Evaluating Model-Agnostic Post-Hoc Visual Explainability Methods for Image Classification Using a ResNet-18 Model Trained on ImageNet-1k

Bachelor Thesis

Bachelor of Science in Applied Computer Science

Pei Shan Keak

March 31, 2026

Supervisor:

1st: Prof. Dr. Christian Ledig
2nd: Sebastian Doerrich, M.Sc.

Chair of Explainable Machine Learning
Faculty of Information Systems and Applied Computer Sciences
Otto-Friedrich-University Bamberg

Abstract

Deep learning models have achieved remarkable performance in various tasks. However, these models often operate as a black box and the reasoning behind the model's decisions is unknown. To ensure these predictions are based on valid evidence, explainability methods are implemented to interpret the model's internal logic and foster trust.

This thesis evaluates visual explainability methods for image classification tasks using a ResNet-18 model trained on ImageNet-1k. We evaluate these methods using qualitative and quantitative approaches. An evaluation framework that integrates budget attention into IoU metrics is introduced to allow for a fair comparison between different output types, such as coarse heatmaps and segmented explanations. The analysis shows that Grad-CAM and LIME successfully focus on correct regions, such as the faces of mammals or the functional parts of artifacts. However, our study also reveals that the model learned shortcuts. Furthermore, Grad-CAM generally performs better than LIME across all metrics. This suggests that model-specific methods leveraging internal gradients are currently more faithful than model-agnostic perturbation approaches.

Overall, our results demonstrate that both quantitative and qualitative evaluation approaches are necessary when investigating the performance of visual explainability methods to ensure that the model's correct predictions are based on the correct focus, rather than on hints or shortcuts.

The code written for this thesis can be found at a GitHub repository. [1]

[1] https://github.com/KeakPS/thesis-bachelor

Acknowledgements

I want to thank Prof. Dr. Christian Ledig for accepting my thesis at the chair. I am deeply grateful and feel very fortunate to have Sebastian Doerrich as my supervisor. His constant support and insightful guidance have meant a lot to me throughout my research.

Contents

List of Figures

List of Tables

List of Acronyms

AI	Artificial Intelligence
ML	Machine learning
DL	Deep learning
CNN	Convolutional neural network
DNN	Deep neural network
Grad-CAM	Gradient-weighted Class Activation Mapping
LIME	Local Interpretable Model-agnostic Explanations
ResNet	Residual Network
ILSVRC	ImageNet Large Scale Visual Recognition Challenge

1 Introduction

In recent years, artificial intelligence (AI) has gained significant attention and has become more popular each year (Xu et al., 2021). Deep learning (DL) is a subset of machine learning (ML) that performs well and even exceeds human capabilities on specific datasets in the areas of computer vision (CV). This is because DL models utilize multiple layers of neural networks, which are able to extract patterns and correlations from complex data quickly and efficiently (Udegbe et al., 2024). However, the challenges of DL models include a lack of interpretability due to their "black-box" nature (Wan et al., 2026). The complexity of these models leads to difficulty in understanding how they make decisions or why they fail (La Rosa et al., 2023).

DL models do not explain their predictions, thus we need explainability methods to interpret them (Molnar, 2025). It is important to understand the decision logic behind the models, especially in the healthcare domain (Akgündoğdu and Çelikbaş, 2025). Explainability methods can generate visual explanations by highlighting the most important features in the input data (Dugăeșescu and Florea, 2025). Thus, these methods can enhance generalizability, transparency and the accurate comprehension of predictions, thereby fostering trust in DL models and their deployment across various domains.

There are various visual explainability methods that can interpret the model's decisions (Gupta et al., 2023). In this study, we focus on exploring different visual explainability methods, including Gradient-weighted Class Activation Mapping (Grad-CAM) (Selvaraju et al., 2017) and Local Interpretable Model-agnostic Explanations (LIME) (Ribeiro et al., 2016), on a ResNet-18 model pretrained on the ImageNet-1k dataset for image classification tasks.

We explore whether these post-hoc visual explainability methods can make the model's decisions easier to understand and reveal misclassification or failure scenarios. This is because predictions given by the model are not always accurate. By using these methods, we can analyze whether the model's predictions are based on the correct focus and reasoning. The evaluations are conducted both qualitatively and quantitatively, including an approach that integrates budget attention using Top-K% important pixels into the Intersection over Union (IoU) metric to ensure a fair evaluation due to the different outputs generated by the Grad-CAM and LIME explainability methods.

2 Related Work

There are many explainability techniques that can interpret the model's decisions and they can be categorized into different classes based on their mechanisms (Gupta et al., 2023). Gradient-weighted Class Activation Mapping (Grad-CAM) (Selvaraju et al., 2017), Integrated Gradients (Sundararajan et al., 2017), Guided Backpropagation (Springenberg et al., 2015) are some of the common gradient-based explainability methods. The Grad-CAM method utilizes the weighted gradients of the

target class with respect to the last convolutional layer and generate a heatmap highlighting the important sections in the image for that predicted class (Selvaraju et al., 2017). The Integrated Gradients technique differs by backpropagating to the input layer of the model and integrating those gradients to provide a pixel-level attribution map (Sundararajan et al., 2017). The Guided Backpropagation uses the same backpropagation process as Integrated Gradients, but adds a filter to prevent the backward flow of negative gradients (Springenberg et al., 2015).

The core mechanism of these gradient-based explainability methods is based on the calculation of the network gradient of the output with respect to the input image, focusing on the features with the highest gradients to explain the individual classification. All these explainability methods provide heatmap visualizations as an explanation (Gupta et al., 2023). The main difference between these gradient-based methods is how they propagate the gradients (Molnar et al., 2020).

Another class of explainability methods consists of perturbation-based methods, including Local Interpretable Model-agnostic Explanations (LIME) (Ribeiro et al., 2016), Randomized Input Sampling for Explanation of Black-box Models (RISE) (Petsiuk et al., 2018), SHapley Additive exPlanations (SHAP) (Lundberg and Lee, 2017). The LIME method generates multiple variants of the original image by using image segmentation technique and employs a surrogate model to approximate the predictions (Ribeiro et al., 2016). The RISE technique performs occlusion by using randomly sampled binary masks to observe changes in the output (Petsiuk et al., 2018). The SHAP method tests the presence or absence of feature combinations in the original image and assigns each feature an importance value (Shapely) for a particular prediction (Lundberg and Lee, 2017).

These perturbation-based methods modify parts of the image by blurring them or replace them with occlusion masks to observe how much this affects the model's decision (Zhang et al., 2025). A notable advantage of perturbation-based methods is that nearly all methods are model-agnostic and can be applied to any architecture, whereas most of the gradient-based methods require access to the internal structure of the model to interpret the decisions (Gupta et al., 2023).

We will analyze two popular and widely used, but very different, local interpretation methods, which are Gradient-weighted Class Activation Mapping (Grad-CAM) (Selvaraju et al., 2017) and Local Interpretable Model-agnostic Explanations (LIME) (Ribeiro et al., 2016). Both are post-hoc visual explainability methods, meaning they are applied after the model has been trained to highlight the important features in the image based on the model's output (Zhang et al., 2025).

Figure 1 shows a brief categorization of explainability methods (Gupta et al., 2023). In our study, we focus on analyzing the model classifications of image data.

There are two main categories of explainability techniques, which are global explanation or local explanation (Henninger and Strobl, 2025). Global explainability methods tend to understand the mechanisms of the model for general decisions, while local explainability methods focus on specific individual predictions. For example, a local explanation shows why the model classifies a specific image as a "cat" instead

