Explainable AI - Understanding and Interpreting Computer Vision Models

Gunti Lahari
IIT Hyderabad
ai22btech11008@iith.ac.in

J Hima Chandh
IIT Hyderabad
ai22btech11009@iith.ac.in

S Divija
IIT Hyderabad
ai22btech11026@iith.ac.in

K Anuraga Chandan
IIT Hyderabad
ai22btech11011@iith.ac.in

***Abstract*—This project explores Explainable AI (XAI) techniques to help us understand how computer vision models, like YOLOv5 and DeepLabV3, make decisions. We applied methods such as Eigen-CAM, Grad-CAM, Grad-CAM++, and BODEM to highlight image regions that influence model predictions. We also studied a new approach combining Grad-CAM++ with Mask R-CNN and classic machine learning to improve both detection and explanation. Using evaluation metrics and comparisons with human attention data, we assessed how well these techniques align with human reasoning. Our work aims to make AI models more transparent and trustworthy, especially in real-world applications like healthcare and autonomous systems.**

Keywords—XAI(Explainable AI),Grad-CAM ,Grad-CAM++, Human Attention ,Deep Learning , saliency map ,Computer vision,Black-box testing ,Object detection,Hierarchical masking

I. Progress after midterm

Following the midterm of the project, we extended our work to explore various XAI techniques particulary for object detection.

We implemented and evaluated Eigen-CAM on object detection models, notably YOLOv5, using the pytorch-grad-cam library.

In addition, we applied SHAP (SHapley Additive exPlanations) to object detection outputs.

We also experimented with Grad-CAM in the domain of semantic segmentation, using models such as Deeplabv3_resnet50. T

Toward the conclusion of the project, we studied a novel approach introduced in [1], which proposes Grad-CAM for Multi-Label RCNN (GradCAM-MLRCNN).

II. Implementations

First let us see CAM Methods.

A. *Eigen-CAM on YOLOv5*

Eigen-CAM (Eigenvalue-based Class Activation Mapping) is a class-discriminative visualization technique that highlights salient regions in an image without requiring gradient computation. Unlike Grad-CAM, which uses backpropagation to compute gradients with respect to a class label, Eigen-CAM utilizes the principal components of the feature maps from convolutional layers. This gradient-free approach makes it highly efficient and less noisy.

1) Mathematical Formulation of Eigen-CAM: Let $F \in \mathbb{R}^{C\times H\times W}$ be the activation tensor from a selected convolutional layer, where C is the number of channels, and $H \times W$ is the spatial resolution. The activation tensor is reshaped to flatten the spatial dimensions:

$$F' = \text{reshape}(F) \in \mathbb{R}^{C\times(H\cdot W)}$$

The covariance matrix of the feature channels is computed as:

$$\Sigma = F' \cdot (F')^\top \in \mathbb{R}^{C\times C}$$

We perform eigen decomposition on the covariance matrix:

$$\Sigma \cdot v_i = \lambda_i \cdot v_i$$

where v_i and λ_i represent the i-th eigenvector and its corresponding eigenvalue, respectively. The eigenvector v_1 with the largest eigenvalue λ_1 is selected as the dominant component.

The Eigen-CAM heatmap is computed by projecting the activation tensor onto this principal component:

$$\text{CAM}_{\text{eigen}} = v_1^\top \cdot F'$$

Finally, the result is reshaped back to the spatial resolution $H \times W$, and normalized for visualization.

In this work, Eigen-CAM was integrated into the YOLOv5 object detection framework using the *pytorch-grad-cam library*. YOLOv5, a fast and accurate object detector, was selected due to its widespread adoption in real-time computer vision tasks. The intermediate convolutional layers of the YOLOv5 backbone were selected as target layers for generating Eigen-CAM heatmaps.

The following steps were used in the pipeline:

1) **Model Loading**: A pretrained YOLOv5 model was loaded using the Ultralytics repository.
2) **Layer Selection**: A convolutional layer in the backbone (typically `model.model[6]` or similar) was chosen as the target for visualization.
3) **Input Processing**: Input images were resized to human attention data size(576,1024) and normalized to match YOLOv5's preprocessing.
4) **Heatmap Generation**: The `EigenCAM` method computed the principal components of the feature maps,

and the dominant eigenvalue map was visualized as a heatmap. and displayed

Eigen-CAM proved especially effective in producing smoother and more interpretable saliency maps for object detection tasks. Unlike gradient-based methods, it avoids issues such as noisy activations and misaligned class focus.

