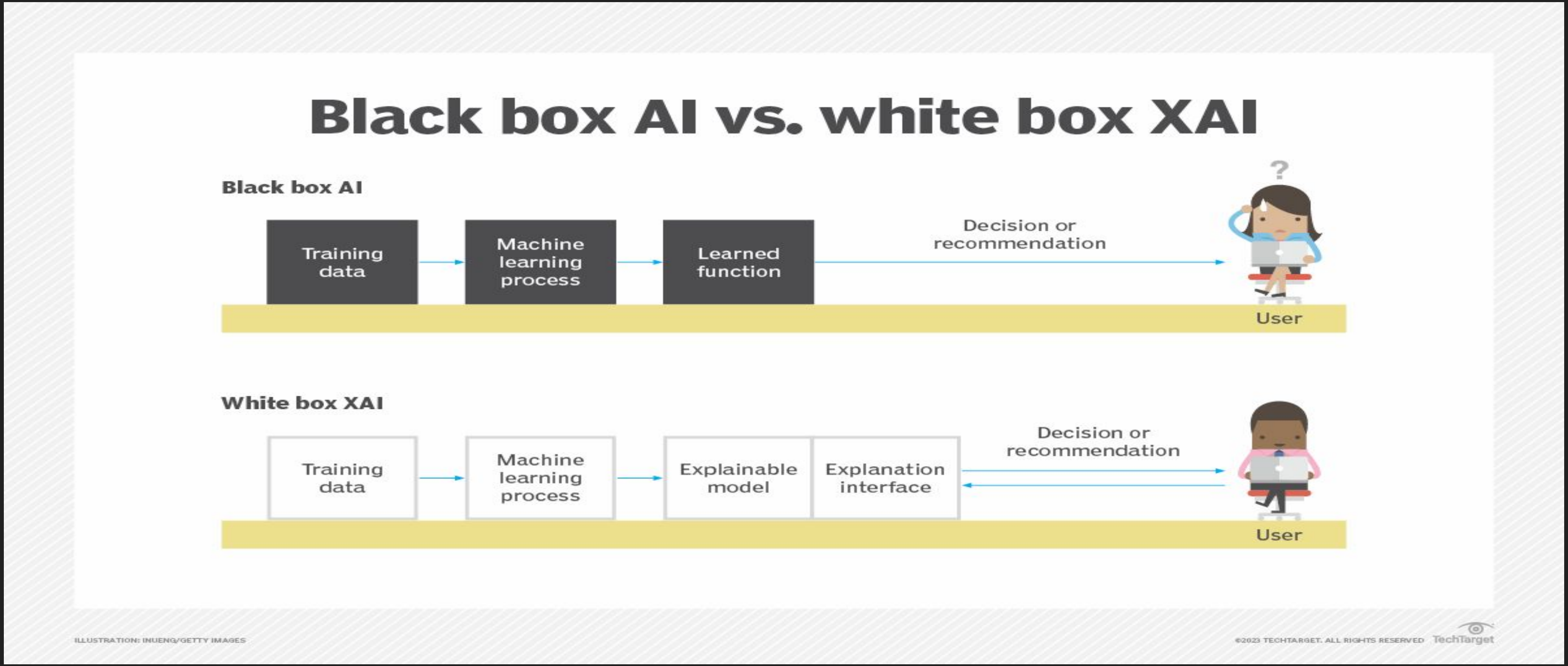
Explainability for Deep Learning Models

Group 5:
Manoj Padala
Ram Mannuru
Sailesh

WHAT & WHY??

