import numpy as np

from keras.preprocessing.image import img\_to\_array

from keras.layers import Dense, Conv2D, MaxPooling2D, UpSampling2D

from keras.models import Sequential

import cv2

import matplotlib.pyplot as plt

img\_size=256

img\_data=[]

img=cv2.imread('01.jpg', 1)

rgb\_img=cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

rgb\_img=cv2.resize(rgb\_img, (256,256))

img\_data.append(img\_to\_array(rgb\_img))

img\_final=np.reshape(img\_data, (len(img\_data),256, 256, 3))

img\_final=img\_final.astype('float32')/255

model=Sequential()

model.add(Conv2D(64, (3,3), activation='relu', padding='same', input\_shape=(256,256,3)))

model.add(MaxPooling2D((2,2), padding='same'))

model.add(Conv2D(32, (3,3),activation='relu',padding='same'))

model.add(MaxPooling2D((2,2), padding='same'))

model.add(Conv2D(16, (3,3),activation='relu',padding='same'))

model.add(MaxPooling2D((2,2), padding='same'))

model.add(Conv2D(16, (3,3), activation='relu', padding='same'))

model.add(UpSampling2D((2,2)))

model.add(Conv2D(32, (3,3), activation='relu', padding='same'))

model.add(UpSampling2D((2,2)))

model.add(Conv2D(64, (3,3), activation='relu', padding='same'))

model.add(UpSampling2D((2,2)))

model.add(Conv2D(3, (3,3), activation='relu', padding='same'))

model.compile(optimizer='adam',loss='mean\_squared\_error',metrics=['accuracy'])

model.summary()

model.fit(img\_final, img\_final, epochs=2000, shuffle=True)

pred=model.predict(img\_final)

plt.imshow(pred[0].reshape(256,256,3))



import numpy as np

import matplotlib.pyplot as plt

import numpy as np

from keras.models import load\_model

from keras.models import Sequential, Model

from keras.layers import Dense, Activation

from keras.layers import Conv2D, MaxPooling2D, Input, ZeroPadding2D, merge

from keras.preprocessing.image import img\_to\_array

from keras.layers import Dense, Conv2D, MaxPooling2D, UpSampling2D

from keras.models import Sequential

import cv2

img\_size=256

img\_data=[]

img=cv2.imread('21.jpeg', 1)

rgb\_img=cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

rgb\_img=cv2.resize(rgb\_img, (256,256))

img\_data.append(img\_to\_array(rgb\_img))

img\_final=np.reshape(img\_data, (len(img\_data),256, 256, 3))

img\_final=img\_final.astype('float32')/255

img\_shape = (32, 32, 1)

model = Sequential()

model = Conv2D(56, (5, 5), padding='same', kernel\_initializer='he\_normal')(input\_img)

model = PReLU()(model)

model = Conv2D(16, (1, 1), padding='same', kernel\_initializer='he\_normal')(input\_img)

model = PReLU()(model)

model = Conv2D(12, (3, 3), padding='same', kernel\_initializer='he\_normal')(input\_img)

model = PReLU()(model)

model = Conv2D(12, (3, 3), padding='same', kernel\_initializer='he\_normal')(input\_img)

model = PReLU()(model)

model = Conv2D(12, (3, 3), padding='same', kernel\_initializer='he\_normal')(input\_img)

model = PReLU()(model)

model = Conv2D(12, (3, 3), padding='same', kernel\_initializer='he\_normal')(input\_img)

model = PReLU()(model)

model = Conv2D(56, (1, 1), padding='same', kernel\_initializer='he\_normal')(input\_img)

model = PReLU()(model)

model = Conv2DTranspose(1, (9, 9), strides=(3, 3), padding='same')(model)

output\_img = model

model = Model(input\_img, output\_img)

model.compile(optimizer='adam',loss='mean\_squared\_error',metrics=['accuracy'])

model.summary()

model.fit(img\_final, img\_final, epochs=2000, shuffle=True)

pred=model.predict(img\_final)

plt.imshow(pred[0].reshape(256,256,3))

 