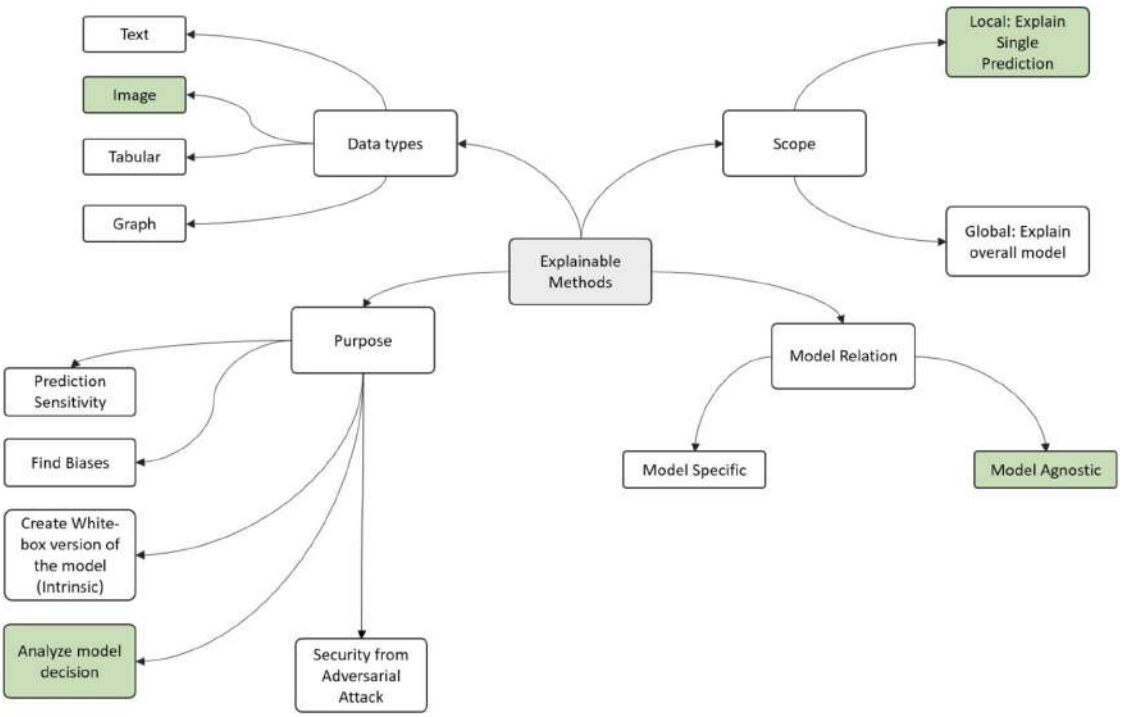


Figure 1: A diagram of high-level categorization of explainability methods (Gupta et al., 2023).

of a "dog", whereas a global explanation tries to interpret the model to show which features generally contributed the most across all predictions. Both Grad-CAM and LIME are categorized as local explainability methods.

Explainability methods can also be classified into model-specific or model-agnostic based on their dependency on the architecture (Gupta et al., 2023). Grad-CAM is an example of model-specific method. It utilizes the feature maps of the final convolutional layer and their corresponding gradients to compute importance weights. Thus, the explanation is dependent on the structure of the model. In contrast, model-agnostic methods, such as LIME, are able to explain any model by treating it as a black box (Ribeiro et al., 2016).

In this study, we selected Grad-CAM and LIME as our visual explainability methods to evaluate and interpret the image classifications output of the deep learning model. Out of those gradient-based methods, we chose Grad-CAM because it performs significantly better than pixel-based gradient visualizations like Guided Backpropagation, especially in localizing objects in images (Gupta et al., 2023). Its heatmap visualization can discriminate between classes more accurately and helps in identifying biases (Selvaraju et al., 2017). This is highly suitable for our evaluation, as it allows us to detect whether the model relies on the background or other noise when making incorrect predictions.

On the other hand, LIME was chosen among those perturbation-based explainability methods because it is popular and widely used. The LIME method is able to show both visual explanations and rank feature importance indices, which are suitable for identifying the most important regions of the input image (Kondaveeti and Simhadri, 2025). Its explanation provides intuitive visualizations of the influential features in segments, unlike SHAP, which provides Shapley values that represent the numerical importance of those features (Henninger and Strobl, 2025). Thus, by using LIME, we can identify specific features like the eyes or ears of a cat, that contribute to the prediction.

The ImageNet dataset was used for this study because it is the gold standard for training and evaluating the performance of deep learning models (Deng et al., 2009). The ImageNet-1k subset, specifically, is a benchmark for various computer vision tasks due to its large scale and diverse labels (Russakovsky et al., 2015). Moreover, the dataset was collected from natural image data, meaning real-world objects commonly encountered in daily life, which makes our analysis approachable to non-experts while effectively demonstrating the performance of visual explainability methods in realistic scenarios.

In this experiment, we utilized the Deep Residual Learning architecture, also widely known as the Residual Network (ResNet), a popular deep convolutional neural network in image recognition tasks (He et al., 2016). This architecture won the first place in the ILSVRC 2015 competition and this excellent achievement is evidence that it performs well in visual tasks. While ResNet has five standard depths (18, 34, 50, 101 and 152 layers), we chose the 18 layers variant in this study. Our objective is to analyze how visual explainability methods interpret the decisions of the model, instead of maximizing classification accuracy at this stage. By understanding the reasons behind the model's decisions, we can eventually refine and improve the performance. Hence, we employed the ResNet-18 architecture and it is more computationally efficient.

Figure 2 illustrates the workflow of the study and evaluates the interpretability of the ResNet-18 model using both qualitative and quantitative measures. Our pipeline starts with data preparation by filtering the ImageNet-1k Russakovsky et al. (2015) validation set and only images with a single object and a single bounding box annotation were selected for the experiments. We used a ResNet-18 model He et al. (2016) pretrained on ImageNet-1k from the PyTorch Image Models `timm` open-source library (Wightman, 2019) for image classification for consistent evaluation. Then, we employed explainability methods to interpret the prediction of the model and identify the features that are considered important by the model. Finally, we performed both qualitative and quantitative evaluations to compare these explanation methods.

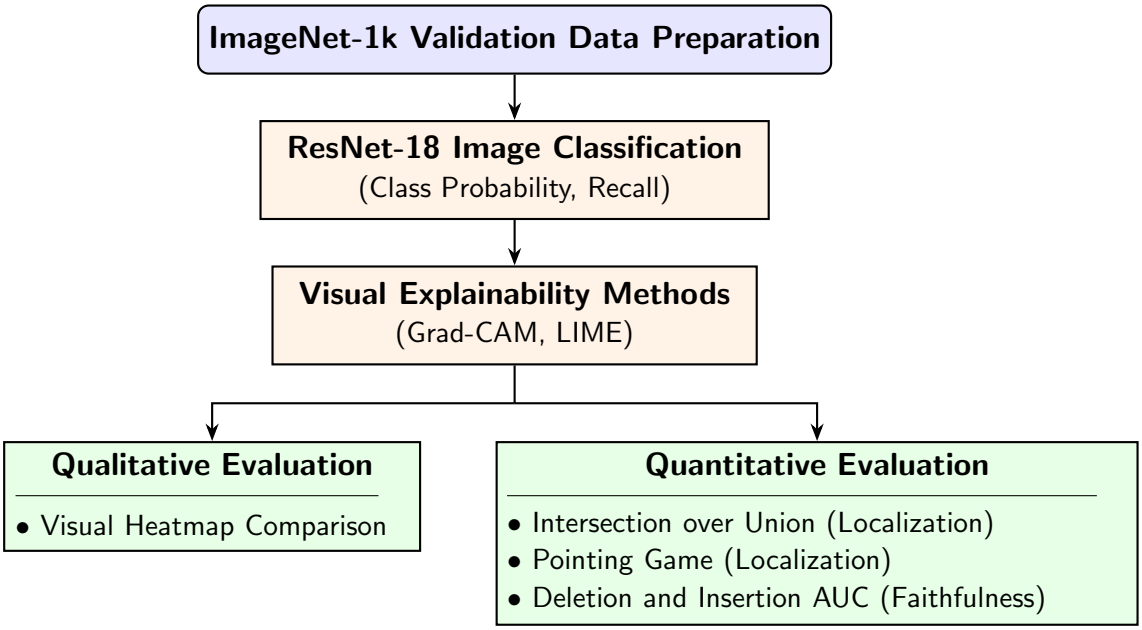


Figure 2: Workflow for evaluating visual explainability methods on ResNet-18 image classification on the ImageNet-1k validation dataset.

3 Visual Explainability Methods

The purpose of employing explainability methods on a complex model is to provide explanations for its predictions in order to build trust in the model (Ribeiro et al., 2016). It is important to ensure the predictions provided are reliable, especially when integrating artificial intelligence (AI) into the healthcare domain (Zhang et al., 2025). If an image is predicted to be "cancerous", the AI model should focus on the lesion areas, meaning the prediction should be based on correct reasoning. Thus, explainability methods are helpful in interpreting the model's decisions and providing human-readable reasoning (Gupta et al., 2023).

One model-specific and one model-agnostic methods were chosen for our study to evaluate their interpretations of a ResNet-18 model He et al. (2016) on image classification tasks using the ImageNet-1k validation dataset (Russakovsky et al., 2015). These methods are Gradient-weighted Class Activation Mapping (Grad-CAM) (Selvaraju et al., 2017) and Local Interpretable Model-agnostic Explanations (LIME) (Ribeiro et al., 2016).

3.1 Grad-CAM

Gradient-weighted Class Activation Mapping (Grad-CAM) was introduced by Selvaraju et al. (2017). Grad-CAM is a post-hoc explainability method that interprets any Convolutional Neural Network (CNN) architecture. This framework aims to address the architectural limitations of a previous study on the Class Activation Map (CAM) procedure proposed by Zhou et al. (2016). Both interpretable approaches are designed for explaining CNN model predictions.