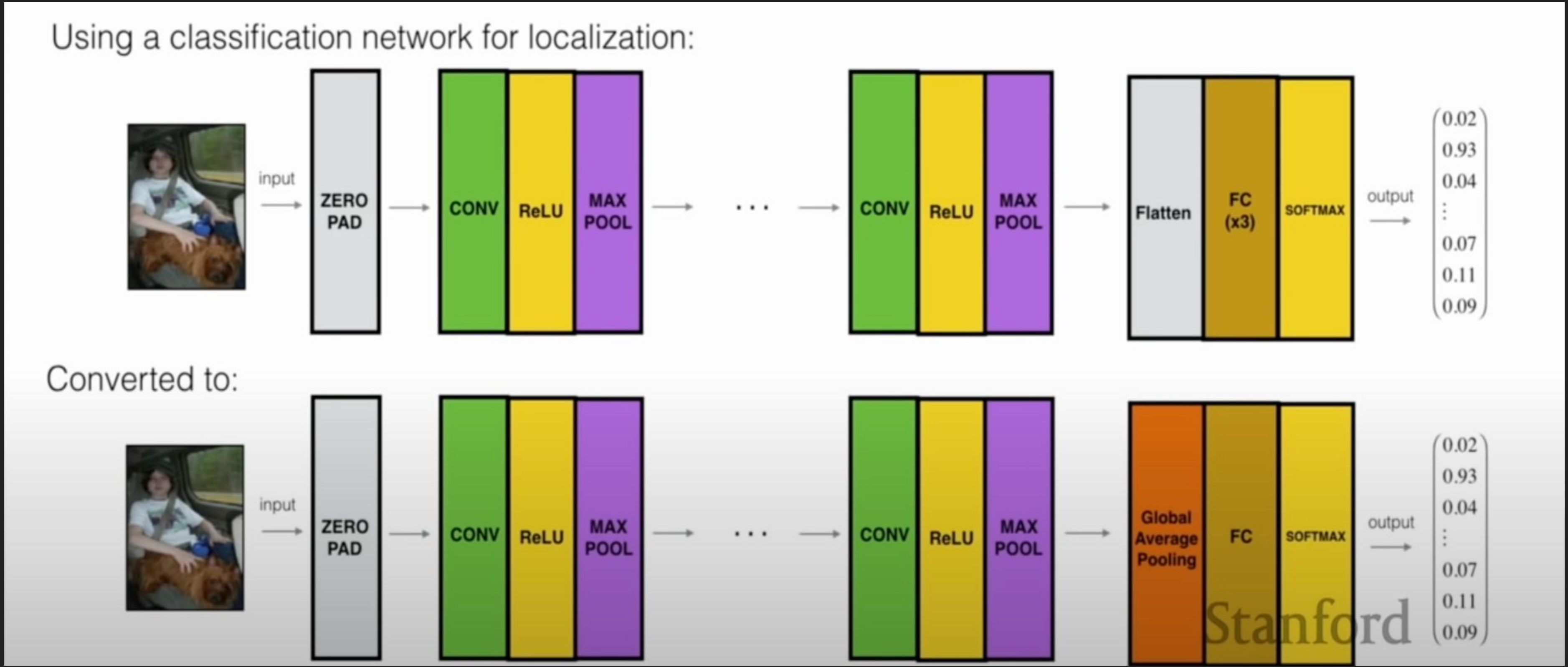


Black box AI vs. white box XAI
Black box AI
Training data
Machine learning process
Learned function
Decision or recommendation
?
User
White box XAI
Training data
Machine learning process
Explainable model
Explanation interface
Decision or recommendation
User
ILLUSTRATION: INUENG/GETTY IMAGES
©2023 TECHTARGET. ALL RIGHTS RESERVED
TechTarget

