss: 0.4666 - accuracy:
0.9282 - jacard_coef: 0.3113 - val_loss: 0.7869 - val_accuracy: 0.7081 -
val_jacard_coef: 0.1738
Epoch 140/200
2/2 [==============================] - 19s 1s/step - loss: 0.4624 - accuracy:
0.9295 - jacard_coef: 0.2601 - val_loss: 0.7845 - val_accuracy: 0.7110 -
val_jacard_coef: 0.1751
Epoch 141/200
2/2 [==============================] - 18s 1s/step - loss: 0.4479 - accuracy:
0.9327 - jacard_coef: 0.3516 - val_loss: 0.7825 - val_accuracy: 0.7144 -
val_jacard_coef: 0.1755
Epoch 142/200
```

```
2/2 [==============================] - 19s 1s/step - loss: 0.4496 - accuracy:
0.9341 - jacard_coef: 0.2959 - val_loss: 0.7807 - val_accuracy: 0.7153 -
val_jacard_coef: 0.1759
Epoch 143/200
2/2 [==============================] - 20s 1s/step - loss: 0.4474 - accuracy:
0.9362 - jacard_coef: 0.3123 - val_loss: 0.7774 - val_accuracy: 0.7212 -
val_jacard_coef: 0.1772
Epoch 144/200
2/2 [==============================] - 20s 1s/step - loss: 0.4364 - accuracy:
0.9395 - jacard_coef: 0.3109 - val_loss: 0.7902 - val_accuracy: 0.7076 -
val_jacard_coef: 0.1681
Epoch 145/200
2/2 [==============================] - 20s 1s/step - loss: 0.4388 - accuracy:
0.9400 - jacard_coef: 0.2588 - val_loss: 0.8146 - val_accuracy: 0.6690 -
val_jacard_coef: 0.1522
Epoch 146/200
2/2 [==============================] - 18s 926ms/step - loss: 0.4419 - accuracy:
0.9395 - jacard_coef: 0.2640 - val_loss: 0.8198 - val_accuracy: 0.6578 -
val_jacard_coef: 0.1490
Epoch 147/200
2/2 [==============================] - 19s 1s/step - loss: 0.4413 - accuracy:
0.9403 - jacard_coef: 0.3222 - val_loss: 0.8014 - val_accuracy: 0.6903 -
val_jacard_coef: 0.1607
Epoch 148/200
2/2 [==============================] - 19s 1s/step - loss: 0.4342 - accuracy:
0.9423 - jacard_coef: 0.3349 - val_loss: 0.7953 - val_accuracy: 0.6994 -
val_jacard_coef: 0.1652
Epoch 149/200
2/2 [==============================] - 21s 1s/step - loss: 0.4376 - accuracy:
0.9408 - jacard_coef: 0.2679 - val_loss: 0.7992 - val_accuracy: 0.6932 -
val_jacard_coef: 0.1638
Epoch 150/200
2/2 [==============================] - 20s 1s/step - loss: 0.4198 - accuracy:
0.9422 - jacard_coef: 0.3333 - val_loss: 0.8054 - val_accuracy: 0.6837 -
val_jacard_coef: 0.1606
Epoch 151/200
2/2 [==============================] - 18s 988ms/step - loss: 0.4207 - accuracy:
0.9402 - jacard_coef: 0.3866 - val_loss: 0.8208 - val_accuracy: 0.6513 -
val_jacard_coef: 0.1516
Epoch 152/200
2/2 [==============================] - 17s 968ms/step - loss: 0.4372 - accuracy:
0.9340 - jacard_coef: 0.2835 - val_loss: 0.8309 - val_accuracy: 0.6186 -
val_jacard_coef: 0.1459
Epoch 153/200
2/2 [==============================] - 19s 1s/step - loss: 0.4432 - accuracy:
0.9311 - jacard_coef: 0.3651 - val_loss: 0.8418 - val_accuracy: 0.5791 -
val_jacard_coef: 0.1390
Epoch 154/200
```

