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

path = "/content/drive/My Drive/暑期科研/"

os.chdir(path)
os.listdir(path)
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/c
['chromosome_r_y_train.npy',
 'chromosome_l_y_test.npy',
 'chromosome_r_y_test.npy',
 'chromosome_l_y_train.npy',
 'chromosome_l_x_test.npy',
 'chromosome_l_x_train.npy',
 'chromosome_r_x_test.npy',
 'chromosome_r_x_train.npy']
```

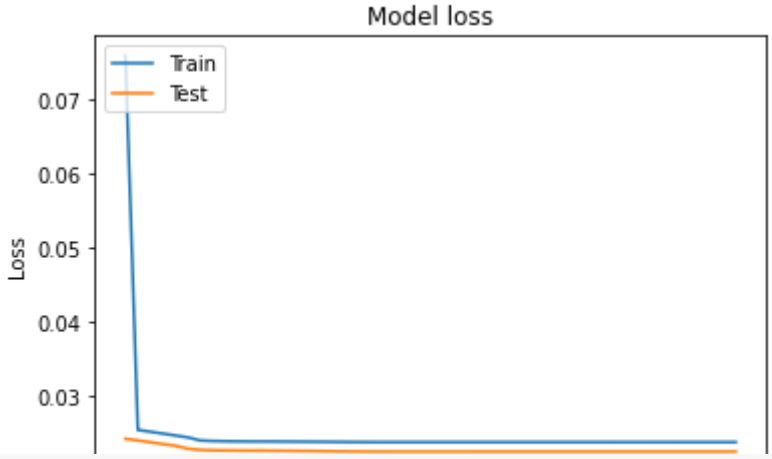
```
import glob

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt



from keras.preprocessing import image
from keras.models import Model
from keras.optimizers import Adam
from keras.callbacks import EarlyStopping
from keras.layers import Input, Dense, Activation, BatchNormalization, Flatten, Conv2D
from keras.layers import MaxPooling2D, Dropout, UpSampling2D
import os

x_train_savepath = './chromosome_r_x_train.npy'
y_train_savepath = './chromosome_r_y_train.npy'

x_test_savepath = './chromosome_r_x_test.npy'
y_test_savepath = './chromosome_r_y_test.npy'
print('-------------Load Datasets-----------------')
x_train_save = np.load(x_train_savepath)
y_train = np.load(y_train_savepath)
x_test_save = np.load(x_test_savepath)
y_test = np.load(y_test_savepath)
x_train = np.reshape(x_train_save, (len(x_train_save), 150, 150,1))
x_test = np.reshape(x_test_save, (len(x_test_save), 150, 150,1))

x_train = x_train.astype('float32') / 255.
x_test = x_test.astype('float32') / 255.
# x_train = x_train.reshape((len(x_train), np.prod(x_train.shape[1:])))
# x_test = x_test.reshape((len(x_test), np.prod(x_test.shape[1:])))
print(x_train.shape)
print(x_test.shape)
#
```

```
class  Autoencoder():
        def  __init__(self):

                self.img_shape  =  (150,150,1)

                optimizer  =  Adam(lr=0.001)

                self.autoencoder_model  =  self.build_model()
                self.autoencoder_model.compile(loss='binary_crossentropy',  optimizer=optimizer)
                self.autoencoder_model.summary()

        def  build_model(self):
                input_layer  =  Input(shape=self.img_shape)

                #  encoder
                h  =  Conv2D(64,  (3,  3),  activation='relu',  padding='same')(input_layer)
                h  =  MaxPooling2D((3,  3),  padding='same')(h)

                #  decoder
                h  =  Conv2D(64,  (3,  3),  activation='relu',  padding='same')(h)
                h  =  UpSampling2D((3,  3))(h)
                output_layer  =  Conv2D(1,  (3,  3),  activation='sigmoid',  padding='same')(h)

                return  Model(input_layer,  output_layer)

        def  train_model(self,  x_train,  y_train,  x_test,  y_test,  epochs,  batch_size):
                early_stopping  =  EarlyStopping(monitor='val_loss',
                                                                                        min_delta=0,
                                                                                        patience=5,
                                                                                        verbose=1,
                                                                                        mode='auto')
                history  =  self.autoencoder_model.fit(x_train,  x_train,
                                                                                                batch_siz
                                                                                                epochs=ep
                                                                                                validatio
                                                                                                callbacks
                plt.plot(history.history['loss'])
                plt.plot(history.history['val_loss'])
                plt.title('Model  loss')
                plt.ylabel('Loss')
                plt.xlabel('Epoch')
                plt.legend(['Train',  'Test'],  loc='upper  left')
                plt.show()

        def  eval_model(self,  x_test):
                preds  =  self.autoencoder_model.predict(x_test)
                return  preds

ae  =  Autoencoder()
ae.train_model(x_train,  y_train,  x_test,  y_test,  epochs=50,  batch_size=4)
```

```
-------------Load Datasets-----------------
(988, 150, 150, 1)
(188, 150, 150, 1)
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 150, 150, 1)       0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 150, 150, 64)      640
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 50, 50, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 50, 50, 64)        36928
_________________________________________________________________
up_sampling2d_1 (UpSampling2 (None, 150, 150, 64)      0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 150, 150, 1)       577
=================================================================
Total params: 38,145
Trainable params: 38,145
Non-trainable params: 0
_________________________________________________________________
Train on 988 samples, validate on 188 samples
Epoch 1/50
988/988 [==============================] - 3s 3ms/step - loss: 0.0759 - val_loss: 0.0242
Epoch 2/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0254 - val_loss: 0.0240
Epoch 3/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0252 - val_loss: 0.0237
Epoch 4/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0249 - val_loss: 0.0235
Epoch 5/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0247 - val_loss: 0.0233
Epoch 6/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0244 - val_loss: 0.0229
Epoch 7/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0240 - val_loss: 0.0227
Epoch 8/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0239 - val_loss: 0.0227
Epoch 9/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0239 - val_loss: 0.0226
Epoch 10/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0239 - val_loss: 0.0226
Epoch 11/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0239 - val_loss: 0.0226
Epoch 12/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0226
Epoch 13/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0226
Epoch 14/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0226
Epoch 15/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0226
Epoch 16/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 17/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 18/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 19/50
```

988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 20/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 21/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 22/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 23/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 24/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 25/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 26/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 27/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 28/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 29/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 30/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 31/50
988/988 [==============================] - 2s 2ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 32/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 33/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 34/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 35/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 36/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 37/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 38/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 39/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 40/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 41/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 42/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 43/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 44/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 45/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 46/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 47/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 48/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 49/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
Epoch 50/50

```
Epoch 50/50
988/988 [==============================] - 1s 1ms/step - loss: 0.0238 - val_loss: 0.0225
```




```
decoded_test = ae.eval_model(x_test)

import matplotlib.pyplot as plt
%matplotlib inline
shape = (150, 150)
fig, axes = plt.subplots(2,10,
                                                figsize=(10, 2),
                                                subplot_kw={
                                                        'xticks': [],
                                                        'yticks': []
                                                },
                                                gridspec_kw=dict(hspace=0.1, wspace=0.1))
for i in range(10):
        axes[0][i].imshow(np.reshape(x_test[i], shape))
        axes[1][i].imshow(np.reshape(decoded_test[i], shape))
plt.show()
```