Class Activation Map (CAM) is a foundational approach in visualizing the regions of an image that a CNN model focuses on to make a decision (Zhou et al., 2016). A standard CAM explanation relies on a CNN model having a specific internal structure, which is Global Average Pooling (GAP) (Cao et al., 2023). This GAP layer is placed before the final output layer, which is softmax layer for classification task. This specific structure ensures that each feature map's weight corresponds directly to a particular class (Fayyaz et al., 2025).

Thus, CAM can only be applied to CNN architectures that possess this specific GAP followed by softmax structure. However, Grad-CAM is applicable to any CNN architecture because it is a generalized version of CAM by combining CAM technique along with gradient-based method (Cao et al., 2023). While CAM uses the final layer weights of the model, the Grad-CAM explanation is based on gradients derived from backpropagation (Gupta et al., 2023). Grad-CAM uses the gradients of the target class to calculate importance weights, effectively performing a global average pooling operation on the gradients rather than requiring a GAP layer within the model's structure (Selvaraju et al., 2017). Thus, Grad-CAM is more flexible compared to the CAM technique because it can generate visual activation maps for any CNN architecture without requiring a specific structural component (Gupta et al., 2023).

Grad-CAM is a popular explainability method that can generate visual explanation highlighting the important regions of an image (Devireddy, 2025). Grad-CAM is also class-discriminative, meaning it helps in localizing distinct features. For example, if an image is predicted as a "dog", Grad-CAM method can reveal which features of the dog played a decisive role (Fayyaz et al., 2025). It highlights the regions of the input that most characterize the prediction (Ventura et al., 2024). Thus, the visual explanation from Grad-CAM can perform a direct qualitative visual comparison with other methods.

Figure 3 demonstrates how the Grad-CAM explainability method works in classification (Fayyaz et al., 2025). The implementation of Grad-CAM starts with a selected input image passed through a deep neural network model to get a prediction. Then, the gradients of the target class score are backpropagated to the feature maps of the last convolutional layer (Yoon and Lin, 2025). These gradients indicate the significance of each feature map in the model's decision. Then, global average pooling is applied to obtain one weight for each feature map. Finally, a linearly weighted combination of the maps is passed through a Rectified Linear Unit (ReLU) activation to remove all negative values because only the features with a positive influence

on the prediction are of interest (Molnar, 2025). Grad-CAM generates a heatmap based on these positive weights and overlays it on the original image to highlight the model's attention area (Sun et al., 2022).

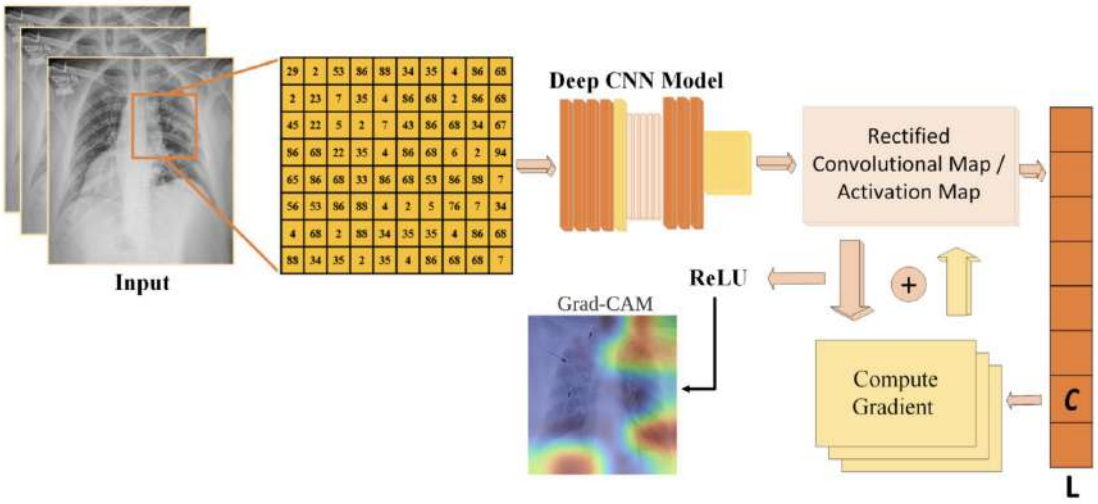


Figure 3: The visualization of internal working process of Grad-CAM (Fayyaz et al., 2025).

Grad-CAM heatmap as explanation on the model's decision, revealing the regions of the image that have the most influence on the model's prediction (Nazir et al., 2024). Figure 4 shows a Grad-CAM visual explanation on the ResNet-18 model prediction of an image as "Goldfish". The Grad-CAM heatmap uses color coded gradients (Nazir et al., 2024). Warmer colors like red indicate areas of high activation in the model's feature maps (Yoon and Lin, 2025). In contrast, cooler colors like blue represent the least important regions (Ventura et al., 2024).

3.1.1 Mathematical Formulation of Grad-CAM

The goal of the Grad-CAM explainability method is to find a localization map $L^c_{\text{Grad-CAM}} \in \mathbb{R}^{u \times v}$ of width u and height v for a specific class c (Selvaraju et al., 2017). First the gradient of the class score y^c before the softmax layer is computed with respect to the feature map activations A^k. Then, these gradients are global-average-pooled to obtain the neuron importance weights α^c_k, which is defined as Equation 1. This weight α^c_k represents the importance of a particular feature map k for a target class c.

$$\alpha_k^c = \overbrace{\frac{1}{Z}\sum_i\sum_j}^{\text{global average pooling}} \underbrace{\frac{\partial y^c}{\partial A_{ij}^k}}_{\text{gradients via backprop}} \tag{1}$$

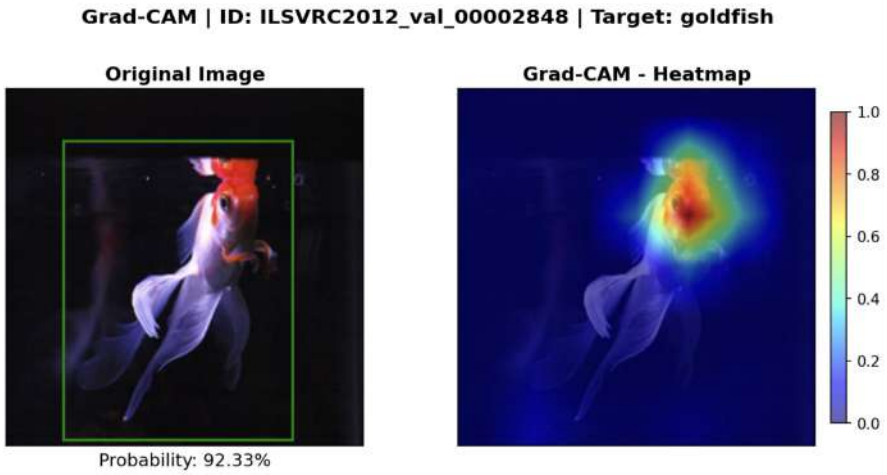


Figure 4: A Grad-CAM visualization showing an explanation of the image in predicting the target class "Goldfish". The visual explanation highlights the regions contributing to the target class, revealing that the head is the decisive feature for the ResNet-18 model prediction.

Where:

- y^c is the score for class c (before the softmax layer).
- A_{ij}^k is the activation value at location (i, j) in feature map k.
- Z is the total number of pixels in the feature map (the normalizing factor).

Once we have the weights α_k^c, we perform a weighted combination of forward activation maps and follow it with a ReLU function to pick positive values in order to produce the final heatmap, which is defined as Equation 2.

$$L^c_{\text{Grad-CAM}} = ReLU \underbrace{\left(\sum_k \alpha_k^c A^k \right)}_{\text{linear combination}} \tag{2}$$

The ReLU is important because we are only interested in the features that have a positive contribution to the class score, meaning an increase in pixel intensity leads to an increase in the class score y^c. Negative weights usually represent other classes. Thus, the ReLU improves localization in Grad-CAM and prevents highlighting classes other than the predicted class.

3.1.2 Workflow of Grad-CAM

In this experiment, we implemented the Grad-CAM explainability method using the `pytorch-grad-cam` library (Selvaraju et al., 2017). The mechanism of Grad-CAM

is based on using the gradients from the last convolutional layer to create saliency maps through linearly aggregated activation maps to highlight the most influential regions (Sun et al., 2022). This process involves one forward pass and one backward pass (Selvaraju et al., 2017). Figure 5 demonstrates the Grad-CAM pipeline of the experiment. The specific aspects of each step are discussed in more detail below.