B. *Grad-CAM for Semantic Segmentation*

Gradient-weighted Class Activation Mapping (Grad-CAM) is a widely used technique for visualizing the regions of an input image that are most influential for a model's predictions. While originally developed for image classification tasks, Grad-CAM can be adapted for semantic segmentation models to provide pixel-level interpretability.

In this work, we applied Grad-CAM to a semantic segmentation model using the `pytorch-grad-cam` library. The process involves computing the gradients of the target class scores with respect to the feature maps of a selected convolutional layer. These gradients are then used to weight the feature maps, highlighting the areas of the image that contribute most to the prediction of each class.

Mathematical formulation was discussed in midterm report

Implementation Steps:

1) **Model Selection:** A pre-trained semantic segmentation model, such as DeepLabV3 with a ResNet-50 backbone, was utilized.
2) **Target Layer Identification:** A suitable convolutional layer was chosen as the target for Grad-CAM. This is typically one of the deeper layers in the network that captures high-level features.
3) **Target Definition:** For semantic segmentation, the target is defined as the class of interest at a specific spatial location. This involves creating a custom target that specifies both the class index and the pixel location.
4) **Grad-CAM Computation:** The `GradCAM` class from the `pytorch-grad-cam` library was instantiated with the model and target layer. The custom target was passed to the `__call__` method to compute the class activation map.
5) **Visualization:** The resulting heatmap was overlaid on the original image to visualize the regions that most influenced the model's prediction for the specified class at the given location.

C. *GradCAM-MLRCNN*

This is a new method from[1] which is not explainable method but done using explainable ai. This paper combines

- **Grad-CAM++**: An advanced visualization technique that enhances object localization by generating refined heatmaps.
- **Mask R-CNN**: A robust framework for object detection and instance segmentation.
- **Machine Learning Algorithms**: Including logistic regression, decision trees, Gaussian classifiers, k-means clustering, and k-nearest neighbors (KNN) to bolster classification accuracy.

1) Methodology: The proposed approach operates as follows:

1) **Feature Extraction**: Utilizes pre-trained convolutional neural networks (CNNs) such as VGG16, VGG19, ResNet101, and ResNet152 to extract rich feature representations from input images.
2) **Object Localization**: Applies Grad-CAM++ to the extracted features to generate heatmaps that highlight regions of interest, effectively localizing objects within the images.
3) **Object Detection and Segmentation**: Employs Mask R-CNN to detect objects and generate precise segmentation masks, leveraging the localized information from Grad-CAM++.
4) **Classification Enhancement**: Integrates traditional machine learning classifiers to improve the accuracy of object classification, particularly in complex scenarios.

points to be noted

- **Grad-CAM++** is chosen over Grad-CAM because it works better when multiple objects are present.
- **Mask R-CNN** provides pixel-level precision — not just bounding boxes.
- **Machine Learning models** give flexibility in improving the final prediction (different classifiers can be used based on the task)

D. *BODEM-SSD*

1) **Implementation Details** The Black-box Object Detection Explanation by Masking (BODEM) method was implemented using Python to generate saliency maps for object detection models. The implementation utilized several libraries: PyTorch for the object detection model (SSD300 with VGG16 backbone), OpenCV for image processing, NumPy for numerical computations, and Matplotlib for visualization.The PASCAL VOC 2012 was used, with the model trained on the 'trainval' split and evaluated on the 'val' split.
2) **Object Detection Model Setup** The SSD300 model with a VGG16 backbone was employed as the object detector, consistent with the vehicle detection task. The model was initially loaded with pretrained weights from the ImageNet dataset, and the classification head was adjusted to accommodate the 21 classes of the PASCAL VOC dataset (20 object classes plus background). The backbone was frozen for the first 5 epochs to stabilize training, after which all layers were unfrozen for fine-tuning. The model was trained for 10 epochs using a batch size of 16, with gradient accumulation over 2 steps to manage memory constraints. An SGD optimizer was used with a learning rate of 0.0001, momentum of 0.9, and weight decay of 1e-4. A step learning rate scheduler reduced the learning rate by a factor of 0.1 every 5 epochs. The trained model was saved and reloaded for inference, achieving object detection on the test image with a confidence threshold of 0.3.

