CHENNAI MATHEMATICAL INSTITUTE

Explainability with Saliency Maps

Deep Learners

Anwesha Paul (MDS202213) | **Jessica Vipin** (MCS202310)

Adarsha Mondal (MDS202205) | **Gourab Ghosh** (BMC202016)

Applied Machine Learning | **Raghav Kulkarni**

Motivation behind visualizing CNNs

1. Despite their excellent performance on computer vision tasks, understanding why they perform well has been a challenge.

2. Visualizing the parameters and extracted features of CNNs can provide useful insights.

3. We will introduce three popular techniques: Class Based Image Generation, Guided Backpropagation, and Saliency Heatmaps, with the main focus on Saliency Map

Uses of Saliency Maps

1. **Debugging and Model Analysis:** Visualizations can help us figure out if a CNN is learning the right things from data.

2. **Network Architecture Design:** By looking inside a CNN with visualizations, we can see which layers are doing the most important work of recognizing objects. This helps us design CNNs better by knowing which parts are essential and which might not be necessary.

3. **Transfer Learning and Domain Adaptation:** When we visualize what a pre-trained CNN has learned, we can decide how well those lessons can be used in new situations or tasks.

4. **Explainable AI and Trust:** Making CNNs understandable through visualizations is like showing their work. This is vital in areas like healthcare or finance, where we need to be sure the AI is making decisions for the right reasons.

5. **Human-in-the-Loop Training:** Visualizations can point out where a CNN is struggling, helping humans guide its learning.

Class-Based Image Generation

1. This technique helps visualize how a neural network "sees" a particular class.

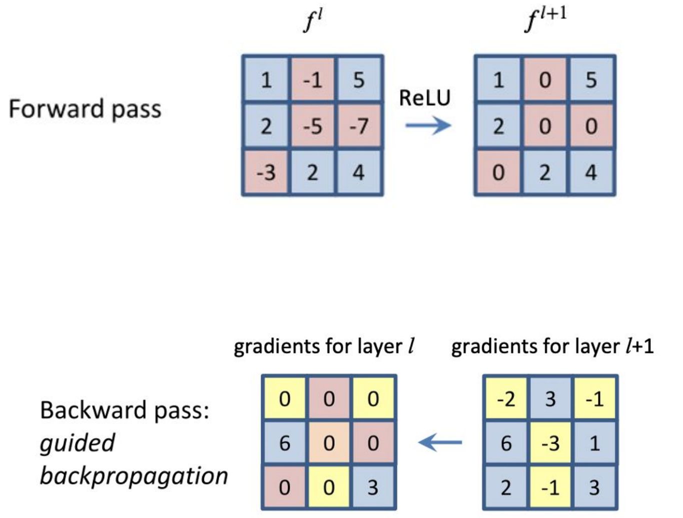
2. A pre-trained CNN with frozen weights is used to optimize a random input image so that the logits (output values) corresponding to a specific class are maximized.

3. We compare this visualization method using a ***custom CNN*** trained on images of cats and dogs, and the ResNet models ***ResNet18*** and ***ResNet50*** trained on the ImageNet Dataset.

4. ***ResNet50*** performed better, possibly due to both its architecture as well as the fine-grained nature of the ImageNet dataset.

Guided Backpropagation

1. This method visualizes the features extracted in the convolutional layers of the CNN.
2. A forward pass is performed until the layer of interest, and then the gradients are backpropagated in a "guided" fashion.
3. Only the gradients that are positive and whose corresponding input values are also positive are backpropagated through the network.

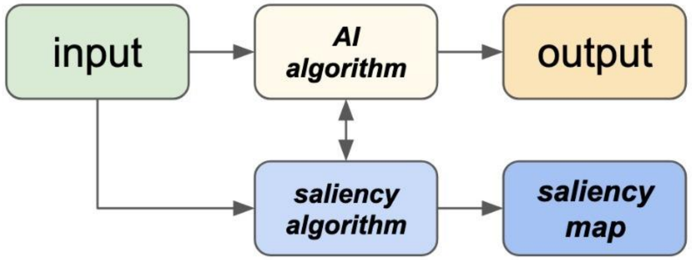


The gradients for both the red and yellow neurons are not backpropagated.

Saliency Mapping

$$adv\_x = x + \epsilon * \text{sign}(\nabla_x J(\theta, x, y))$$

1. Saliency mapping identifies the parts of an image that are most important for a CNN's decision.
2. It generates a heatmap that highlights the regions of the image that contributed the most to the output.
3. Saliency mapping can be applied to any layer of a CNN.
4. An AI model pipeline typically involves an algorithm that transforms input data into output predictions. When augmented with a saliency algorithm, a visual explanation in the form of a saliency map is also produced, which can provide a user additional insight into how the algorithm generated its output.