```
2/2 [==============================] - 18s 1s/step - loss: 0.4262 - accuracy:
0.9349 - jacard_coef: 0.3447 - val_loss: 0.8514 - val_accuracy: 0.5454 -
val_jacard_coef: 0.1321
Epoch 155/200
2/2 [==============================] - 20s 1s/step - loss: 0.4393 - accuracy:
0.9318 - jacard_coef: 0.3144 - val_loss: 0.8575 - val_accuracy: 0.5243 -
val_jacard_coef: 0.1275
Epoch 156/200
2/2 [==============================] - 22s 1s/step - loss: 0.4405 - accuracy:
0.9339 - jacard_coef: 0.2587 - val_loss: 0.8564 - val_accuracy: 0.5324 -
val_jacard_coef: 0.1283
Epoch 157/200
2/2 [==============================] - 19s 1s/step - loss: 0.4698 - accuracy:
0.9303 - jacard_coef: 0.2960 - val_loss: 0.8452 - val_accuracy: 0.5743 -
val_jacard_coef: 0.1363
Epoch 158/200
2/2 [==============================] - 19s 1s/step - loss: 0.4810 - accuracy:
0.9283 - jacard_coef: 0.3285 - val_loss: 0.8357 - val_accuracy: 0.6003 -
val_jacard_coef: 0.1432
Epoch 159/200
2/2 [==============================] - 20s 1s/step - loss: 0.4758 - accuracy:
0.9292 - jacard_coef: 0.3473 - val_loss: 0.8340 - val_accuracy: 0.6006 -
val_jacard_coef: 0.1446
Epoch 160/200
2/2 [==============================] - 19s 1s/step - loss: 0.4759 - accuracy:
0.9289 - jacard_coef: 0.2616 - val_loss: 0.8261 - val_accuracy: 0.6204 -
val_jacard_coef: 0.1503
Epoch 161/200
2/2 [==============================] - 19s 1s/step - loss: 0.4791 - accuracy:
0.9281 - jacard_coef: 0.3397 - val_loss: 0.8151 - val_accuracy: 0.6491 -
val_jacard_coef: 0.1584
Epoch 162/200
2/2 [==============================] - 19s 1s/step - loss: 0.4825 - accuracy:
0.9267 - jacard_coef: 0.2899 - val_loss: 0.8003 - val_accuracy: 0.6822 -
val_jacard_coef: 0.1689
Epoch 163/200
2/2 [==============================] - 19s 1s/step - loss: 0.4809 - accuracy:
0.9266 - jacard_coef: 0.3010 - val_loss: 0.7813 - val_accuracy: 0.7158 -
val_jacard_coef: 0.1817
Epoch 164/200
2/2 [==============================] - 20s 1s/step - loss: 0.4744 - accuracy:
0.9284 - jacard_coef: 0.2679 - val_loss: 0.7637 - val_accuracy: 0.7416 -
val_jacard_coef: 0.1925
Epoch 165/200
2/2 [==============================] - 20s 1s/step - loss: 0.4700 - accuracy:
0.9295 - jacard_coef: 0.3749 - val_loss: 0.7518 - val_accuracy: 0.7554 -
val_jacard_coef: 0.1991
Epoch 166/200
```

```
2/2 [==============================] - 20s 1s/step - loss: 0.4607 - accuracy:
0.9315 - jacard_coef: 0.3107 - val_loss: 0.7566 - val_accuracy: 0.7470 -
val_jacard_coef: 0.1958
Epoch 167/200
2/2 [==============================] - 27s 1s/step - loss: 0.4561 - accuracy:
0.9318 - jacard_coef: 0.2915 - val_loss: 0.7752 - val_accuracy: 0.7195 -
val_jacard_coef: 0.1839
Epoch 168/200
2/2 [==============================] - 24s 1s/step - loss: 0.4481 - accuracy:
0.9328 - jacard_coef: 0.3456 - val_loss: 0.7887 - val_accuracy: 0.6959 -
val_jacard_coef: 0.1749
Epoch 169/200
2/2 [==============================] - 23s 1s/step - loss: 0.4518 - accuracy:
0.9332 - jacard_coef: 0.2015 - val_loss: 0.7854 - val_accuracy: 0.7007 -
val_jacard_coef: 0.1771
Epoch 170/200
2/2 [==============================] - 22s 1s/step - loss: 0.4477 - accuracy:
0.9340 - jacard_coef: 0.3792 - val_loss: 0.7727 - val_accuracy: 0.7196 -
val_jacard_coef: 0.1850
Epoch 171/200
2/2 [==============================] - 24s 1s/step - loss: 0.4314 - accuracy:
0.9373 - jacard_coef: 0.2640 - val_loss: 0.7562 - val_accuracy: 0.7487 -
val_jacard_coef: 0.1956
Epoch 172/200
2/2 [==============================] - 25s 2s/step - loss: 0.4094 - accuracy:
0.9406 - jacard_coef: 0.3404 - val_loss: 0.7438 - val_accuracy: 0.7785 -
val_jacard_coef: 0.2028
Epoch 173/200
2/2 [==============================] - 23s 1s/step - loss: 0.4047 - accuracy:
0.9406 - jacard_coef: 0.3186 - val_loss: 0.7753 - val_accuracy: 0.7665 -
val_jacard_coef: 0.1801
Epoch 174/200
2/2 [==============================] - 24s 1s/step - loss: 0.4232 - accuracy:
0.9363 - jacard_coef: 0.3621 - val_loss: 0.7988 - val_accuracy: 0.7215 -
val_jacard_coef: 0.1659
Epoch 175/200
2/2 [==============================] - 21s 2s/step - loss: 0.4590 - accuracy:
0.9286 - jacard_coef: 0.3614 - val_loss: 0.8079 - val_accuracy: 0.6736 -
val_jacard_coef: 0.1625
Epoch 176/200
2/2 [==============================] - 21s 903ms/step - loss: 0.4714 - accuracy:
0.9248 - jacard_coef: 0.2867 - val_loss: 0.8075 - val_accuracy: 0.6670 -
val_jacard_coef: 0.1639
Epoch 177/200
2/2 [==============================] - 18s 1s/step - loss: 0.4492 - accuracy:
0.9284 - jacard_coef: 0.2619 - val_loss: 0.7883 - val_accuracy: 0.7027 -
val_jacard_coef: 0.1776
Epoch 178/200
```