3) **Saliency Map Generation with BODEM**

The BODEM method generates saliency maps through a hierarchical random masking strategy, consisting of three main stages: mask generation, model inquiry, and saliency estimation. The process was implemented to operate in a black-box manner, requiring only the input image I of dimensions $W \times H$ (resized to 512×512) and the detected bounding boxes $O = \{o_1, o_2, \ldots, o_N\}$, where each $o_n = (x_1, y_1, x_2, y_2)$, without access to the model's internals, class probabilities, or objectness scores.

The mask generation phase divides the image into blocks at each level l (from 1 to $L = 6$), with block size $BS^l = K/2^{l-1}$, where $K = 128$. At $l = 1$, blocks are 128×128, and by $l = 6$, they are 4×4, matching the experimental setup in Moradi et al. [**?**]. A set of candidate seed blocks CS is initialized with all blocks at $l = 1$, or blocks with non-zero saliency from the previous level for $l > 1$. A seed block is selected randomly, weighted by prior saliency for $l > 1$, and 50% of its neighbors within distance l are masked. The process terminates when CS is empty (for $l = 1$) or when all blocks with non-zero prior saliency are masked (for $l > 1$).

The model inquiry phase applies each mask m_q to the image, setting masked block pixels to zero, and queries the SSD model to obtain new bounding boxes $o'_n = (x'_1, y'_1, x'_2, y'_2)$. The saliency estimation phase computes the importance of each block by comparing original and masked predictions. The similarity between the original box o_n and the new box o'_n is calculated using Intersection over Union (IoU):

$$\text{Similarity}(o_n, o'_n) = \text{IoU}(o_n, o'_n) = \frac{|o_n \cap o'_n|}{|o_n \cup o'_n|}$$

The importance score $IS(b_p)$ for a masked block b_p is:

$$IS(b_p) = 1 - \text{Similarity}(o_n, o'_n)$$

An overall importance score $OIS(b_p)$ is computed by averaging $IS(b_p)$ across all masks where b_p is masked:

$$OIS(b_p) = \frac{\sum_{m_q \in M^l} IS(b_p) \mid m_q(b_p) = 1}{\sum_{m_q \in M^l} m_q(b_p) = 1}$$

The saliency map SM_n^l at level l is updated for each block b_p with non-zero prior saliency:

$$SM_n^l[b_p] = \begin{cases} \alpha \cdot SM_n^{l-1}[b_p] \\ +(1-\alpha) \cdot OIS(b_p), & \text{if condition (†)} \\ \beta \cdot SM_n^{l-1}[b_p], & \text{otherwise} \end{cases}$$

(†) $\exists\, m_q \in M^l$ such that $m_q(b_p) = 1$ and $OIS(b_p) \neq 0$

The saliency map SM_n^l at level l is updated for each block b_p with non-zero prior saliency:
the hyperparameters were set to $\alpha = 0.3$ and $\beta = 0.2$, as determined through their hyperparameter tuning experiments. The final saliency map at $l = 6$ was normalized, smoothed with a 5×5 Gaussian blur, and thresholded at the 80th percentile to highlight salient regions.

III. Evaluation Metrics

A. Insertion Score

Measures how quickly the model's confidence increases as important pixels (from the saliency map) are gradually added to a blank image. Higher is better — a high score means the saliency map highlights truly informative regions.

B. Deletion Score

Measures how quickly the model's confidence drops as important pixels are removed from the image. Lower is better — a low score suggests that the removed pixels were critical to the model's decision.

for these both ground truth human attention map is not required. Because insertion/deletion metrics measure how the model's output changes as parts of the saliency map are added/removed.They evaluate faithfulness of the saliency map to the model's own decision.

Some other metrics calculated from human attention data got from[2]

C. Pearson Correlation Coefficient (PCC)

- Measures linear similarity between the saliency map and human attention map.
- Values range from -1 to 1; closer to 1 is better, indicating strong positive correlation.

D. Root Mean Squared Error (RMSE)

- Measures the average difference between the saliency map and human attention.
- Lower is better, meaning the model's explanation is closer to human reasoning.

E. Spearman Rank Correlation Coefficient (SROCC)

- Measures rank-based similarity between model and human attention (not exact values).
- Closer to 1 is better, indicating similar ordering of important regions.