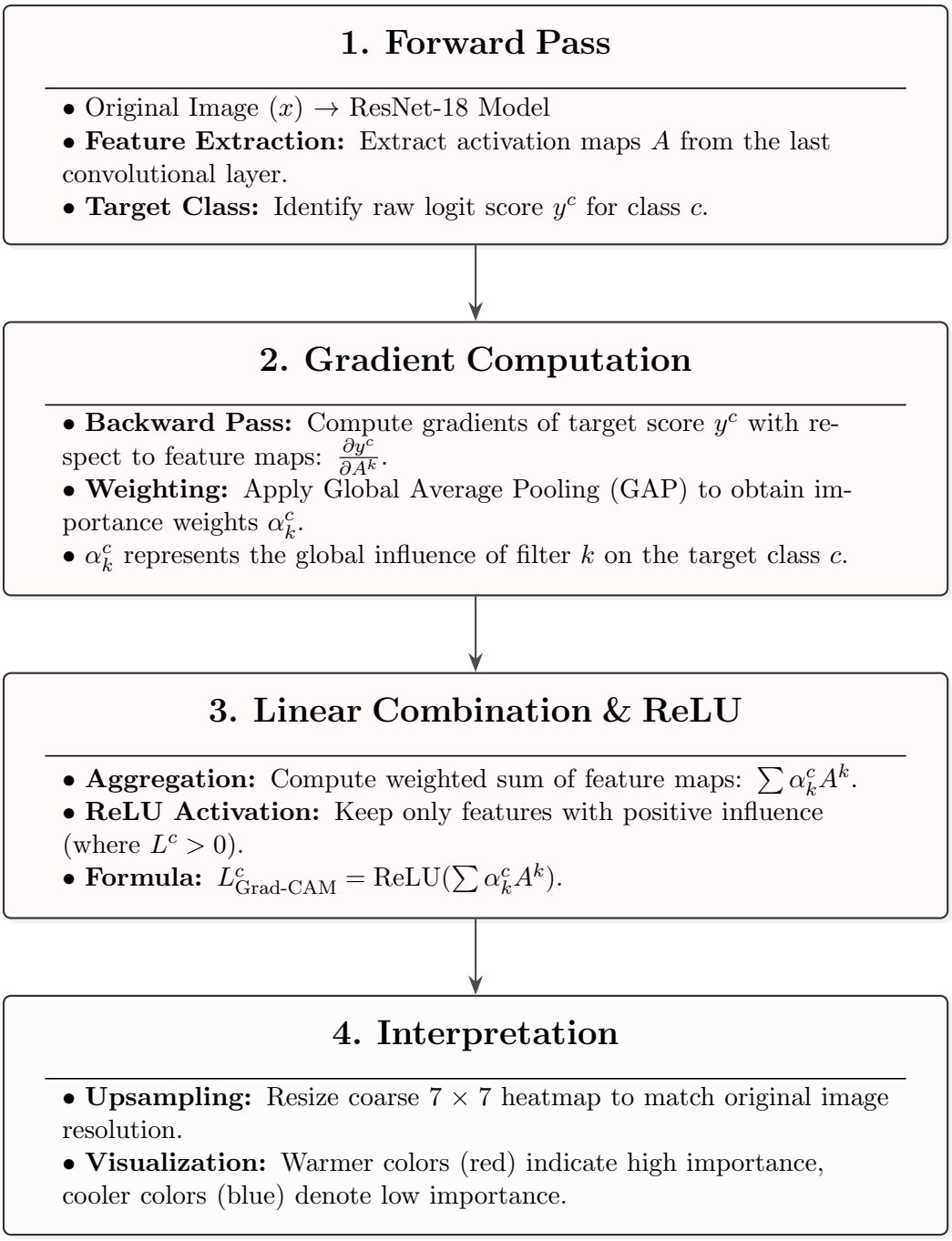


Figure 5: Workflow of the Grad-CAM pipeline for ResNet-18 interpretability on ImageNet-1k.

Step 1: Forward Pass

The implementation of Grad-CAM explainability method starts with selecting a sample image x from the ImageNet-1k validation dataset, which is then passed through the ResNet-18 model to obtain its classification prediction (Selvaraju et al., 2017). Next, the final convolutional layer of the deep neural network is identified. For the ResNet-18 architecture, the target layer is the second convolutional layer (conv2) within the final residual block of the fourth stage (layer4) (Arias-Duart et al., 2022). There are 512 feature maps, also known as activation maps A, with 7×7 resolution in this final convolutional layer of ResNet-18. During the forward pass, these activation maps are extracted.

Grad-CAM provides an explanation based on the final convolutional layer as this layer contains high-level semantics learned by the network (Selvaraju et al., 2017). The spatial information of this layer enables Grad-CAM to localize the regions containing the distinct features of the object in the image, such as a bird's beak, a dog's tail or a cat's eyes.

The raw logit scores of all 1000 classes are obtained after the ResNet-18 model classifies the input image. The class with the highest score is identified as the predicted class, which is also known as the target class c (Selvaraju et al., 2017).

Step 2: Gradient Computation

After the target class c is identified, a backward pass is performed to the last convolutional layer before the fully connected layers in order to compute the gradients of the logit score y^c with respect to the extracted feature maps A^k (Yoon and Lin, 2025). Each feature map k collects different information for classifying the input image.

These gradients are denoted as $\frac{\partial y^c}{\partial A^k}$ and represent the importance of each pixel in the corresponding feature map k toward the final classification of class c (Selvaraju et al., 2017). Thus, these gradients reveal how much each pixel influences the score (Yoon and Lin, 2025). In this experiment, the `ClassifierOutputTarget` object from the `pytorch-grad-cam` library is utilized to handle the gradients, ensuring that the backpropagation process focuses only on the target class while ignoring other class activations (Selvaraju et al., 2017).

Then, the Global Average Pooling (GAP) in Equation 1 is applied to obtain the mean of the gradients of each of the 512 feature maps (Molnar, 2025). Figure 6 illustrates how the GAP technique works. GAP is a dimensionality reduction technique designed to reduce the total number of parameters (Gupta et al., 2023). In this experiment, the GAP operation is applied to reduce the spatial dimensions (7×7) of the gradients into a single value, α_k^c for each feature map k (Yoon and Lin, 2025). This is performed by averaging across the width ($i = 7$) and the height ($j = 7$) dimensions, where the total number of pixels in each feature map is $Z = 49$. Thus, the importance of each feature map is summarized into one weight, which is known as the neuron importance weight α_k^c.

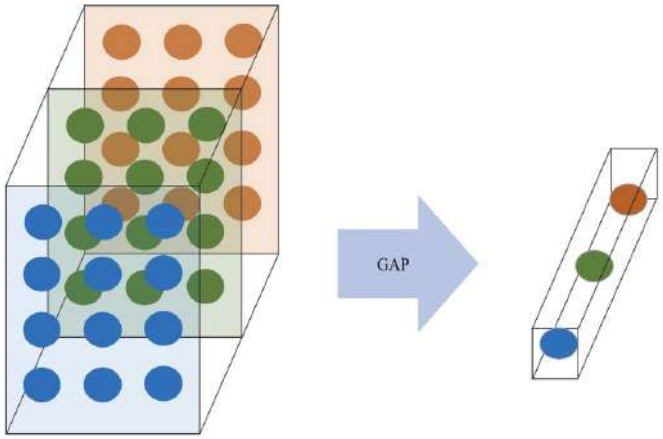


Figure 6: Global Average Pooling (GAP) reduces the dimension by taking the average of each feature map, meaning only one value represents each feature map. Thus, the dimension of each feature map is reduced from 49 to 1 (Gupta et al., 2023).

Step 3: Linear Combination & ReLU

Grad-CAM generates a heatmap by utilizing a weighted linear combination of activation maps (Sun et al., 2022). Each feature map A^k is multiplied by its corresponding neuron importance weight α_k^c (Yoon and Lin, 2025). Thus, the highly contributed feature maps for the target class are identified. These 512 weighted feature maps are then aggregated to form a single combined map (Fayyaz et al., 2025).

Next, this combined map is passed through a Rectified Linear Unit (ReLU) activation function in Equation 2. The ReLU function sets all negative value to zeros (Fayyaz et al., 2025). This is because Grad-CAM explanation is only interest in regions that support the target class, which have a positive influence on the target class score (Molnar, 2025). Usually the negative values indicate features of other classes. Thus, the Grad-CAM heatmap visualizes and explains the evidence for the predicted class c.

This result is stored as a grayscale saliency map in the Grad-CAM implementation, where the pixel values range from 0 to 1, representing the normalized importance of each region 6 (Selvaraju et al., 2017).

Step 4: Interpretation

The output of Grad-CAM (7×7) has smaller dimensions than the original input image (224×224) as it was derived from the last convolutional layer of the ResNet-18 model (Yoon and Lin, 2025). Thus, an upsampling process is carried out using bilinear interpolation in order to match the input size. Then, the heatmap can be overlaid on the original image.