```
2/2 [==============================] - 19s 1s/step - loss: 0.4283 - accuracy:
0.9321 - jacard_coef: 0.3782 - val_loss: 0.7779 - val_accuracy: 0.7203 -
val_jacard_coef: 0.1843
Epoch 179/200
2/2 [==============================] - 20s 2s/step - loss: 0.4319 - accuracy:
0.9315 - jacard_coef: 0.3855 - val_loss: 0.7976 - val_accuracy: 0.6880 -
val_jacard_coef: 0.1706
Epoch 180/200
2/2 [==============================] - 23s 1s/step - loss: 0.4308 - accuracy:
0.9315 - jacard_coef: 0.3917 - val_loss: 0.8187 - val_accuracy: 0.6426 -
val_jacard_coef: 0.1562
Epoch 181/200
2/2 [==============================] - 21s 1s/step - loss: 0.4339 - accuracy:
0.9308 - jacard_coef: 0.3050 - val_loss: 0.8335 - val_accuracy: 0.6003 -
val_jacard_coef: 0.1456
Epoch 182/200
2/2 [==============================] - 19s 1s/step - loss: 0.4135 - accuracy:
0.9356 - jacard_coef: 0.2778 - val_loss: 0.8390 - val_accuracy: 0.5839 -
val_jacard_coef: 0.1415
Epoch 183/200
2/2 [==============================] - 18s 913ms/step - loss: 0.4083 - accuracy:
0.9373 - jacard_coef: 0.2421 - val_loss: 0.8318 - val_accuracy: 0.6055 -
val_jacard_coef: 0.1468
Epoch 184/200
2/2 [==============================] - 24s 2s/step - loss: 0.4254 - accuracy:
0.9354 - jacard_coef: 0.2126 - val_loss: 0.8211 - val_accuracy: 0.6345 -
val_jacard_coef: 0.1550
Epoch 185/200
2/2 [==============================] - 20s 1s/step - loss: 0.4309 - accuracy:
0.9348 - jacard_coef: 0.3301 - val_loss: 0.8084 - val_accuracy: 0.6621 -
val_jacard_coef: 0.1644
Epoch 186/200
2/2 [==============================] - 23s 1s/step - loss: 0.4208 - accuracy:
0.9381 - jacard_coef: 0.4091 - val_loss: 0.7980 - val_accuracy: 0.6813 -
val_jacard_coef: 0.1719
Epoch 187/200
2/2 [==============================] - 25s 1s/step - loss: 0.4036 - accuracy:
0.9406 - jacard_coef: 0.3890 - val_loss: 0.7985 - val_accuracy: 0.6783 -
val_jacard_coef: 0.1720
Epoch 188/200
2/2 [==============================] - 19s 935ms/step - loss: 0.3974 - accuracy:
0.9411 - jacard_coef: 0.3946 - val_loss: 0.8115 - val_accuracy: 0.6504 -
val_jacard_coef: 0.1631
Epoch 189/200
2/2 [==============================] - 18s 1s/step - loss: 0.3855 - accuracy:
0.9428 - jacard_coef: 0.3511 - val_loss: 0.8294 - val_accuracy: 0.6059 -
val_jacard_coef: 0.1500
Epoch 190/200
```