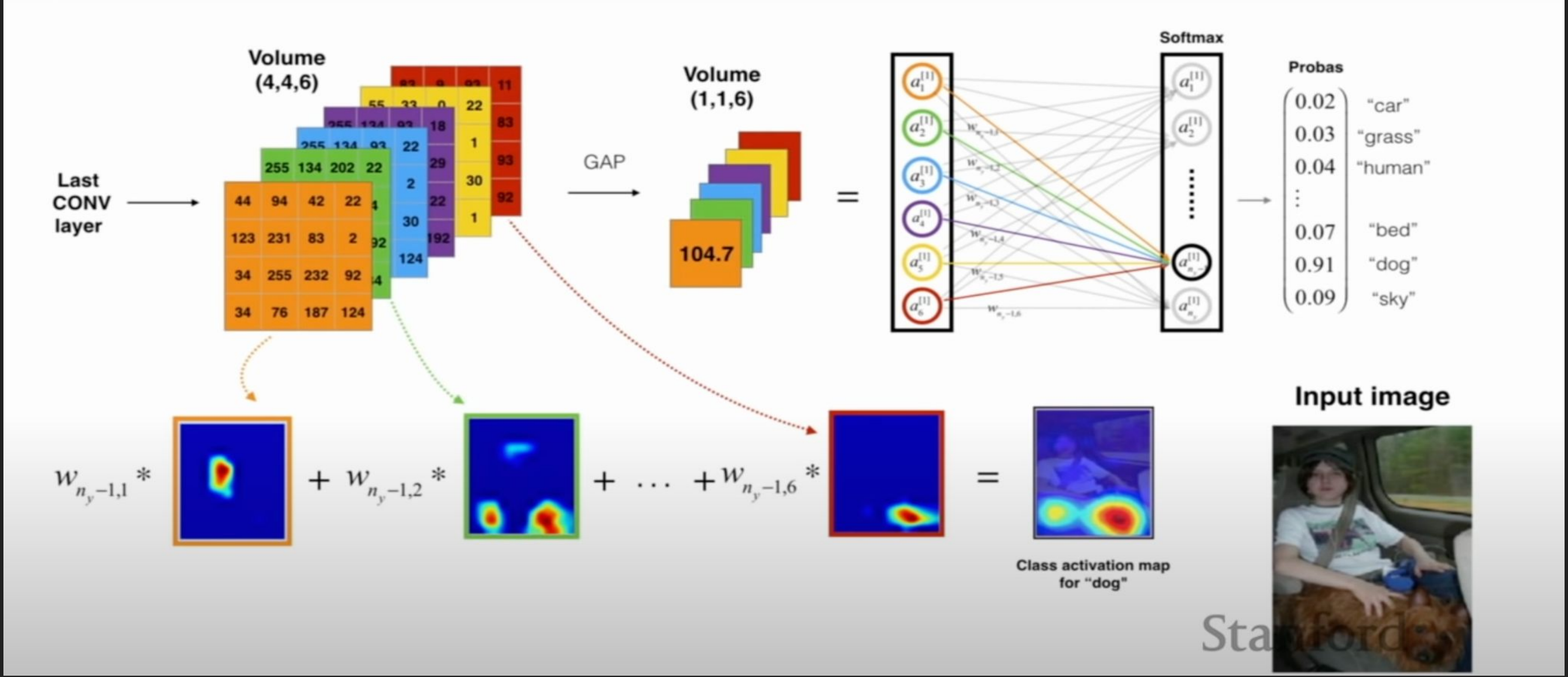
XAI Models

1. Class Activation Maps
2. Deep Feature Factorization
3. LIME and SHAP
4. GradCam
5. Occlusion Sensitivity

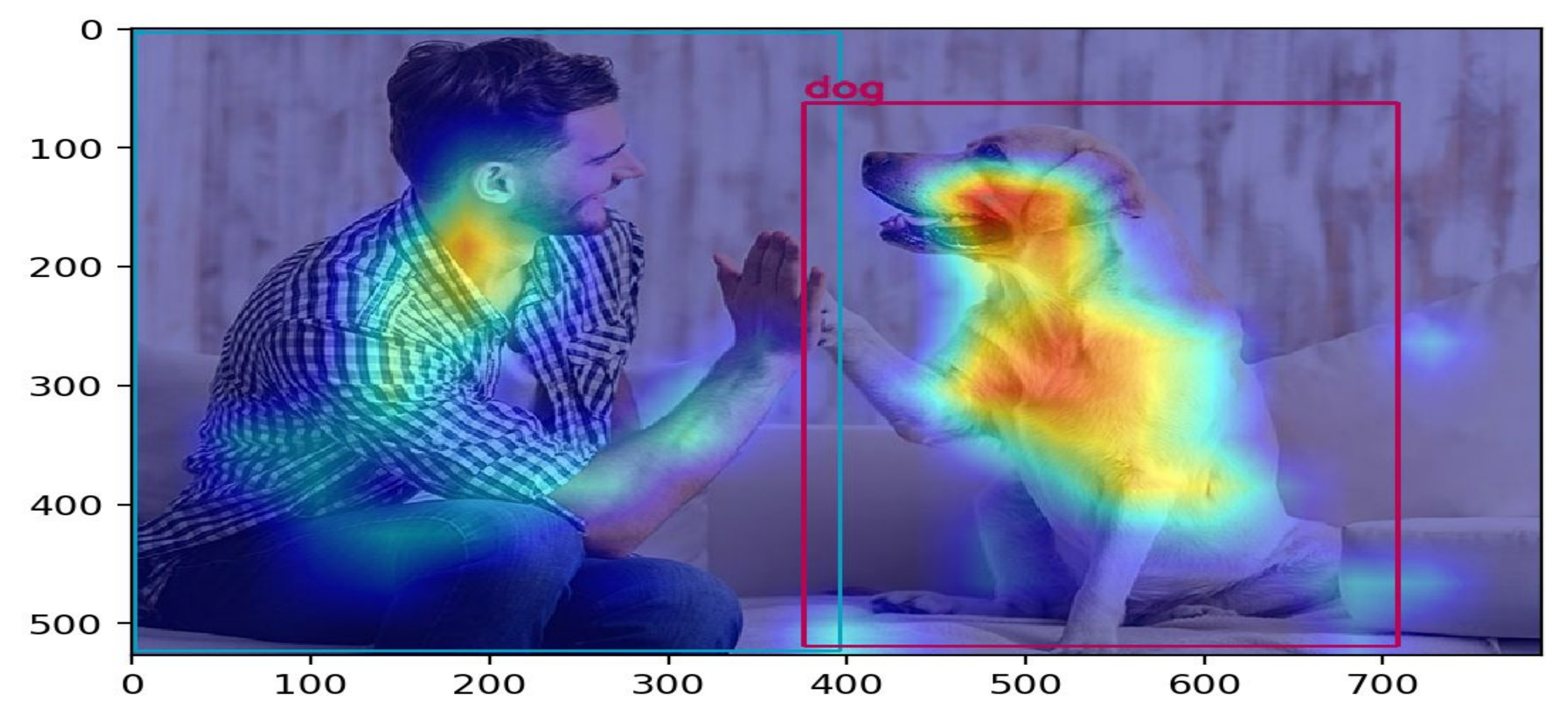
Interpreting Neural networks using Class Activation Maps:



Interpreting Neural networks using Class Activation Maps:



Dashboard:



What do Models See?