F. Convergence

This metric assesses the stability of the saliency maps by computing the average Euclidean distance between saliency maps generated from three independent runs of the BODEM method on the same input image and object:

$$\text{Convergence} = \frac{\|SM_1 - SM_2\| + \|SM_1 - SM_3\| + \|SM_2 - SM_3\|}{3}$$

where $\|SM_i - SM_j\|$ denotes the Euclidean distance between saliency maps SM_i and SM_j. A lower Convergence value indicates higher stability in the generated saliency maps.

These metrics were computed for each detected object in the test image, and the results were visualized as heatmaps overlaid on the original image. The implementation ensured that metrics were not combined across objects, providing individual insights into the saliency map's effectiveness for each detection.

Formulas and other details are discussed in previous report.

IV. Results

A. Eigen-CAM on YOLOv5

The proposed explainability framework was evaluated using Eigen-CAM on a YOLOv5 object detection model. The approach enabled visualization of spatial regions that most influenced the model's object detection decisions.

1) Visualization of Salient Regions: Using the Eigen-CAM method, heatmaps were generated for detected objects and overlaid on the input image. The highlighted regions correlate with the most influential areas used by the model to detect each object. The visualizations indicate strong semantic alignment between model focus and object contours.

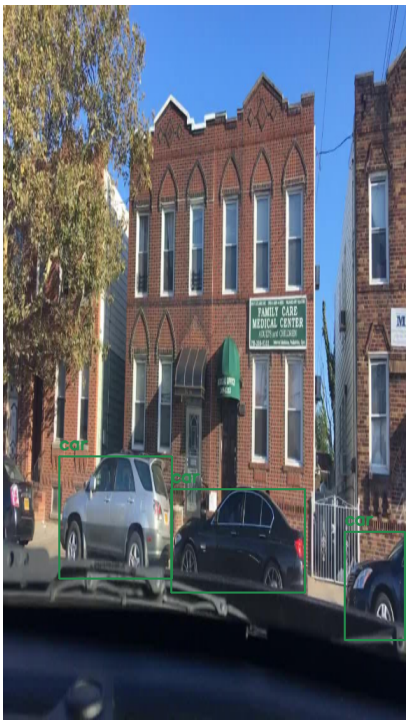

Fig. 1: Image with Object Detections

Now let us see saliency or heat map of above original image where it says which parts is used to detect

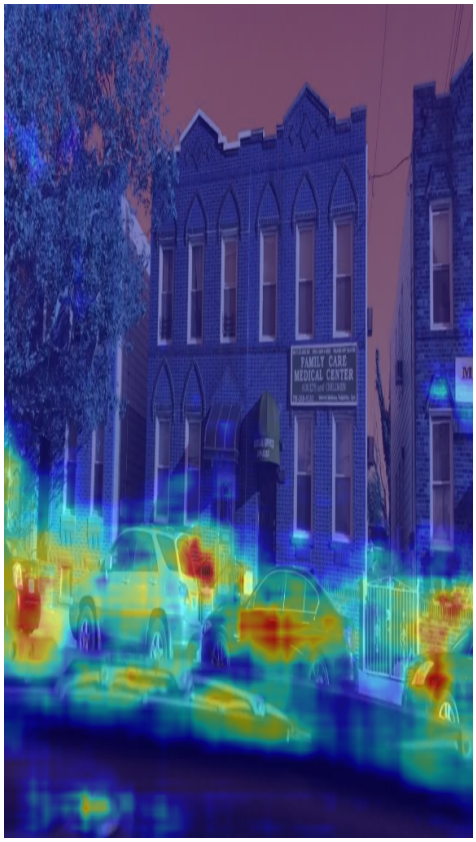

Fig. 2: Heat Map of Above object detection

2) Quantitative Evaluation of CAM Quality: To assess the stability and reliability of the generated CAMs, we computed standard statistical measures across repeated runs and image classes. These include Pearson correlation, Spearman rank correlation, and Mean Squared Error (MSE) between multiple Eigen-CAM maps generated for the same input image.

TABLE I: Quantitative Evaluation of Eigen-CAM Maps

Metric	Value
Pearson Correlation Coefficient	0.482
Spearman Rank Correlation	0.595
Mean Squared Error (MSE)	0.186

Quantitative evaluation using statistical metrics further reinforced the reliability of the generated explanations. The modeerate Pearson (0.482) and relatively high Spearman (0.595) correlation coefficients indicate strong consistency and monotonicity across repeated visualizations. A low Mean Squared Error (0.186) suggests minimal variation in the spatial attention patterns, reflecting the stability of the model's internal feature representations.