```
2/2 [==============================] - 19s 1s/step - loss: 0.3791 - accuracy:
0.9441 - jacard_coef: 0.2785 - val_loss: 0.8332 - val_accuracy: 0.5952 -
val_jacard_coef: 0.1469
Epoch 191/200
2/2 [==============================] - 18s 1s/step - loss: 0.3838 - accuracy:
0.9439 - jacard_coef: 0.3350 - val_loss: 0.8153 - val_accuracy: 0.6416 -
val_jacard_coef: 0.1594
Epoch 192/200
2/2 [==============================] - 19s 1s/step - loss: 0.3864 - accuracy:
0.9451 - jacard_coef: 0.3373 - val_loss: 0.7862 - val_accuracy: 0.7030 -
val_jacard_coef: 0.1795
Epoch 193/200
2/2 [==============================] - 18s 1s/step - loss: 0.3972 - accuracy:
0.9447 - jacard_coef: 0.2994 - val_loss: 0.7568 - val_accuracy: 0.7540 -
val_jacard_coef: 0.1981
Epoch 194/200
2/2 [==============================] - 20s 1s/step - loss: 0.4098 - accuracy:
0.9436 - jacard_coef: 0.3280 - val_loss: 0.7657 - val_accuracy: 0.7397 -
val_jacard_coef: 0.1923
Epoch 195/200
2/2 [==============================] - 19s 1s/step - loss: 0.4085 - accuracy:
0.9446 - jacard_coef: 0.2895 - val_loss: 0.8043 - val_accuracy: 0.6703 -
val_jacard_coef: 0.1670
Epoch 196/200
2/2 [==============================] - 19s 945ms/step - loss: 0.3946 - accuracy:
0.9466 - jacard_coef: 0.3836 - val_loss: 0.8300 - val_accuracy: 0.6075 -
val_jacard_coef: 0.1488
Epoch 197/200
2/2 [==============================] - 20s 1s/step - loss: 0.3964 - accuracy:
0.9453 - jacard_coef: 0.3147 - val_loss: 0.8444 - val_accuracy: 0.5654 -
val_jacard_coef: 0.1386
Epoch 198/200
2/2 [==============================] - 21s 1s/step - loss: 0.3897 - accuracy:
