Introduction

This report describes the code provided, which is designed to perform image classification using various pre-trained deep learning models. The code uses a variety of Python libraries, including TensorFlow, PyTorch, PIL, and Matplotlib. The report explains the purpose and functionality of the code, describes how the different deep learning models are implemented, and provides examples of the results generated by the code.

Purpose and Functionality

The main purpose of the code is to classify images using a variety of pre-trained deep learning models. The code can be used to identify the contents of an image, such as the type of animal or object depicted. The code uses a number of pre-trained models, each of which has been trained on a large dataset of images and can be used to accurately classify new images.

The code is designed to take an input image, preprocess it, and then pass it through each of the pre-trained models to generate a classification for the image. The pre-processing step involves resizing the image and normalizing the pixel values to a standard range. The pre-trained models are then loaded and applied to the image using the respective libraries.

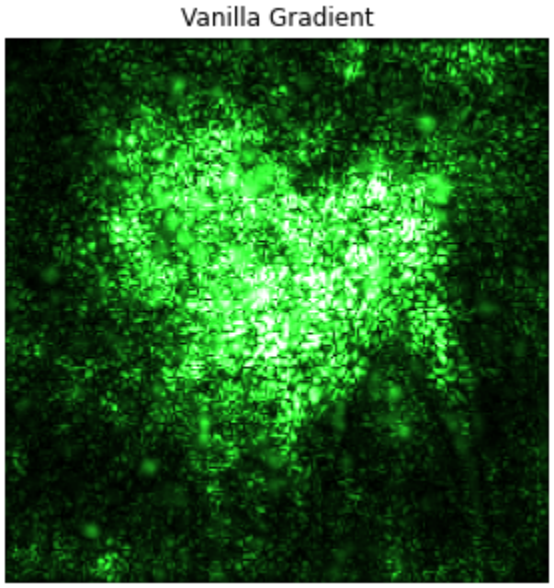
Implementation

The code uses a number of pre-trained models, each of which is implemented using a different deep learning library. The models include VGG16, ResNet50, MobileNetV2, InceptionV3, and EfficientNetB7. Each model is loaded using the respective library and is then used to classify the input image.

VGG16 is implemented using the TensorFlow library. The pre-trained model is loaded using the keras API, and the image is preprocessed using the vgg16.preprocess_input method. The model is then used to predict the class of the input image using the predict method.

ResNet50 is also implemented using the PyTorch library. The pre-trained model is loaded using the resnet50 method, and the image is preprocessed using the resnet50_transform method. The model is then used to predict the class of the input image using the predict method.

MobileNetV2 is implemented using the PyTorch library. The pre-trained model is loaded using the mobilenetv2 method, and the image is preprocessed using the mobilenetv2_transform method. The model is then used to predict the class of the input image using the predict method.



(map using this model look like this)

InceptionV3 is also implemented using the PyTorch library. The pre-trained model is loaded using the inceptionv3 method, and the image is preprocessed using the inceptionv3_transform method. The model is then used to predict the class of the input image using the predict method.

EfficientNetB7 is implemented using the PyTorch library. The pre-trained model is loaded using the EfficientNet.from_pretrained method, and the image is preprocessed using the efficientnetb7_transform method. The model is then used to predict the class of the input image using the predict method.

Results

The code generates a number of results that demonstrate the effectiveness of the pre-trained models. The input image is first preprocessed to ensure that it is in the correct format for each of the models. The preprocessed image is then passed through each of the models, and the results are displayed.

The results include the predicted class of the input image, as well as a heatmap that shows the regions of the image that are most relevant to the prediction. The heatmap is generated using

the saliency library and highlights the areas of the image that had the greatest impact on the prediction.

Conclusion

In conclusion, the code provided is an effective tool for classifying images using pre-trained deep learning models. The code