Interpreting NNs using Deep feature factorization:

1.Model Selection: Pre Trained (CNNS)

2. Target Layer Selection : Typically deeper layer where representations are more abstract.

3. Computation On Concepts: Analysing activations of neurons.

4. Factorization: decomposing a matrix to understand more.

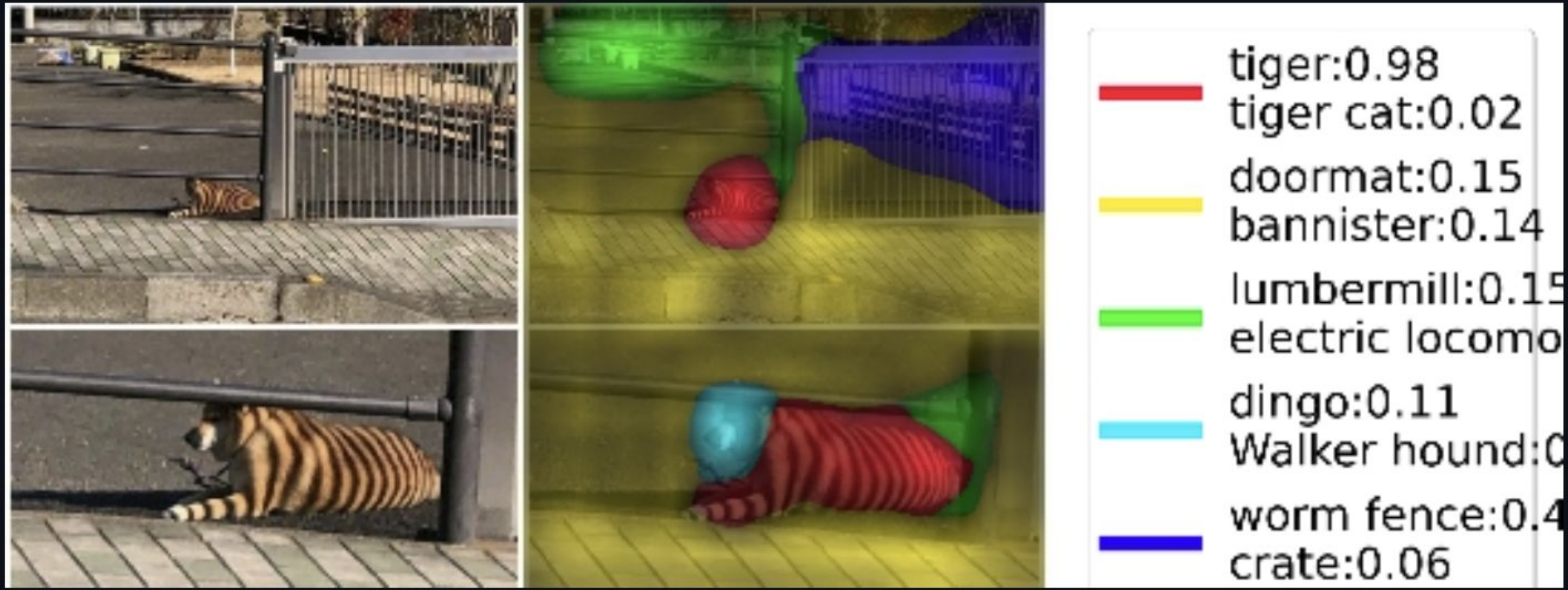
Dashboard:

Uploaded Image

Predicted class using Resnet


```
[
  0 : [
    0 : "n02129604"
    1 : "tiger"
    2 : "0.9453495"
  ]
  1 : [
    0 : "n02123159"
    1 : "tiger_cat"
    2 : "0.05418514"
  ]
]
```

Why is that result??



Visualization Result

LIME Image Classifier

LIME - **L**ocal **I**nterpretable **M**odel agnostic **E**xplanation

Model agnostic - Pre Trained, Custom or Pre Trained + Custom

Interpretable Representation

Real Time Usage of LIME - HealthCare Decision Support System, Financial Fraud Detection and in some IOT industries.