How does Saliency Mapping Work?

- The aim of the Saliency Mapping is to get an idea of how sensitive a pixel is in classifying the Image as class c
- We need to find the locally optimal Image I' from Image I w.r.t S_c(Score w.r.t class c) (i.e, a very close neighbour of I with a high Score) i.e, $\arg\max_I S_c(I) - \lambda\|I\|_2^2,$
- This is calculated for saliency map using back propagation
- It computes the gradients of the output with respect to the input image.
- The gradients indicate how much each pixel in the input image contributes to the output.
- The gradients are then used to generate a heatmap that highlights the salient regions of the image.

Mathematics behind Saliency Mapping

The technique is based on computing the gradient of the model's output with respect to the input image pixels.

Steps are as following:

1. **Forward Pass:** Calculate the prediction/output of the input image through the model. From the output, you can calculate the score ***S_c*** for the target class ***c*** is extracted.
2. **Backward Pass:** The gradients of the target class score ***S_c*** with respect to the input image pixels I are computed using backpropagation. Mathematically expressed as:

$$\frac{\partial S_c}{\partial I} = \nabla_I S_c = \left[\frac{\partial S_c}{\partial I_1}, \frac{\partial S_c}{\partial I_2}, \ldots, \frac{\partial S_c}{\partial I_n}\right]$$

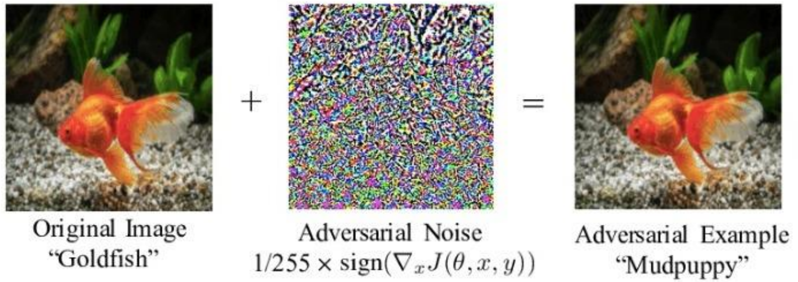
Consider linear Scoring function . $S_c(I) = w_c^T I + b_c$ Then the back propagation would give us *w_c*

3. **Absolute Values:** We take the absolute value of each gradient.
4. **Max pooling**: We find the max over the channels for each pixel, to know which colour was the most influential
5. Normalization Of Saliency Map
6. Saliency Map is visualized to show the regions of the image that contribute the most

Fast Sign Gradient Attack (FSGA)

The FSGA algorithm works as follows:

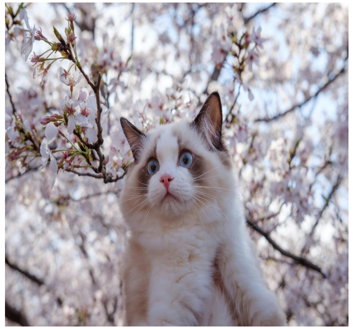
1. Given an input image and a target CNN model, the algorithm computes the gradient of the loss function with respect to the input image.
2. It then takes the sign of the gradient, which provides the direction in which the input image should be perturbed to maximize the loss function (and potentially cause misclassification).
3. The algorithm generates an adversarial example by adding a small perturbation in the direction of the sign of the gradient to the original input image.
4. The magnitude of the perturbation is constrained to ensure that the adversarial example remains visually similar to the original image.



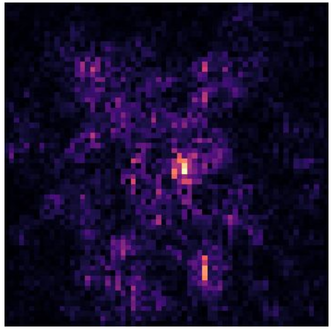
Key Advantages of the Fast Sign Gradient Attack

1. **Computational Efficiency:** FSGA is computationally efficient compared to other adversarial attack methods, as it only requires computing the gradient of the loss function once.
2. **High Success Rate:** FSGA has a high success rate in generating adversarial examples that can fool machine learning models, even with small perturbations.
3. **Transferability:** The adversarial examples generated by FSGA can often transfer across different models, making it a versatile attack method.
4. **Low Perturbation:** FSGA can generate adversarial examples with small, imperceptible perturbations, making the attack more practical and challenging to defend against.
5. **Simplicity:** FSGA is a relatively simple and straightforward algorithm, making it easy to implement and understand.

