```
# To find what a model holds in different channels.
# 13.1.2022
# Sangeeta Biswas

from tensorflow.keras.applications.vgg16 import VGG16,
preprocess_input
import cv2
import matplotlib.pyplot as plt
import numpy as np
from tensorflow.keras.models import Model

DIR = '/home/bibrity/DeepLearning/'

def main():
        # Load a pre-trained model.
        baseModel = VGG16()
        baseModel.summary()

        # Prepare a new model having the desired layer as the output
layer.
        inputs = baseModel.input
        outputs = baseModel.layers[5].output
        model = Model(inputs, outputs)

        # Prepare data.
        img = prepare_data()

        # Predict output of a specific layer.
        outputs = model.predict(img)

        # Display what different channls see.
        display_channels(outputs)

def display_channels(chSet):
        plt.figure(figsize = (20, 20))
        for i in range(25):
                plt.subplot(5, 5, i + 1)
                plt.imshow(chSet[0, :, :, i], cmap = 'gray')
                plt.axis('off')
        plt.show()
        plt.close()

def prepare_data():
        # Load an image
        imgPath = DIR + 'Elephant.jpg' #'Baby.jpeg' #'Rose.jpeg'
#'Boat.jpeg' #
        bgrImg = cv2.imread(imgPath)
        print(bgrImg.shape)

        # Convert the image from BGR into RGB format
        rgbImg = cv2.cvtColor(bgrImg, cv2.COLOR_BGR2RGB)

        # Reshape the image so that it can fit into the model.
        #display_img(rgbImg)
        rgbImg = cv2.resize(rgbImg, (224, 224))
        display_img(rgbImg)
        # Expand dimension since the model accepts 4D data.
        print(rgbImg.shape)
        rgbImg = np.expand_dims(rgbImg, axis = 0)
        print(rgbImg.shape)

        # Preprocess image
        rgbImg = preprocess_input(rgbImg)

        return rgbImg

def display_img(img):
        plt.imshow(img)
        plt.show()
        plt.close()

if __name__ == '__main__':
        main()
```