INPUT IMAGE

How **LIME** predicts?

SEGMENTED IMAGE

SUPERPIXELS TURNED OFF

Model Prediction

Segmentation of the Image

Generate Preturbations

Model Re-evaluation

Train the Local Model

Explanation Generation

Visualization and Interpretation

AI Explainability Dashboard

LIME for Image Classification Model

Select the Deep Learning framework:

- ○ PyTorch
- ● TensorFlow

Pretrained

- ● Pre-trained

Instantiate pre-trained model with corresponding weights. Note: write full library. TensorFlow as tf and torch as torch.

```
model = tf.keras.applications.MobileNetV2(weights='imagenet')
```

```
model = tf.keras.applications.MobileNetV2(weights='imagenet')
```

Enter the image size for your model (Note: For pre-trained models, it must match with image size that was used to train the model)

224

Enter your desired image normalization - Mean

0.5, 0.5, 0.5

Enter your desired image normalization - Standard Deviation

0.5, 0.5, 0.5

Applied pre-processing

```
torchvision.transforms.Compose([torchvision.transforms.ToTensor(),
torchvision.transforms.Resize((224, 224)),
torchvision.transforms.Normalize(
mean=[0.5, 0.5, 0.5],
std=[0.5, 0.5, 0.5])])
```

Upload the image you want to explain

Drag and drop file here

Limit 200MB per file • JPG, JPEG, PNG

Browse files

photo-1560275619-4662e36fa65c.jpg 166.4KB

Uploaded Image

Your Predicted Output from the model is as follows:

0	1	2	3	4	5	6	7	8	9	10	11	12
0.0002	0.0002	0.0029	0.8631	0.0201	0.0021	0.0046	0	0.0001	0.0001	0.0001	0.0001	0.0001

Explain Model

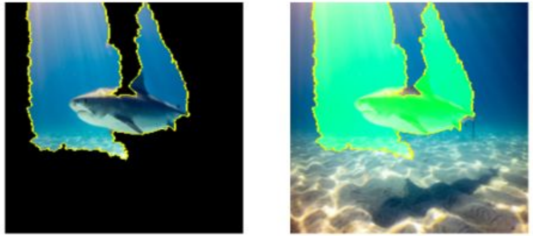
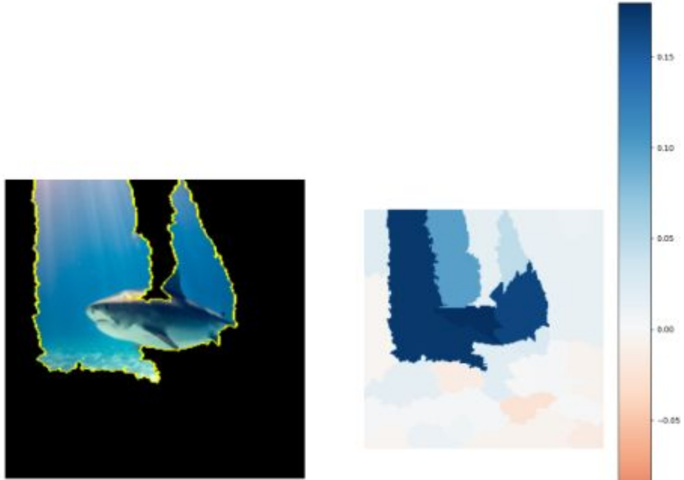


Image on the left denotes the super-pixels or region-of-interest based on LIME analysis. Classification is done due to the highlighted super-pixels. Image on the right imposes this region-of-interest on original image giving a more intuitive understanding.



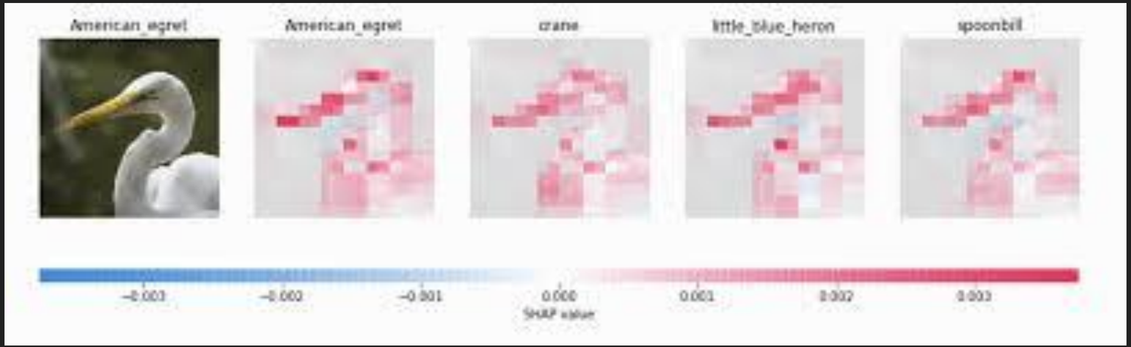
This section shows a heat-map that displays how important each super-pixel is to get some more granular explaianbility. The legend includes what color-coded regions of interest move the decision of the model. Blue indicates the regions that influences the decision of the model in the predicted class and red indicates the regions that influence the decision to other classes.