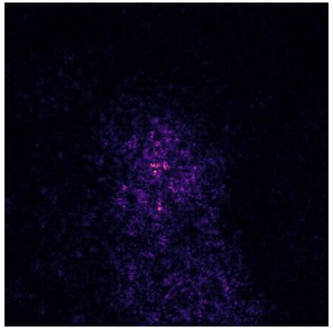
RESULTS



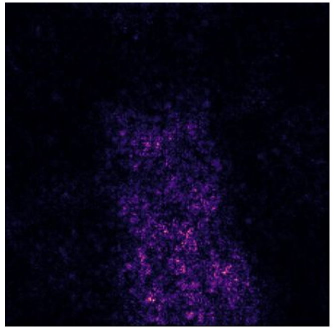
CAT



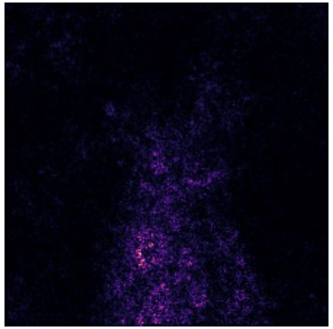
Custom CNN



ResNet18

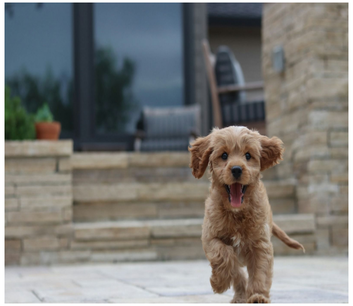


ResNet50

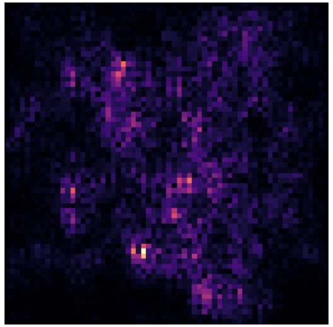


VGG16

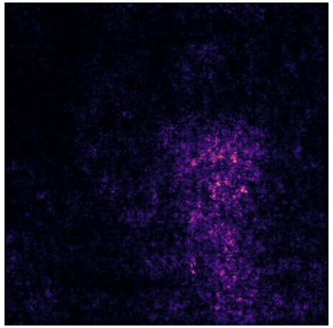
RESULTS



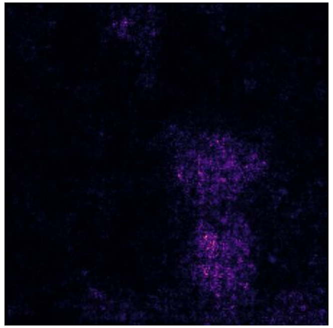
DOG



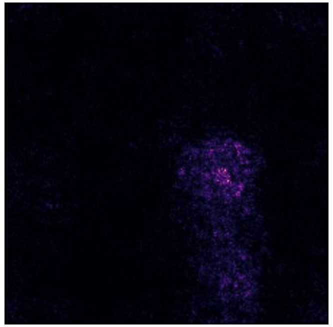
Custom CNN



ResNet18

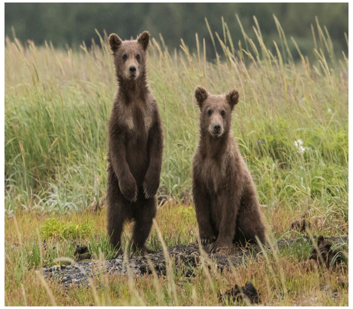


ResNet50

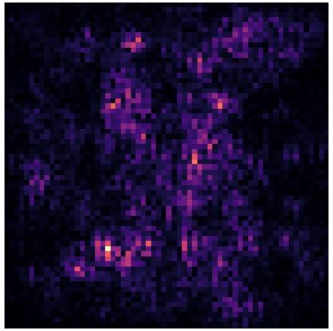


VGG16

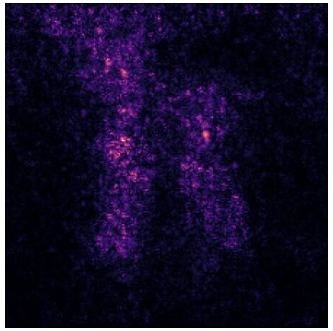
RESULTS



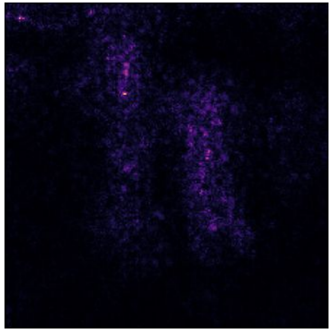
BEAR



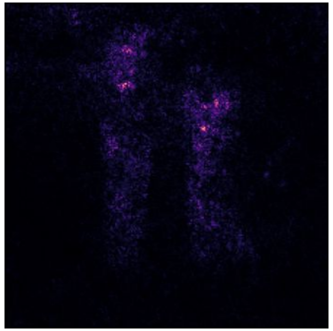
Custom CNN



ResNet18

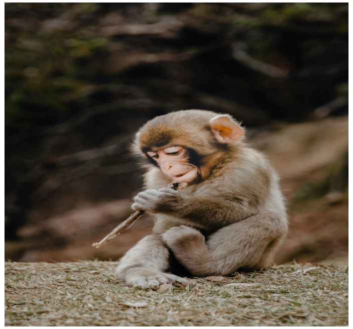


ResNet50

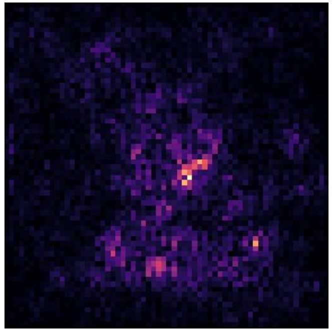