0.9454 - jacard_coef: 0.3207 - val_loss: 0.8380 - val_accuracy: 0.5854 -
val_jacard_coef: 0.1434
Epoch 199/200
2/2 [==============================] - 20s 2s/step - loss: 0.3705 - accuracy:
0.9481 - jacard_coef: 0.4450 - val_loss: 0.8121 - val_accuracy: 0.6519 -
val_jacard_coef: 0.1619
Epoch 200/200
2/2 [==============================] - 21s 1s/step - loss: 0.3726 - accuracy:
0.9469 - jacard_coef: 0.3586 - val_loss: 0.7742 - val_accuracy: 0.7246 -
val_jacard_coef: 0.1880
```

[102]:
```
df_result = pd.DataFrame(history.history)
df_result
```

[102]:

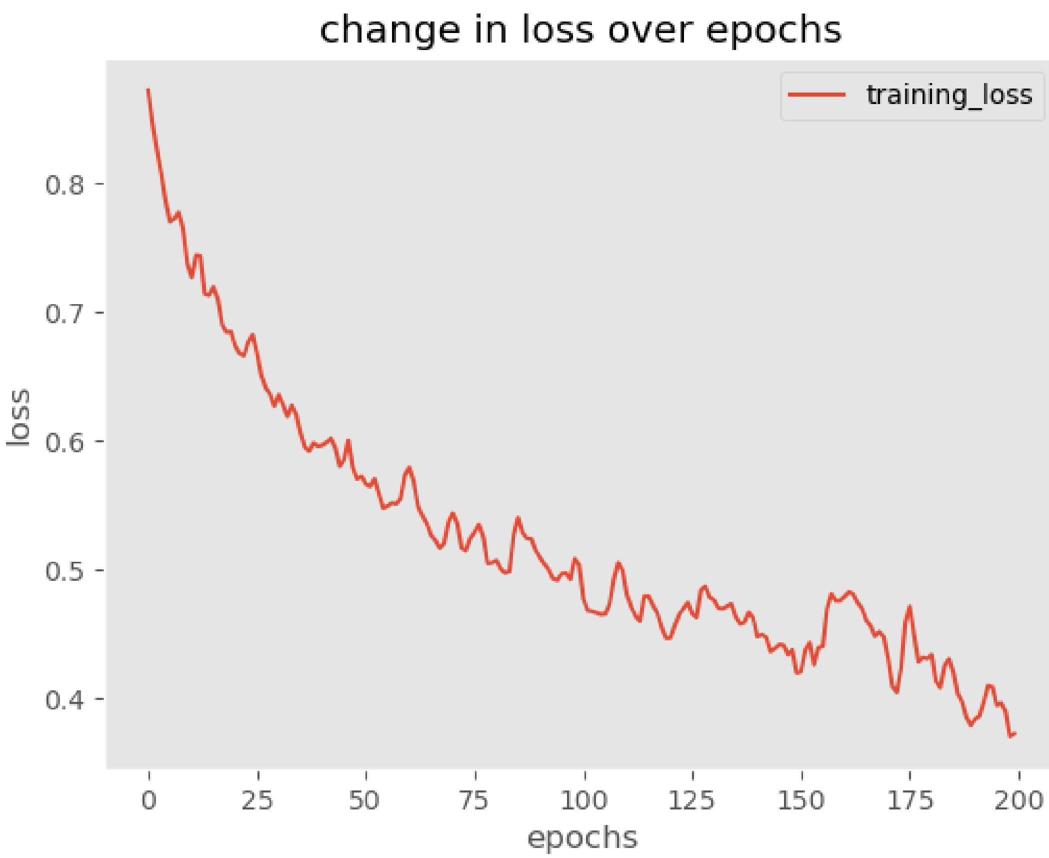
```
          loss  accuracy  jacard_coef  val_loss  val_accuracy  val_jacard_coef
0     0.872040  0.356049     0.093717  0.878686      0.129052         0.071997
1     0.846515  0.416591     0.083771  0.900601      0.078904         0.078783
2     0.826054  0.497650     0.078493  0.914008      0.074944         0.077734
3     0.807635  0.595213     0.101147  0.909849      0.075099         0.076188
4     0.786400  0.692813     0.139161  0.915594      0.075293         0.076855
..         …         …          …         …             …               …
195   0.394641  0.946577     0.383552  0.830006      0.607496         0.148772
196   0.396441  0.945278     0.314745  0.844410      0.565446         0.138572
197   0.389689  0.945366     0.320742  0.838025      0.585351         0.143395
198   0.370472  0.948130     0.445005  0.812102      0.651905         0.161890
199   0.372572  0.946875     0.358567  0.774213      0.724639         0.188048