SHAP Image Classifier

SHAP -**SH**apley **A**dditive ex**P**lanation

Shapley Value

Real Time Usage of LIME - Credit Scoring, E-commerce Product



How does SHAP work?

Feed an Image

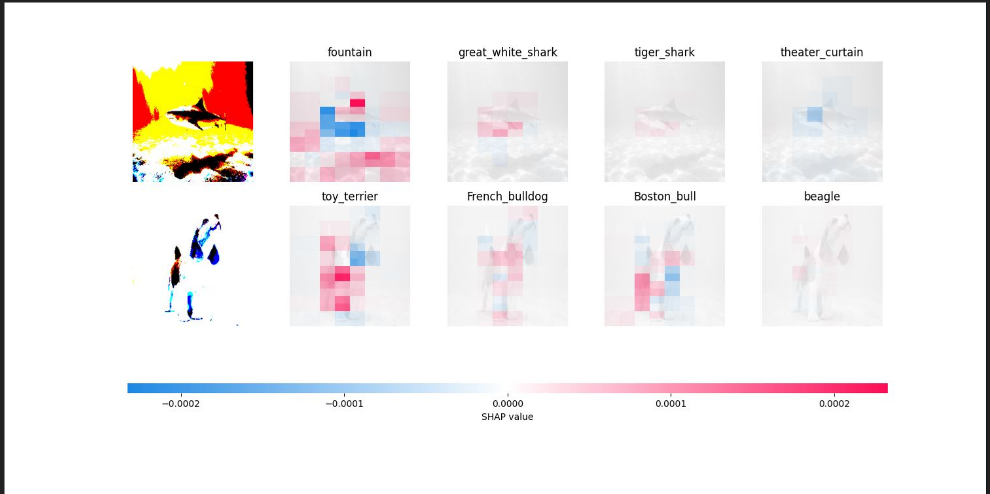
Pixel-Level Feature analysis

Create Perturbations and Recalculate Prediction

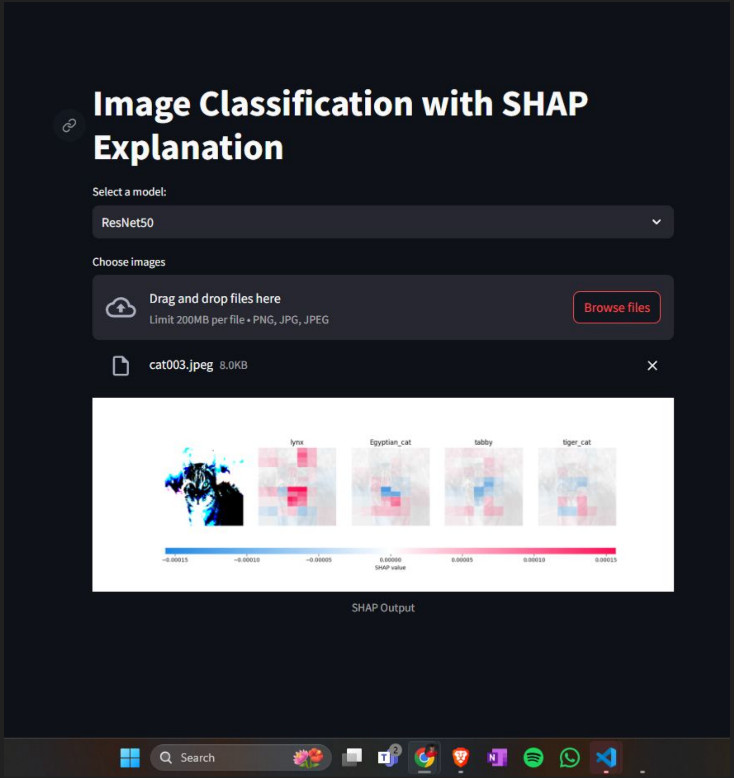
Calculate SHAP Values

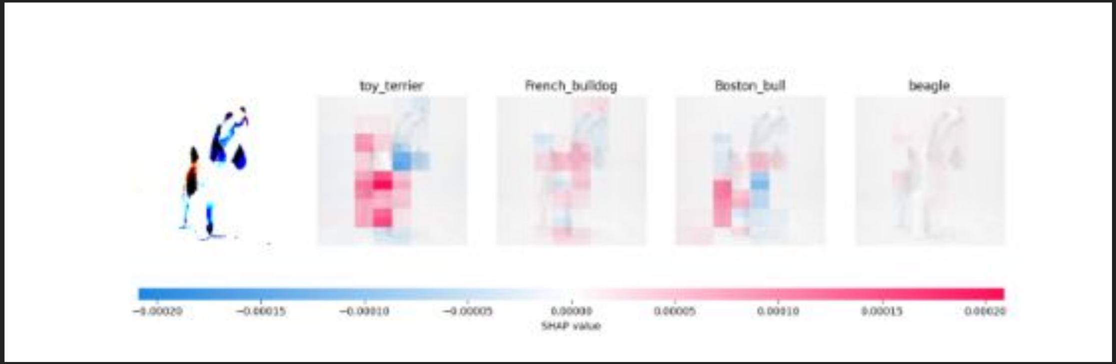
Aggregate SHAP Values and Analyze

Color Coding



SHAP integrated with Streamlit





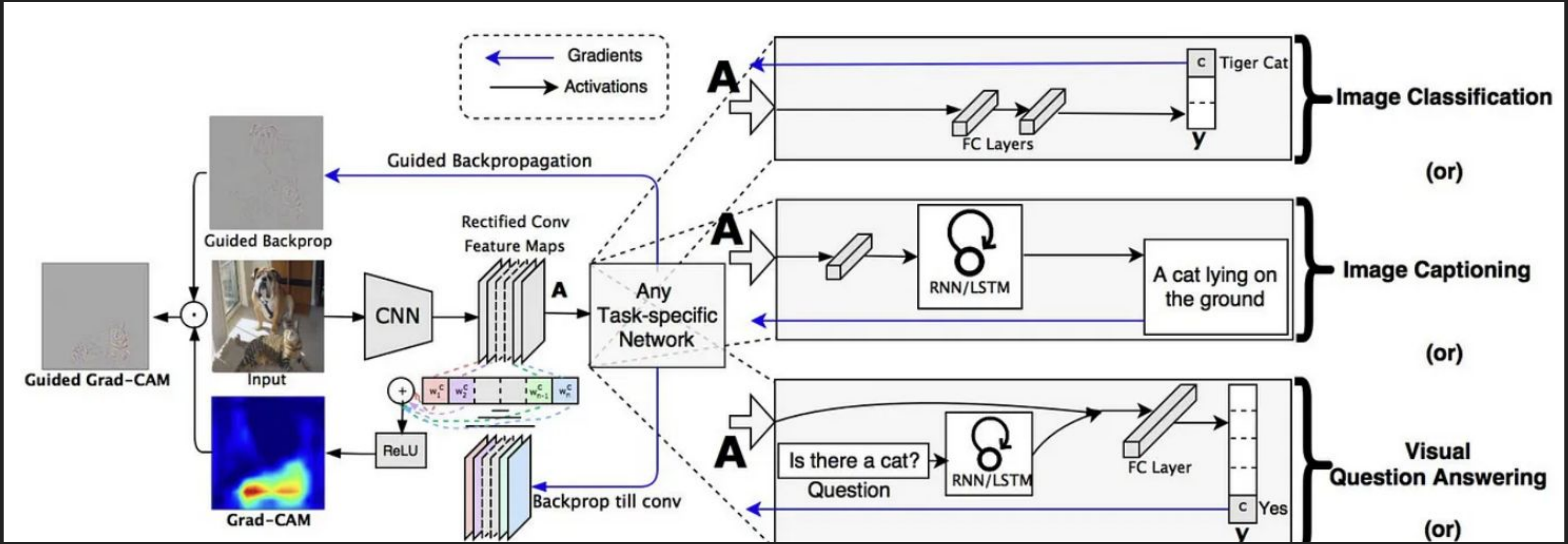
XAI for Convolution

- GRADCAM
- OCCLUSION SENSITIVITY

DEEP LEARNING EXPLAINABILITY

GRADCAM

- Forward Pass
- Gradient Calculation
- Global Average Pooling (GAP)
- Weighted Combination
- ReLU and Upsampling
- Visualization



GRADCAM

1. Forward Pass: Feed Input to last layer of CNN to obtain feature maps

   Ak= Activation functions

2. Gradient Calculation: $\frac{\partial Y_c}{\partial A^k}$
3. Global Average Pooling (GAP):

   Compute the importance of each feature map by taking the global average of the gradients.