The heatmap is then normalized. Thus, a value of 0 indicates that a pixel has a minimum importance and is assigned with a cooler color, such as blue, while a value of 1 indicates maximum importance and is assigned with a warmer color, such as red (Nazir et al., 2024). Figure 7 shows an example of the Grad-CAM heatmap.

Thus, Grad-CAM explainability method can provide class-discriminative heatmap and show which image regions influenced the prediction (Devireddy, 2025).

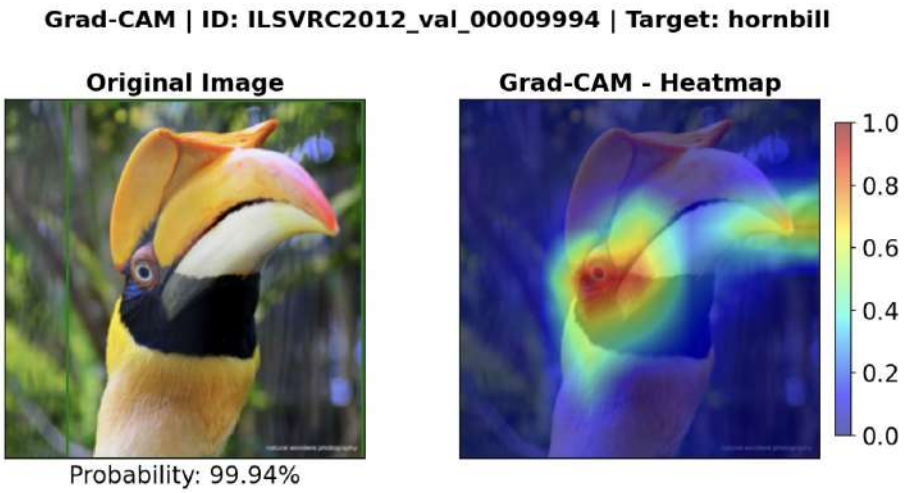


Figure 7: A Grad-CAM heatmap visualization is shown on the right with its original input image on the left. The ResNet-18 shows high confidence in this classification. Warmer-colored regions, such as red, indicate the areas focused on by the model, representing the important features that contributed to the prediction. Thus, based on the heatmap, we can observe that the eyes and the pointy part of the beak are evidence that the model predicted this image as a "Hornbill".

3.2 LIME

Local Interpretable Model-agnostic Explanations (LIME) was introduced by Ribeiro et al. (2016). LIME is a post-hoc explainable technique that aims to interpret predictions from any model. The core mechanism is based on approximating the complex "black-box" model behavior using a simpler, interpretable local surrogate model. The authors of the LIME approach believe that a simple model is sufficient to explain predictions from a complex model (Gupta et al., 2023).

Figure 8 illustrates the LIME interpretation process. The concept behind the LIME explainability method is intuitive (Molnar, 2025). LIME tracks the changes in predictions of the complex model by giving some variations of the original input. For images, these perturbed inputs are created by segmenting the original image into superpixels and randomly hiding them. Then, LIME uses these resulting probabilities on the target class from the complex model, mapped to their corresponding perturbed image, to train a simpler surrogate model that explains the local decision. LIME explanation highlights the segments that support the target class in green and the segments that decrease the probability of the target class in red (Ribeiro et al., 2016).

For example shows in Figure 9, when a perturbed image with the segment containing the dog's ear being hidden, the probability of the target class "German Shepherd"

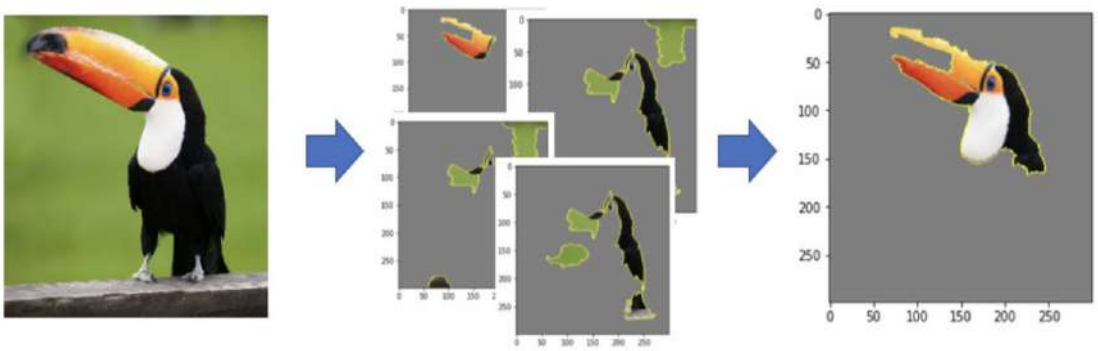


Figure 8: The visualization of LIME interpretation process from left to right, showing the original input followed by the perturbation process used to obtain variations of the input. The final image displays the LIME explanation highlighting the top-n segments contributing to the target class "toucan" (Ng et al., 2022).

drops. Conversely, if the snow background segment is being hidden, the probability of the target class "German Shepherd" increases. This result shows that the feature of the dog's ear supports the prediction of "German Shepherd", but the snow decreases the confidence of the model in predicting that class. Thus, LIME visualization highlights the segment containing the dog's ear in green, showing that it has a positive weight, while highlighting the segment with snow in red, indicating a negative weight toward the prediction.

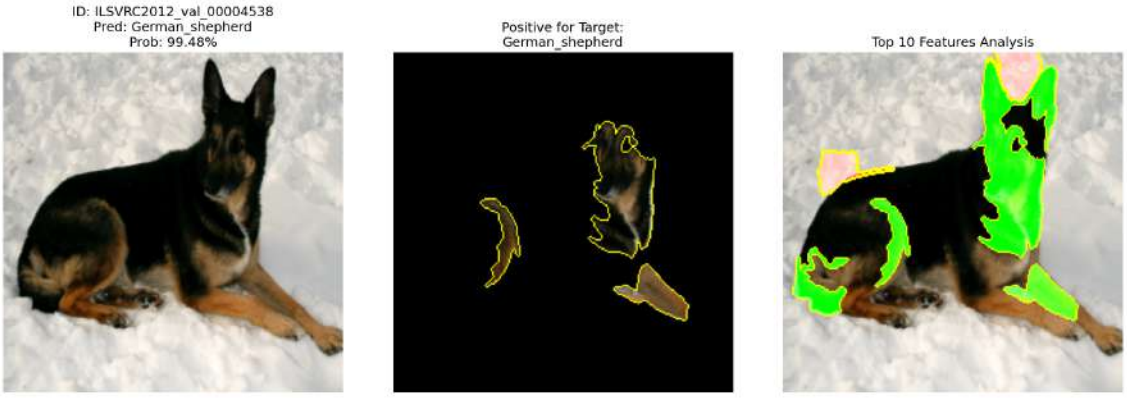


Figure 9: A LIME visualization showing an explanation of the image in predicting the target class "German Shepherd". The image on the left shows the original input. The image in the middle shows the top 5 segments that contributed positively to the target class. The image on the right shows the top 10 segments with the highest absolute weight. The segment in green represents a segment that supports the target class "German Shepherd" and the segment in red means it confused the model when predicting for that target class.

The LIME method is an interesting explainability technique within the social science domain (Henninger and Strobl, 2025). For example, LIME can explain why certain treatments are more effective for specific patients or why certain teaching methods are more suitable for specific students. These scenarios require selecting a specific person of interest for the explanation.

The LIME method gives a local explanation, meaning it focuses on interpreting an individual prediction Garreau and Luxburg. Instead of explaining the global reasoning, which is a significant challenge when the gradient is near to zero, the LIME method aims to solve feasible tasks by using a surrogate model to approximate the original model locally (Ribeiro et al., 2016). The LIME evaluation framework supports any architectures, including explanation model, also known as "white-box" models, such as Decision Trees and Linear Regression. In many cases, LIME is mainly utilized to interpret "black-box" models, such as deep learning models. Thus, LIME is considered as a model-agnostic approach that can explain the predictions from any classifier or regressor in a faithful way. Moreover, most of the model-agnostic explainability techniques are also perturbation-based, meaning they make predictions on modified input features (Molnar et al., 2024).

3.2.1 Mathematical Formulation of LIME

LIME is expressed as Equation 3 and its specific aspects in more detail further below (Ribeiro et al., 2016). The goal of LIME is to find an explanation model $g \in G$, where G is a set of potentially interpretable models, including all linear regression models, decision tree or sparse linear models (Lasso), that is able to provide intuitive presentations, such as visual or textual artifacts (Molnar, 2025).

$$\xi(x) = \text{argmin}_{g \in G} \mathcal{L}(f, g, \pi_x) + \Omega(g) \tag{3}$$

Where:

- f represents the complex, "black-box" model being explained.
- g is the simple, interpretable surrogate model.
- $\pi_x(z)$ is a proximity measure that defines the size of the neighborhood around the original input x.
- $\mathcal{L}(f, g, \pi_x)$ is a measure of how unfaithful g is in approximating f within the locality defined by π_x.
- $\Omega(g)$ represents the complexity of the explanation model g.