[200 rows x 6 columns]
```

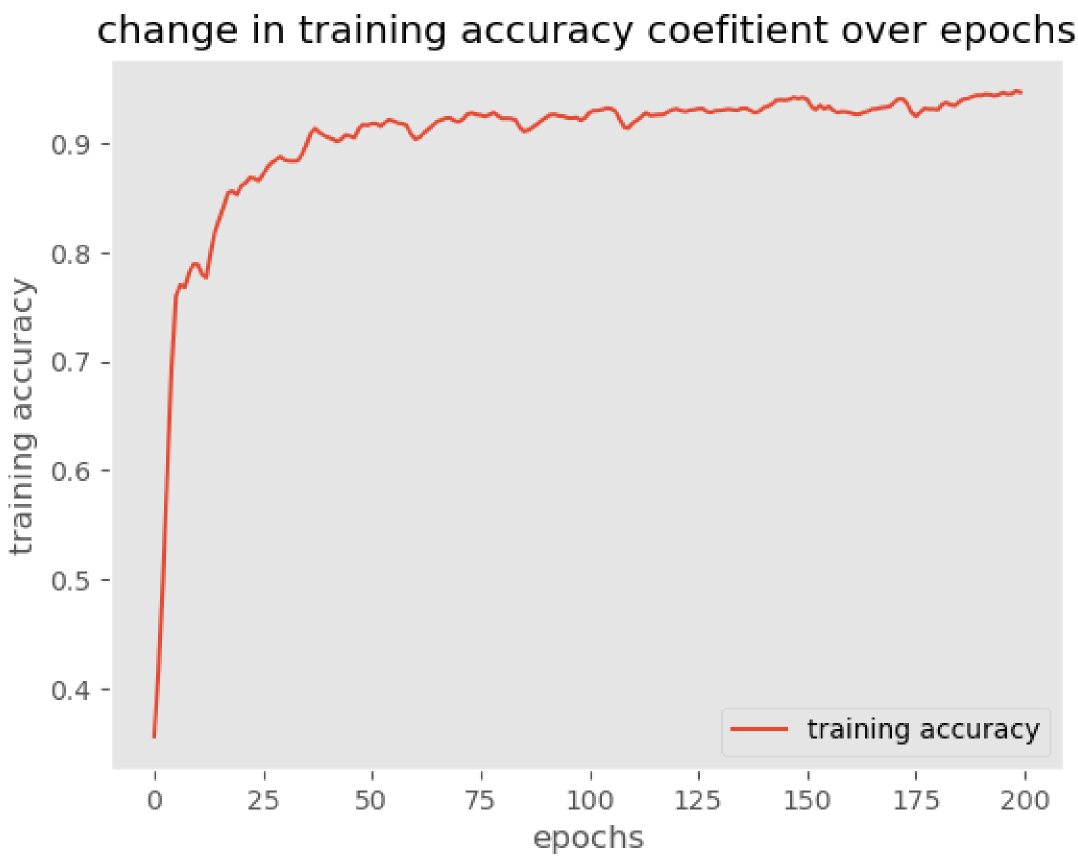
5 Visualize the model predictions

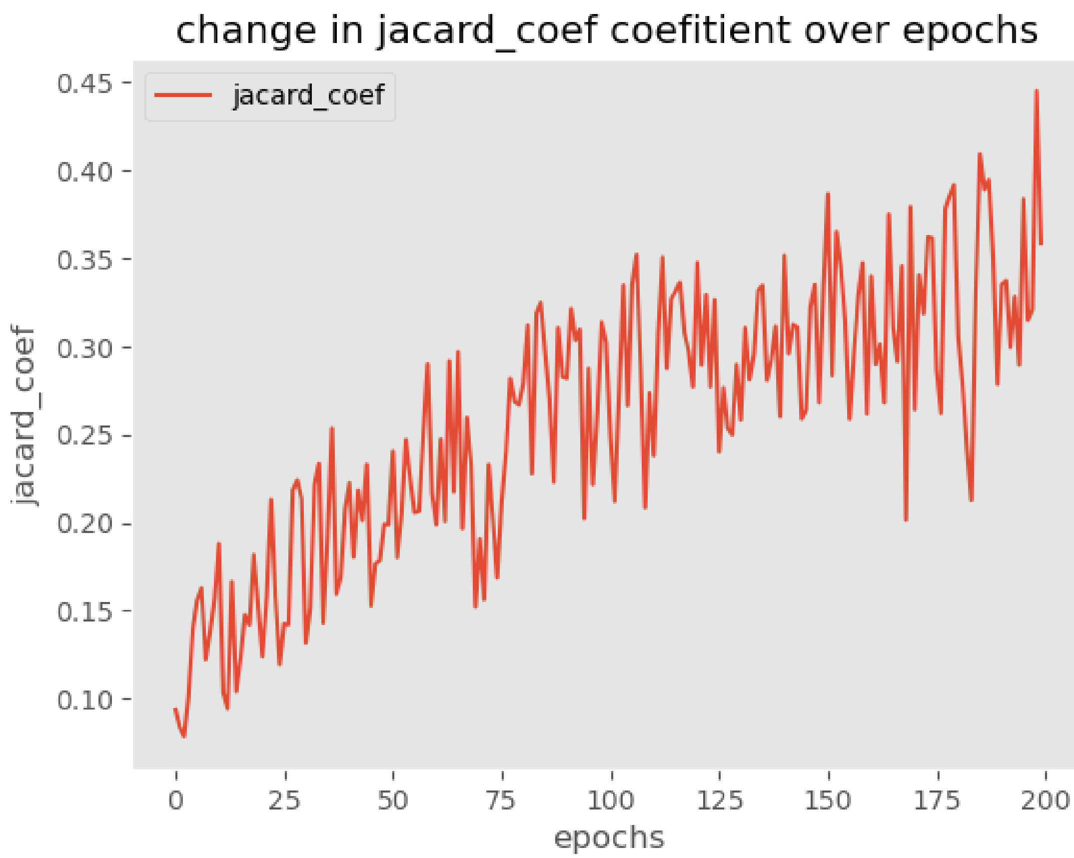
[103]:

```python
# Plotting loss change over epochs---------------
nrange=nepochs
x = [i for i in range(nrange)]
plt.plot(x,history.history['loss'])
plt.title('change in loss over epochs')
plt.legend(['training_loss'])
plt.xlabel('epochs')
plt.ylabel('loss')
#plt.axis('off')
plt.grid(None)
plt.show()
plt.tight_layout()

# Plotting accuracy change over epochs---------------------
x = [i for i in range(nrange)]
plt.plot(x,history.history['accuracy'])
plt.title('change in training accuracy coefitient over epochs')
plt.legend(['training accuracy'])
plt.xlabel('epochs')
plt.ylabel('training accuracy')
plt.grid(None)
plt.show()
plt.tight_layout()

# Plotting accuracy change over epochs---------------------
x = [i for i in range(nrange)]
plt.plot(x,history.history['jacard_coef'])
plt.title('change in jacard_coef coefitient over epochs')
plt.legend(['jacard_coef'])
```

```
plt.xlabel('epochs')
plt.ylabel('jacard_coef')
plt.grid(None)
plt.show()
plt.tight_layout()
```




change in training accuracy coefitient over epochs
training accuracy
0.4
0.5
0.6
0.7
0.8
0.9
0
25
50
75
100
125
150
175
200
epochs
training accuracy




```
<Figure size 640x480 with 0 Axes>
```

[104]:

```
# Creating predictions on our test set------------------
predictions = model.predict(X_test)

# create predictes mask---------------

def create_mask(predictions,input_shape=(W,H,1)):
    mask = np.zeros(input_shape)
    mask[predictions>0.5] = 1
    return mask
```

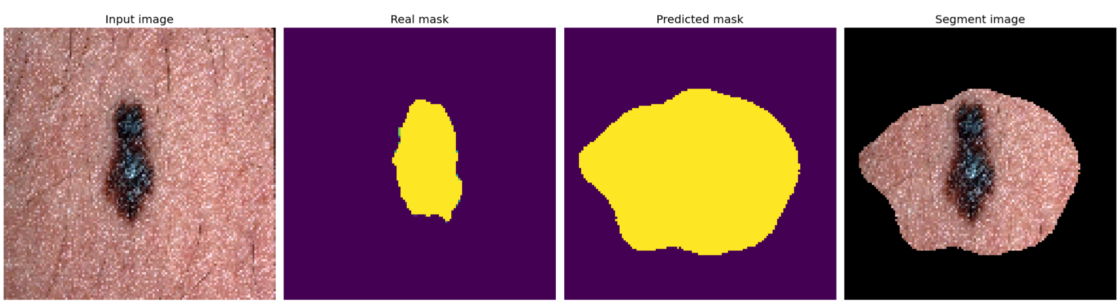
```
2/2 [==============================] - 2s 233ms/step
```

[105]:

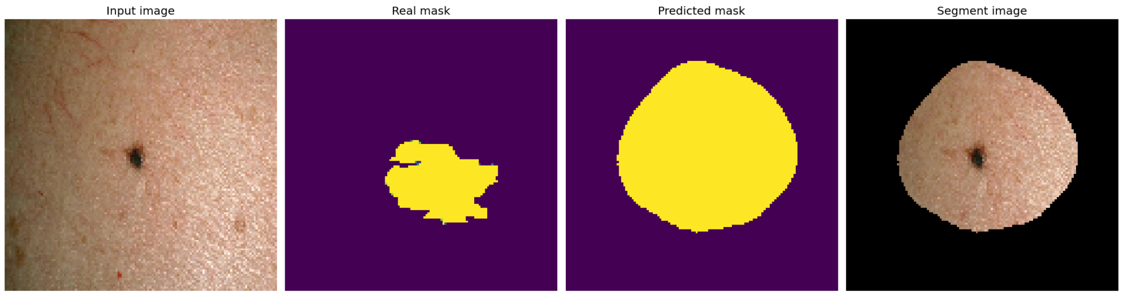
```
# Ploting results for one image

def plot_results_for_one_sample(sample_index):

    mask = create_mask(predictions[sample_index])
    fig = plt.figure(figsize=(20,20))
```