$$\alpha_k = \frac{1}{Z} \sum_i \sum_j \frac{\partial Y_c}{\partial A^k_{ij}}$$

4. Weighted Combination:

   Weight the feature maps by their important scores

$$L^c_{\text{Grad-CAM}} = \text{ReLU}\left(\sum_k \alpha_k A^k\right)$$

OCCLUSION SENSITIVITY

- Model Interpretability
- Detection of Important features
- Localization
- Model Improvement
- Debubbing
- Comparing models

- Input Window
- Occlusion Window
- Sliding Window
- Move Window (Stride)
- Comparing models

OCCLUSION SENSITIVITY

1. Input Image: I_{ij}

2. Model Prediction: Output which represents class probabilities $f(I)$

3. Occlusion Window: size (p,q)

4. Occlusion value: Replaces the value with pixels of input image

5. Occlusion Process: Input, slide occlusion window with stride, Replace pixels and compute model prediction

6. Occlusion sensitivity map

7. Normalization and Visualization.

GRADCAM

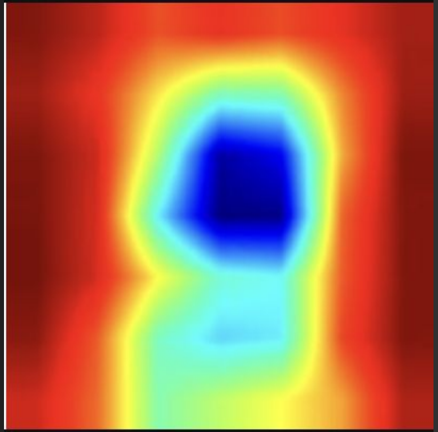
Gradient-weighted Class Activation Mapping

Input Image:

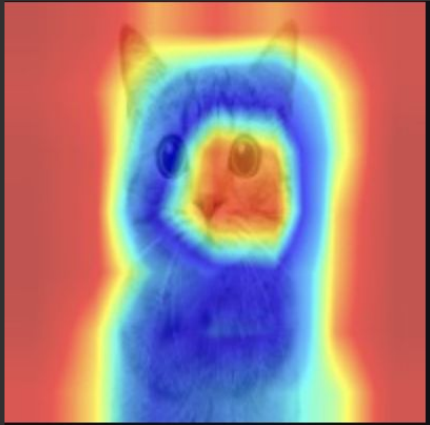
Predicted Class: tabby

Probability: `0.65993613`

Model Used: ResNet50



GradCAM Heatmap



GradCAM Overlayed Image

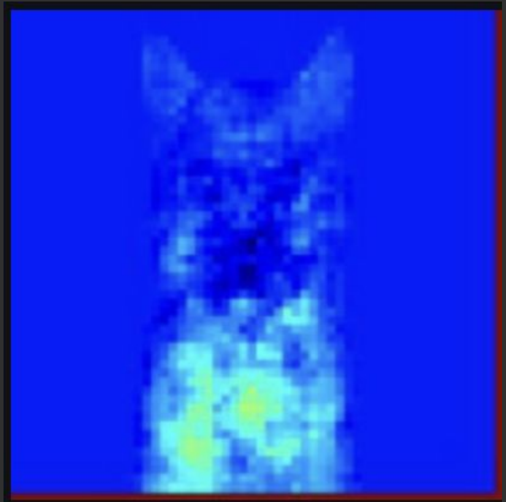
Occlusion Sensitivity

Model Used: ResNet50

Input Image:

Predicted Class: tabby

Probability: `0.65993613`



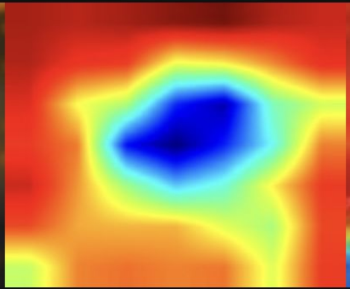
Occlusion Sensitivity Heatmap

Occlusion Sensitivity

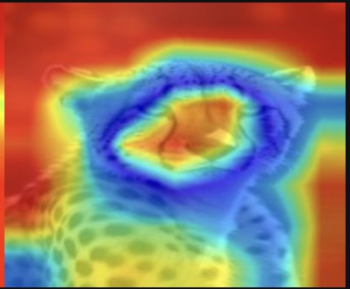
Predicted Class: cheetah

Probability: `0.97878623`

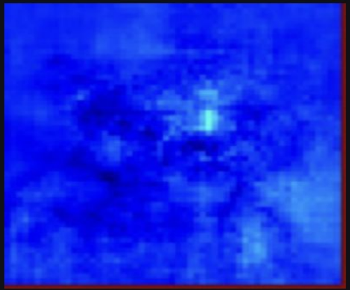
Original Image



GradCAM Heatmap



GradCAM Overlayed Image



Occlusion Sensitivity Heatmap

THANK YOU!!!