3) Insertion and Deletion Metrics: To quantitatively evaluate the faithfulness of the generated Eigen-CAM explanations, we computed the Insertion and Deletion scores. These metrics assess how much the predicted class confidence changes when the most salient regions are added to or removed from the image.

TABLE II: Insertion and Deletion Scores for Eigen-CAM

Metric	Value
Insertion Score	0.66
Deletion Score	0.261

The high insertion score and low deletion score validate that the regions highlighted by Eigen-CAM have a significant causal impact on the model's predictions. These metrics complement the qualitative results, providing a more robust validation of saliency.

B. Grad-CAM for Semantic Segmentation

Grad-CAM was applied to the semantic segmentation model DeepLabV3+ (ResNet-50 backbone) to visualize class-specific saliency maps for pixel-wise predictions. This helps interpret why the model predicts a given label for each pixel region, enhancing model transparency in dense prediction tasks.

1) Qualitative Visualization of Salient Segmentation Areas : A Grad-CAM or class activation mask overlaid for a specific class—likely "road" or "vehicle"—rendered in white, while the rest (black) denotes regions with little to no contribution.

Fig. 3: Mask layer showing areas where the road and vehicle segmentation prediction was there

The activation mask (right) confirms that the model heavily relies on the horizontal bottom regions (road surface) to classify pixels as "road".

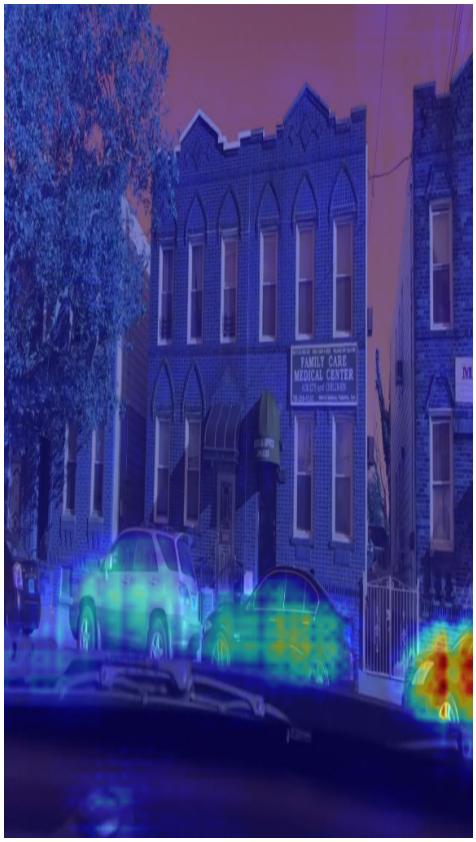

Fig. 4: Heat Map

The boundary with parked vehicles is well-captured, suggesting the model can distinguish between road and adjacent classes.

TABLE III: Quantitative Evaluation of Eigen-CAM Maps

Metric	Value
Pearson Correlation Coefficient	0.273
Spearman Rank Correlation	0.14
Mean Squared Error (MSE)	0.128

2) Quantitative Evaluation of CAM Quality: These results are not that much satisfactory should work on this

Reasons for Low PCC and SROCC:

- **Low Spatial Resolution of CAMs**
  Grad-CAM uses high-level feature maps (e.g., 7×7 or 14×14), which, when upsampled, lead to blurry and imprecise saliency maps.
- **Class Ambiguity or Overlap**
  Visually similar or adjacent classes (e.g., *road* vs *sidewalk*) can confuse Grad-CAM, resulting in class-mixed saliency.
- **Weak Gradient Signal**
  Low-confidence predictions or vanishing gradients produce less informative or noisy CAMs.
- **Scale or Format Mismatch in Evaluation**
  Comparing soft Grad-CAM maps with binary ground truth masks can cause metric misalignment.
- **Background Interference**

If metrics are computed over the entire image (including irrelevant background), correlation scores drop.

- **Coarse Localization of Grad-CAM**
  Grad-CAM may highlight broader regions rather than precise object boundaries needed for segmentation tasks.
- **Model Uncertainty**
  Inconsistent attention by the model across similar images may reduce saliency-map consistency and correlation with expected patterns.
- **Over-smoothing during Upsampling**
  Interpolation techniques may smooth out high-response regions, degrading correlation with sharp masks.

C. GradCAM-MLRCNN

Grad-CAM++: This technique is applied to the ResNet-50 model to generate a heatmap that visually explains which regions of the image contributed most to the prediction. Its output is a saliency map indicating important areas but not precise object boundaries.