VGG16

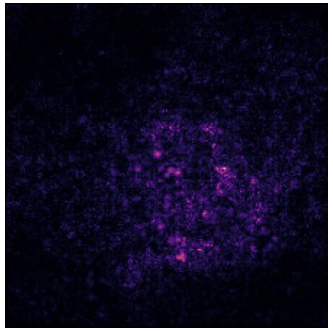
RESULTS



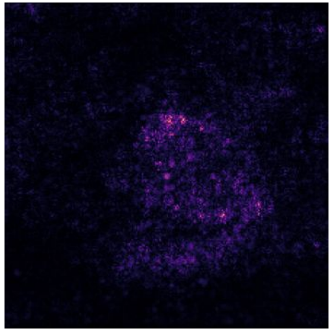
MONKEY



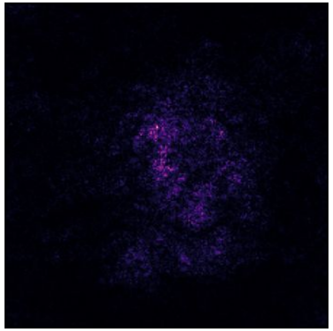
Custom CNN



ResNet18

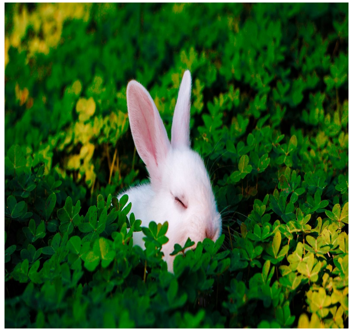


ResNet50

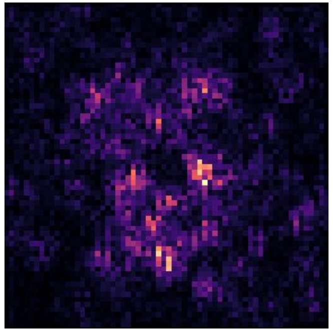


VGG16

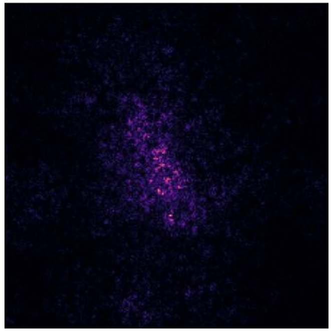
RESULTS



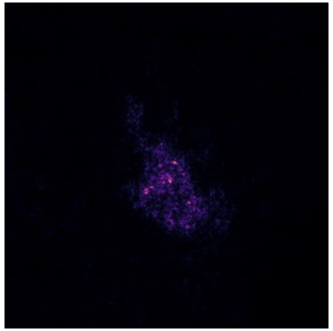
RABBIT



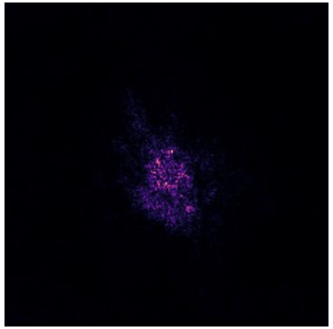
Custom CNN



ResNet18



ResNet50



VGG16

CONTRIBUTIONS

	PRIMARY CONTRIBUTION	SECONDARY CONTRIBUTION
ANWESHA PAUL	Saliency Maps with ResNet18, ResNet50	Presentation Slides
ADARSHA MONDAL	Presentation Slides	-
JESSICA VIPIN	Saliency Maps with Custom_CNN, VGG16	Presentation Slides
GOURAB GHOSH	Saliency Maps with Custom_CNN, VGG	Presentation Slides