```
    #image
    fig.add_subplot(1,4,1)
    plt.title('Input image')
    plt.imshow(X_test[sample_index])
    plt.axis('off')
    plt.grid(None)
    #mask
    fig.add_subplot(1,4,2)
    plt.title('Real mask')
    plt.imshow(Y_test[sample_index])
    plt.axis('off')
    plt.grid(None)
    #Predicted mask
    fig.add_subplot(1,4,3)
    plt.title('Predicted mask')
    plt.imshow(mask)
    plt.axis('off')
    plt.grid(None)
    #Segment
    fig.add_subplot(1,4,4)
    plt.title("Segment image")
    plt.imshow(X_test[sample_index]*mask)
    plt.grid(None)
    plt.axis('off')
    fig.tight_layout()
```

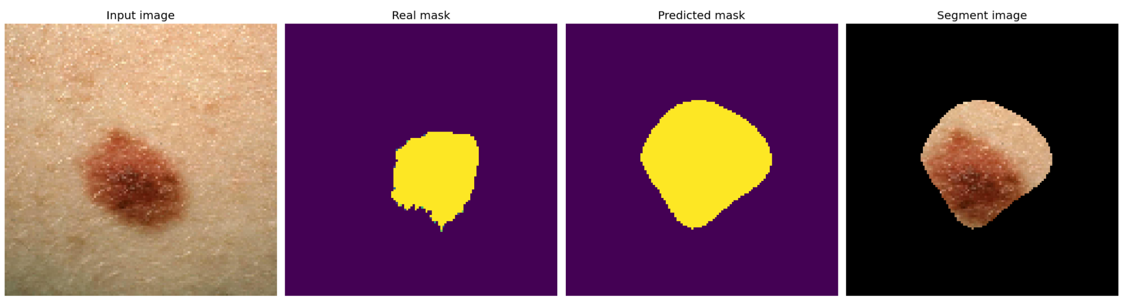
[106]:
```
#Show predicted result---------------
plot_results_for_one_sample(0)
```



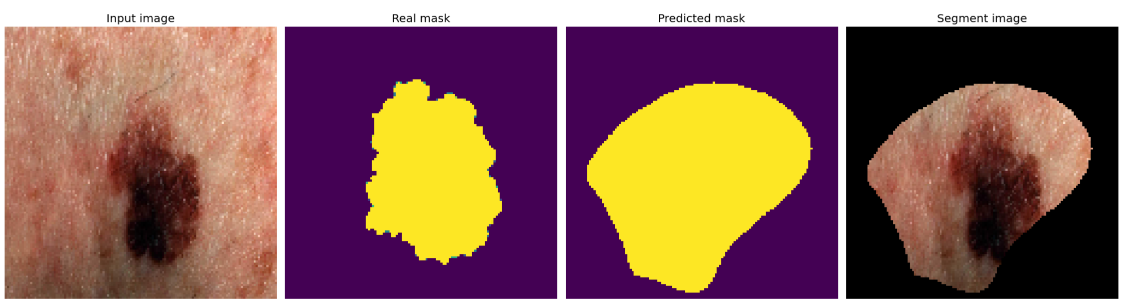
[107]:
```
#Show predicted result---------------
plot_results_for_one_sample(1)
```



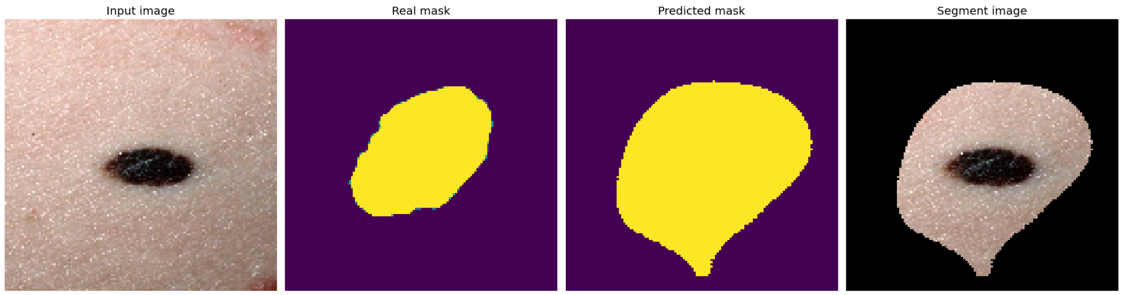
[108]:
```
#Show predicted result---------------
plot_results_for_one_sample(2)
```



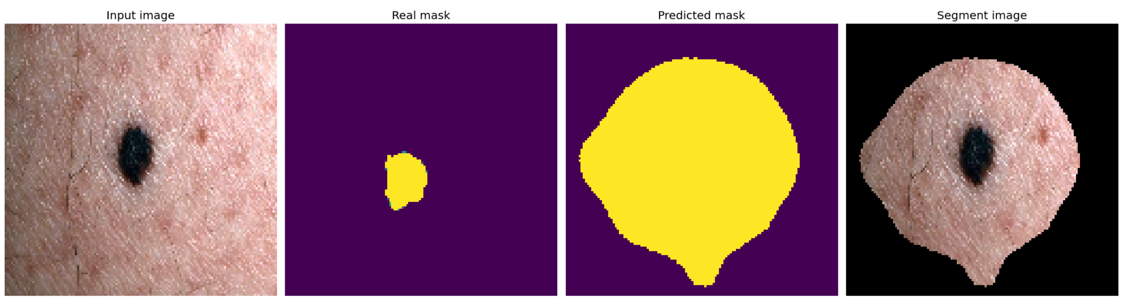
[109]:
```
#Show predicted result---------------
plot_results_for_one_sample(3)
```



[110]:
```
#Show predicted result---------------
plot_results_for_one_sample(4)
```



[111]:
```
#Show predicted result---------------
plot_results_for_one_sample(5)
```



[ ]:
```
```

[ ]:
```
```