Mask R-CNN: This model is responsible for actual object detection. It outputs bounding boxes—and if configured to, segmentation masks—that delineate and localize objects in the image. These outputs provide the object's shape and spatial extent.

Fig. 5: Original image

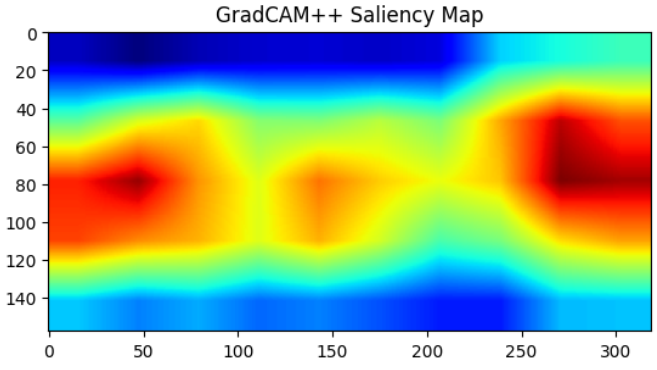

Fig. 6: Heat map for following image

Conclusion

The visualization(Figure 8) clearly demonstrates the evolution from general attention mapping to more precise, object-aware explainability:

1. Grad-CAM++ (Left)

- Applied on a pretrained ResNet-50, the heatmap highlights the most discriminative regions in the full image.
- While it captures areas of interest, the attention is broad and class-agnostic, lacking clear object boundaries.
- Useful for understanding what the network focuses on, but not where the objects are.

2. Bounding Box (Middle)

- Overlays bounding boxes from Mask R-CNN on the Grad-CAM++ heatmap.
- Shows how detected regions align with attention — some boxes align well with high-activation regions.
- However, it lacks shape awareness and includes background noise within boxes.

3. GradCAM-MLRCNN (Right)

- Represents a fused visualization where Grad-CAM is applied over object regions detected by Mask R-CNN.
- This produces object-specific saliency maps, showing not just where the object is, but which parts of the object influence the model's decision.
- Result: Improved localization and interpretability — the contours of objects are more distinct, and background influence is reduced.

By integrating Grad-CAM++ with Mask R-CNN, we bridge the gap between global explainability and object-level understanding. This combination enhances both model interpretability and trustworthiness, especially useful in critical domains like medical imaging, autonomous vehicles, and surveillance.

D. BODEM

TABLE IV: Insertion and Deletion Scores for BODEM for first map

Metric	Value
Insertion Score	0.90
Deletion Score	0.1060
Convergence	0.6667

TABLE V: Insertion and Deletion Scores for BODEM for first map

Metric	Value
Insertion Score	0.90
Deletion Score	0.0500
Convergence	0.6667

REFERENCES

[1] Alphonse Inbaraj Xavier , Charlyn Villavicencio , Julio Jerison Macrohon , Jyh-Horng Jeng and Jer-Guang Hsieh "Object Detection via Gradient-Based Mask R-CNN Using Machine Learning Algorithms" Machines 2022, 10, 340. paper

[2] Guoyang Liu, Jindi Zhang , Antoni B. Chan,Janet H. Hsiao "Human Attention-Guided Explainable Artificial Intelligence for Computer Vision Models" arXiv:2305.03601v1 [cs.CV] 5 May 2023

[3] Milad Moradi, Ke Yan,David Colwell,Matthias Samwald,Rhona Asgari "Model-agnostic explainable artificial intelligence for object detection in image data"Tricentis GmbH, Leonard-Bernstein-Straße 10, 1220 Vienna, Austria

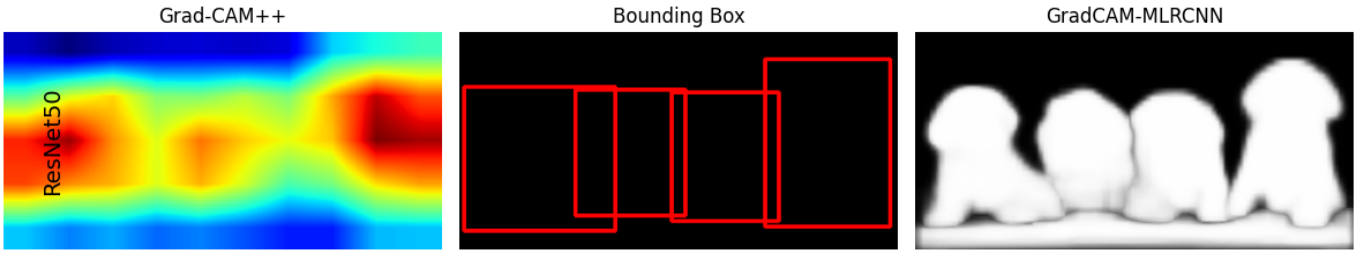

Fig. 7: Grad-CAM visualization for Mask R-CNN showing activated regions responsible for instance detection.