LIME uses a local interpretable model g for instance x that minimize loss $\mathcal{L}$, which measures how close the explanation is to the prediction of the original model f (Ribeiro et al., 2016). $\Omega(g)$ should be kept low, meaning having less features in the

explanation, in order to be interpretable by humans. The proximity measure π_x assigns higher weights to perturbed samples that are closer to x (Devireddy, 2025). LIME needs to balance between interpretability and local fidelity (Ribeiro et al., 2016).

In practice, LIME can only optimize the loss $\mathcal{L}$ under a fixed complexity $\Omega(g)$ (Molnar, 2025). The user is required to define the number of active features for the explanation. For example, in the image context, this is the maximum number of segments that the surrogate model may use in the explanation.

3.2.2 Workflow of LIME

In this experiment, we implemented the LIME explainability method using the `lime` library (Ribeiro et al., 2016). The idea behind the LIME is to build a simpler and more explainable surrogate model, such as a linear function, trained on perturbed variations of the original input (Gupta et al., 2023). Then, LIME identifies the specific feature weights within the perturbed neighborhood that quantify the contribution of each feature to that particular prediction by analyzing these local samples. The details of each step in the process are discussed below and Figure 10 illustrates the LIME workflow used in the experiment.

Step 1: Image Segmentation

First, an original input x is selected from the ImageNet-1k validation dataset. The approach of randomly perturbing individual pixels would not have a noticeable change in model's confidence in the target class, since usually more than one pixel contributes to the prediction (Molnar, 2025). Thus, a segmentation algorithm is applied in order to group pixels into superpixels based on their similarities to create more meaningful interpretable components (Gupta et al., 2023).

In our experiment, we chose Simple Linear Iterative Clustering (SLIC), which is widely adopted in the LIME method for image data (Achanta et al., 2012). This superpixel algorithm adapts the k-means clustering technique with consideration of the three color channels plus two spatial coordinates to generate perceptually homogeneous regions. Thus, by including joint color and spatial features, the SLIC algorithm divides an image into superpixels that are not only the same color but are also physically close to each other.

The authors conducted some tests on three different approaches in generating superpixels, including the SLIC, Felzenszwalb and Q-Shift, found that the SLIC algorithm is more stable in segment ordering (Ng et al., 2022). Stability in the context of LIME means the explanation output is consistent when running LIME multiple times on the same image and the SLIC algorithm is better at ensuring the same segment index consistently represents the same image region and is approximately equally sized.

The SLIC algorithm is simple because it only has one parameter to tune, which is the number of desired segments (Achanta et al., 2012). This parameter determines the granularity of the explanation in LIME, thereby determining its resolution. Too

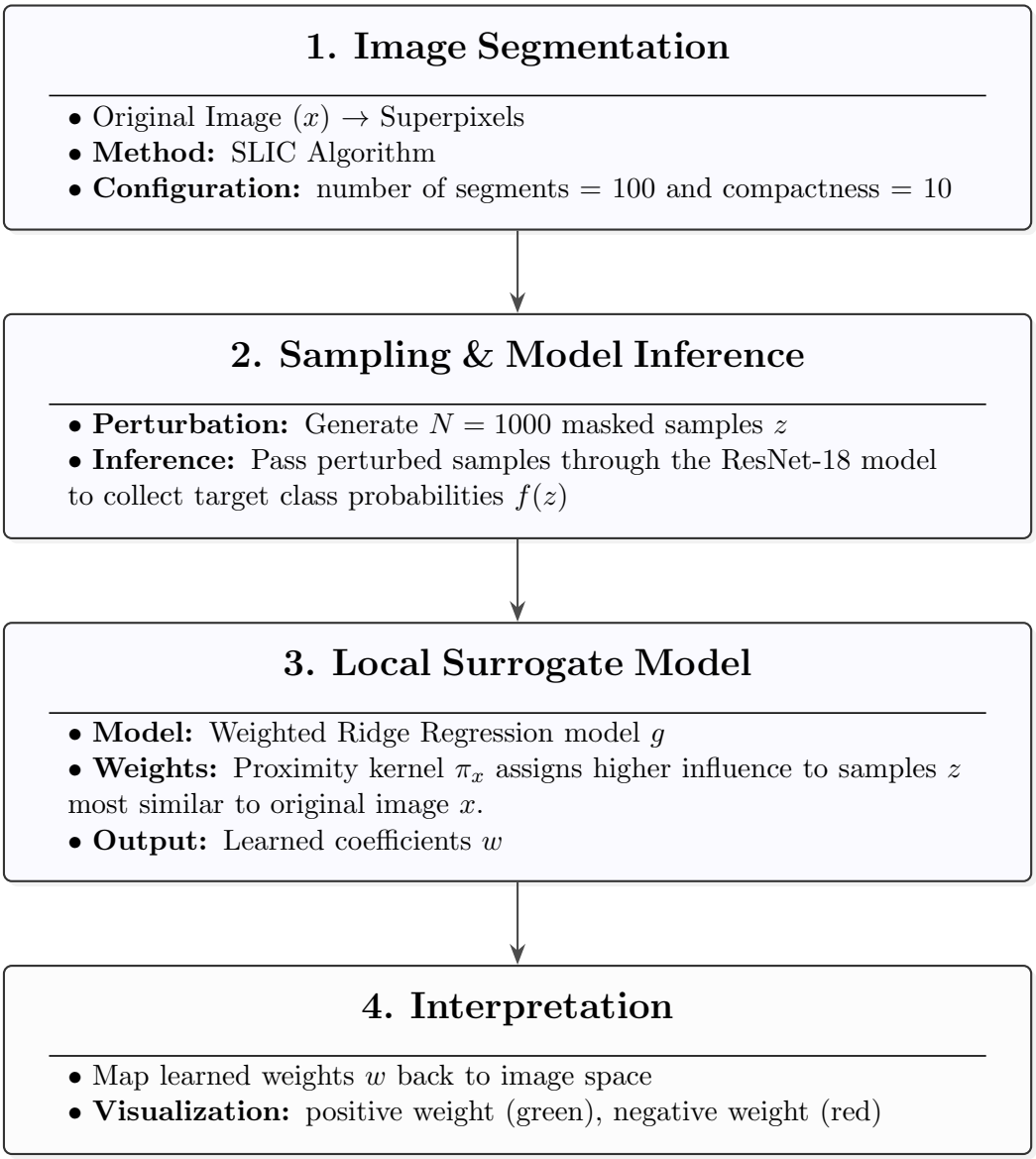


Figure 10: Workflow of the LIME pipeline for ResNet-18 interpretability on ImageNet-1k.

many segments will cause "over-segmentation". Each segment becomes so small that it may fail to capture meaningful textures and patterns, instead focusing on tiny details that make the overall pattern difficult to identify. Conversely, a low number of segments leads to "under-segmentation", where object features and background may exist within one segment or distinct features may be grouped together. After testing across a range of values of segment, including 30, 50, 80, 100, 150 and 300, we observed that the range between 50 and 150 provides an optimal balance for LIME visualizations. Figures 11 through 13 illustrate the impact of varying segment counts on a sample image using LIME explanation. Consequently, we fixed the parameter

at 100 segments for this experiment as it consistently yielded visual components with distinct, interpretable features.

Another parameter is the compactness. We used a compactness value of 10, which is the default in the `scikit-image` implementation and was kept constant throughout our experiments. Superpixels in uniform and regular shape are more intuitive to analyze but may not align with the actual boundaries in the image well (Achanta et al., 2012). We chose a compactness value that balances clean shapes with accurate boundary adherence.

Figure 11: A LIME explanation with 30 segments shows that some superpixels in the middle merge the features of the dog with the road background. Consequently, the entire segment is highlighted in green, even though the road may not be relevant to the "Golden Retriever" target class.

