Original Image:

A picture containing outdoor, mammal, elephant, grass

Description automatically generated

Reproduce Demo1:

|  |  |  |  |
| --- | --- | --- | --- |
| Predict Class | #1 Indian\_elephant | #2 tusker | #3 African elephant |
| Grad-CAM |  |  |  |
| Vanilla backpropagation |  |  |  |
| “Deconvnet” |  |  |  |
| Guided back propagation |  |  |  |
| Guided Grad-CAM |  |  |  |

Demo2

Generate Grad-CAM maps for "Indian elephant" (Class Index 385) class, at different layers of ResNet-152.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Layer | relu | Layer1 | Layer2 | Layer3 | Layer4 |
| Grad-CAM |  |  |  |  |  |

Turn the image upside down:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Layer | relu | Layer1 | Layer2 | Layer3 | Layer4 |
| Grad-CAM |  |  |  |  |  |

Blurred with Guassian filter:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Layer | relu | Layer1 | Layer2 | Layer3 | Layer4 |
| Grad-CAM |  |  |  |  |  |

The attention map is still accurate with Gaussian filter and rotation.

References:

[1] Kazuto Nakashima, Available: kazuto1011/grad-cam-pytorch: PyTorch implementation of Grad-CAM, vanilla/guided backpropagation, deconvnet, and occlusion sensitivity maps (github.com)