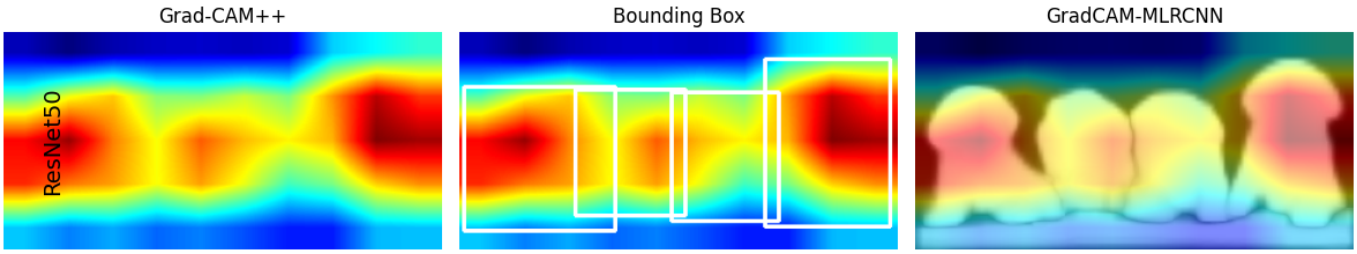

Fig. 8: Grad-CAM visualization for Mask R-CNN showing activated regions responsible for instance detection.

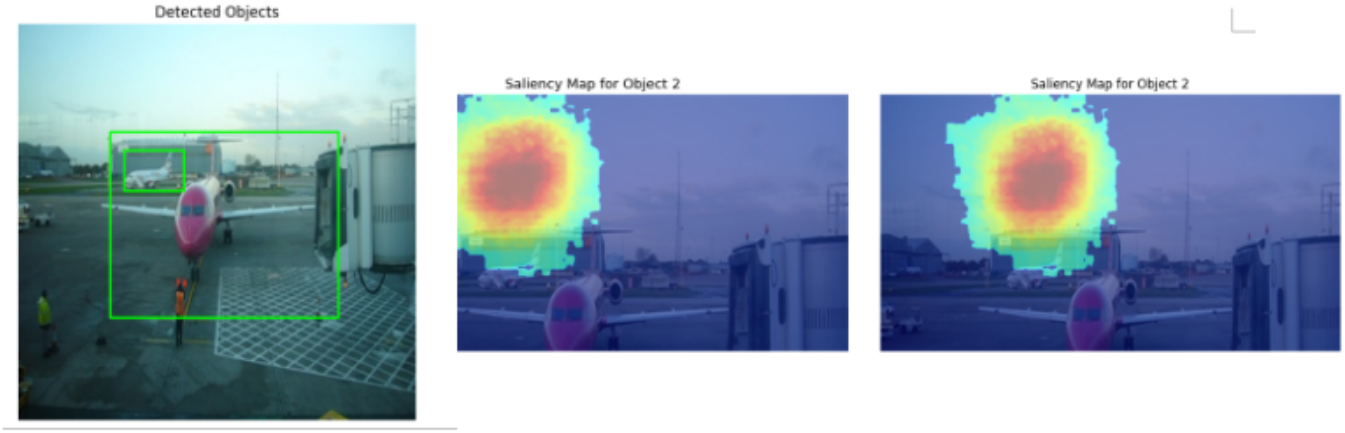

Fig. 9: Saliency maps of two different objects with SSD Detection model using BODEM method

[4] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in Proceedings of the IEEE international conference on computer vision, Conference Proceedings, pp. 618–626.
[5] https://www.sciencedirect.com/org/science/article/pii/S1526149224003084
[6] https://github.com/jacobgil/pytorch-grad-cam?tab=readme-ov-file
[7] https://github.com/GitVirTer/HAG-XAI
[8] https://open-xai.github.io/