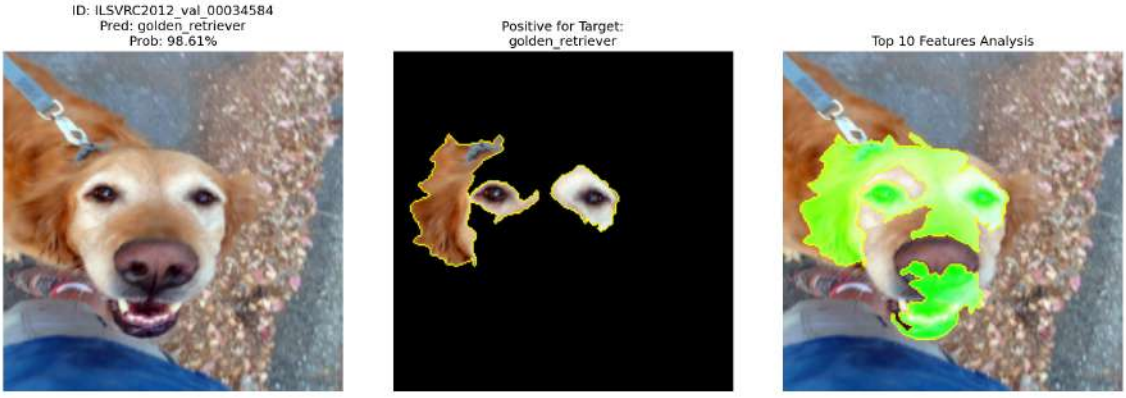


Figure 12: A LIME explanation with 100 segments provides a balanced representation as it effectively separates the features of the dog from the background. Moreover, the dog's distinct features are more identifiable. For example, the ears, eyes and mouth are grouped into different segments.

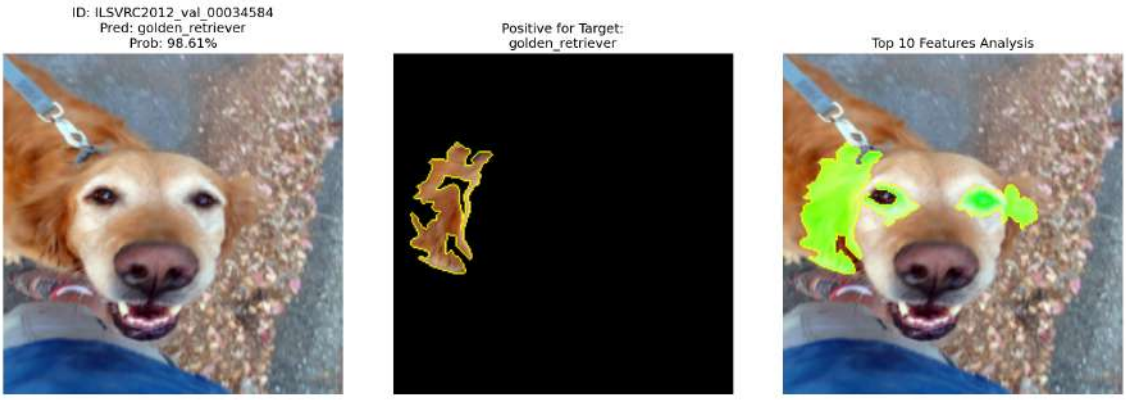


Figure 13: A LIME explanation with 200 segments shows that the size of each segment is too small and may not represent a meaningful, interpretable component. When observing the top 5 positive segments, it is difficult to relate them to the "Golden Retriever" target class.

Step 2: Sampling & Model Inference

After the chosen original image is segmented using the SLIC superpixel algorithm, LIME generates multiple variations of the input image z by perturbing the segments (Devireddy, 2025). In this experiment, we fixed the number of perturbed samples N at 1000 for each input image. This value is chosen because it can produce a stable LIME explanation output, while maintaining a manageable computational cost. A higher number of perturbed variations ensures better stability in the LIME framework, meaning more consistent LIME explanation outputs across multiple runs (Ng et al., 2022).

This perturbation process is used to create a local explanation (Ng et al., 2022). A binary feature vector is created and each element indicates the presence (1) or absence of a specific segment (Lundberg and Lee, 2017). During the perturbation process, the segments are randomly turned "on" or "off" (Molnar, 2025). An "off" segment is replaced by the average color of that specific segment by default in the `lime` library. Thus, the details of that segment are blurred with a uniform color.

Then, each perturbed sample is passed through the complex model, which is ResNet-18 model, to track precisely how the target class probability $f(z)$ changes with different visible regions (Zhang et al., 2025). This allows the LIME to establish a clear relationship between the presence of a superpixel and the resulting classification confidence (Ribeiro et al., 2016). Thus, LIME can quantify the individual contribution of each visual component to the model's final decision, effectively treating the "black-box" ResNet-18 model as a function whose local behavior can be probed through these controlled perturbations.

Step 3: Local Surrogate Model

Then, the perturbed samples with their corresponding target class probability are used to train an interpretable local surrogate model g (Ribeiro et al., 2016). The local surrogate models can provide meaningful and transparent explanations through decision rules or weights, which explain the reasons for the decision (Gupta et al., 2023). The model is simpler and interpretable and can be anything from Lasso to a Decision Tree (Molnar, 2025).

In our experiment, we implemented the default local model from the `lime` library, which is a Weighted Ridge Regression (Ribeiro et al., 2016). This linear model learns feature importance weights by using a proximity kernel π_x to assign higher importance to perturbed samples z that are most similar to original image x (Devireddy, 2025). For example, the segments that are visible when the target probability is high obtain a higher importance score. Thus, the LIME explanation is locally faithful, meaning the interpretation is specific to this individual input (Ribeiro et al., 2016). The learned coefficients w represents the relative importance of each superpixel in contributing the ResNet-18 model's final prediction.

Step 4: Interpretation

The learned coefficients w are mapped back to their corresponding segments to generate a visual explanation (Ribeiro et al., 2016). In this experiment, we have three different presentation formats for LIME visual explanation, including (1) Top 5 positive segments of the target class, showing the most supporting features (2) Top 10 segments to provide a broader view of both positive and negative contributions and (3) LIME heatmap, to provide an intuitive visualization that allows for a direct qualitative comparison with the Grad-CAM heatmap.

These segments are highlighted to show the specific regions that contributed to the model's prediction based on the learned coefficients (Devireddy, 2025). For the (1) and (2) visual presentation, the segments highlighted in green indicate features that support and increase the target class probability, the segments in red represents evidence that decrease the model's confidence in predicting that target class (Molnar, 2025).

In the (3) LIME heatmap, the segments are highlighted according to their influence on the target class using color coded gradients (Kondaveeti and Simhadri, 2025). Each superpixel is assigned with a color based on its significance, with warmer colors such as red indicating higher importance, while cooler colors such as blue show segments with lower importance. These color coded gradients follow the same rules as the Grad-CAM heatmap.

Figure 14 illustrates a LIME visual explanation of an example which has the target class "Siberian Husky". The LIME explanation interprets the regions on which the complex model focused and can be visualized directly on the input sample, providing an intuitive reasoning for the classification decision (Molnar, 2025).

Figure 14: A LIME visualization in three different representations is shown. The left image of the top row shows the original input image, while the image on the right shows (3) the LIME explanation as a heatmap. Here, warmer-colored segments, such as red, indicate the important features that contributed to the target class "Siberian Husky". The bottom row images have a different color representation. The image on the left illustrates (1) the top 5 positive segments, meaning evidence supporting the target class. The image on the right shows (2) the top 10 segments, showing both positive and negative influences on the ResNet-18 model prediction. Segments highlighted in green indicate positive weights that increase the probability of the target class, while segments in red decrease it. Thus, by looking at the positive segments, we can observe that the eyes and nose are evidence that the model predicted this image as a "Siberian Husky".

4 Evaluation Metrics

Different visual explainability methods produce different outputs. LIME results in superpixels and each has one value (Ribeiro et al., 2016). Every pixel in that superpixel gets the same weight. If there are 100 segments, there are only 100 output values for that image. In contrast, Grad-CAM provides continuous output values and may have different values for every pixel that create a smooth heatmap (Sel-

varaju et al., 2017). Hence, evaluation metrics are limited for a fair comparison across the explainability methods. We also included accuracy and probability metrics to evaluate the performance of the pretrained ResNet-18 model on ImageNet-1k image classification task.

We first performed a small scale evaluation using selected classes and reviewed them manually. Then, we expanded this to a larger subset, by evaluating all the images of that particular class to avoid cherry-picking results. Qualitative and quantitative measures are the two main types of evaluation (Gupta et al., 2023). We conduct a comprehensive analysis of the explainability methods by evaluating them both qualitatively and quantitatively to understand their interpretability on the predictions of the ResNet-18 He et al. (2016) model. We selected specific approaches and metrics that are suitable for our explainability methods.

4.1 Qualitative Evaluation

Qualitative evaluation is carried out based on observations and comparisons between the visual heatmap outputs from the explainability methods. The authors found that convolutional neural network models tend to be biased towards texture rather than the shapes (Geirhos et al., 2019). For example, the models may classify an image as a "cat" based on fur texture rather than the cat shape. Conversely, humans observe and recognize the entire object shape as a clue to identify the object. In our experiment, we utilized explainability methods to analyze where the model focuses the most when making that particular prediction and compare whether it aligns with human intuition.

