Image classification in corrupted images

Description:

You are provided with a dataset of cats and dogs and your task is to design and implement a CNN architecture for a classification task. Do the following steps (All need implementation):

1. Implement a CNN-based architecture that archives 80% of accuracy. Be careful that 80% is enough and higher accuracies are desirable but not necessary. Your **way of approaching the architecture is important**.
2. Produce an adversarial example.
3. Now assume that the images could be noisy, for example, additive Gaussian noise. Add noise to drop the test performance to under 80%. Now, propose a solution. **Is your solution dependent on the noise type?**
4. Reduce your model parameters to half in part 3 (the model that is robust against noise), for example by removing some of the layers. Now use a simple knowledge-distillation-based method to train a smaller network on the non-noisy images and investigate its performance against noisy and non-noisy images.
5. Pick an image that contains both cat and dog classes. You can combine two images for example. Can you interpret the result of the classifier? In other words, which parts of the image led to that decision? **Hint:** Find a tool that can interpret your model for specific input.

Code style

It is optional to use the Python code style and type hints. But comments and docstrings are mandatory.

Dataset:

Use the Kaggle dataset: https://www.kaggle.com/c/dogs-vs-cats

Deliverables:

You can put all the following requirements in one notebook for convenience

1. A very short description of your idea to solve the problem
2. A well-documented code with clear steps:
    a. Preprocessing
    b. Training
    c. Validation
    d. …

Also, provide the following:

A module (python file), so anyone can predict an arbitrary image file with your code. Don't forget the requirements.txt and your model. For example, I can use you module like the following:

```
Import catdog
image_file = '/path/to/image'
label, probability = catdog.predict(image_file)
```