We compared these explainability methods to find which features the model considered important when recognizing objects. For example, whether it relied on textures, patterns or fine-grained details, like fur, feathers or surface textures. In some cases, we observed specific scenarios or circumstances where the model used the image background as a hint to make a correct or incorrect prediction. In cases of misclassification, we used these insights to determine if the model focused on the background or on fine-grained features that are shared by very similar classes, thus leading to the mistake. Then, we categorized and highlighted the qualitative differences between explainability techniques.

4.2 Quantitative Evaluation

Quantitative evaluation are based measurable and numerical aspects to evaluate a model's performance (Kondaveeti and Simhadri, 2025). A quantitative methodology for evaluating deep learning models using explainability methods goes beyond traditional accuracy metrics (Ventura et al., 2024). Evaluation from many angles is important and serves as a foundation for integrating reliable and trustworthy artificial intelligence (AI) systems into real-world practice.

However, comparing explainability methods solely based on qualitative analysis is limited by human subjectivity as interpretations may vary between individuals, which leads to different conclusions (Kondaveeti and Simhadri, 2025). Most of the studies depend on subjective visual analysis of heatmaps produced by explainability methods, which is inconsistent, non-reproducible and difficult to scale.

We addressed these limitations by employing a generalizable and fair quantitative evaluation methodology, using metrics such as Intersection over Union (IoU) with pixel budget integration, the Pointing Game and faithfulness metrics, specifically Deletion and Insertion Area Under the Curve (AUC).

4.2.1 Intersection Over Union with Top-K% pixels

Intersection over Union (IoU) is a localization-based metric that primarily used in computer vision to evaluate how well a predicted bounding box aligns with the ground-truth bounding box. In our study, we formally define the IoU with the integration of the pixel budget concept as a metric to measure the performance of explainability methods in image classification. While the IoU used differs from the traditional IoU, the concept and mathematical foundation remain the same. The standard IoU formula in Equation 4 is the area of the intersection of the two regions divided by the area of the union (Rainio et al., 2024). A is defined as the ground-truth region and B is denoted as the prediction. The area of the intersection is the area of overlap between A and B. The area of the union is the total area of A and B.

$$IoU = \frac{|A \cap B|}{|A \cup B|} \tag{4}$$

The IoU score ranges between 0 and 1, where 0 indicates no similarity and 1 shows a perfect match between the predicted and ground-truth bounding boxes. The greater the overlap region, the greater the IoU score. A higher IoU means the explainability method highlights the correct spatial region more accurately.

We computed the mean Intersection over Union (mIoU) using Equation 5 when evaluating across multiple samples or multiple classes (Zhang et al., 2025). The mIoU compares the prediction binary map B with the ground-truth annotation A.

$$\text{mIoU}(E, A) = \frac{1}{N} \sum_{i=1}^{N} \frac{|B_i \cap A_i|}{|B_i \cup A_i|} \tag{5}$$

In Equation 5, E is the generated heatmap and converted into a binary prediction map B. N is defined as the total number of samples. B_i and A_i denote the prediction and ground-truth annotation for the i-th sample.

A direct comparison of standard IoU scores between the different explainability methods might not be possible or fair because each visual explainability method has a different output scale. Grad-CAM produces smooth heatmaps, which is pixel-levels

attention (Selvaraju et al., 2017). LIME output image is segmented into a desired number of segments (superpixels) and assigns one weight to an entire superpixel (Ribeiro et al., 2016). Every pixel in that superpixel gets the same weight. If we want to compare these methods with IoU, by setting a threshold of 0.5 on Grad-CAM and selecting top 1 segment on LIME. The IoU scores are not comparable as the masks are in different sizes. Grad-CAM might have 25% of pixels above the threshold, while LIME might only have 8% of pixels in the top 1 segment. The explainability method with a larger mask has a better chance of overlapping with the ground truth annotation and results in a higher IoU score. However, the explainability method with a smaller mask might be more precise but will be penalized.

We propose an approach on top of the standard IoU to mitigate this issue by giving each explainability method the same "attention budget". Instead of letting each explainability method use its own threshold for the standard IoU calculation, we select the top K% most important pixels. By fixing this percentage, we determine which objects the methods can localize within the same "attention budget", ensuring a fairer comparison. The framework of integrating the top-K% pixels concept into Intersection over Union (IoU) works as follows (Figure 15).

As illustrated in Figure 15, the process begins with the attention budgeting stage followed by two parallel IoU calculation options.

Step 1: Normalization of the Heatmap

First, we filter negative values and normalize all heatmaps to a 0 to 1 scale using the min-max algorithm in Equation 6. The step of removing negative weights is important as we do not want to include weights that negatively contribute to the prediction. A pixel is negatively weighted by an explainability method, if it decreases the model's predicted probability for that class.

Then, the filtered data is rescaled to a range between 0 and 1 according to the min-max algorithm in Equation 6, where max(X) and min(X) are the maximum and minimum values of the original input X (Lima and Souza, 2023). This normalization step makes Grad-CAM's gradient magnitudes directly comparable to LIME's superpixel weights.

$$X_{norm} = \frac{X - \min(X)}{\max(X) - \min(X)} \tag{6}$$

Step 2: Top-K% Pixel Selection

We assign the same attention budget to every explainability method by selecting only the top-K% of pixels (Mastroianni and Sager-Müller, 2024). If $K = 20$, each method must highlight exactly the top 20% most important pixels of the image area. We conducted experiments using $K = 20$ as the baseline for our evaluation because it represents a reasonable spatial occupancy for target objects in the dataset. After testing across multiple thresholds with a small subset, we found that selecting the top 20% of pixels provides a consistent and representative measure of performance.

However, LIME assigns same importance weights to all pixels within a single superpixel. This means there is a possibility that the number of top 20% pixels might

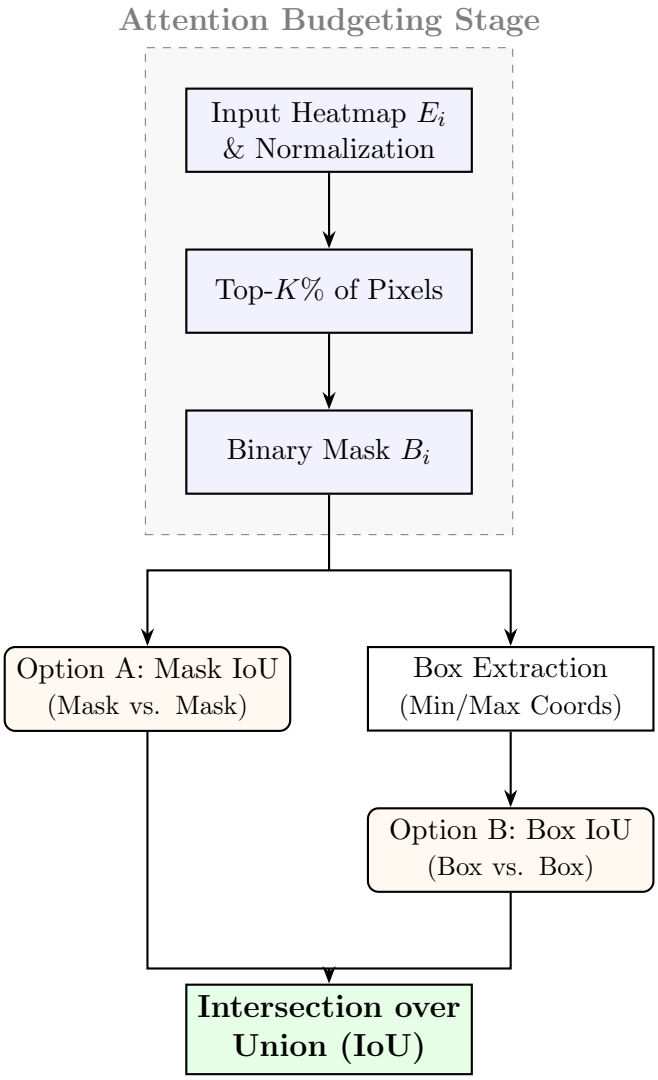


Figure 15: A framework for attention budgeting using Top-K% pixels and IoU calculation.

exceed the threshold of 10,035 pixels (which is 224×224×20%) because multiple pixels may have the same importance weight at the decision boundary.

Thus, we employ argsort function from NumPy library to rank all pixels based on their weight and stop selection exactly when the 10,035th pixel is reached to break ties. This ensures all explainability methods have the same budget of activation area.

Step 3: Binary Mask

Then, we convert heatmaps to binary masks to enforce the top-K% budget. Equation 7 shows how the binary prediction map B_i is derived from the normalized heatmap E_i by selecting the pixels with the highest importance weights. The total number

of activated pixels equals a fixed percentage K of the total image area for a sample i.

$$B_i(x, y) = \begin{cases} 1 & \text{if } E_i(x, y) \geq \tau_K \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

In Equation 7, τ_K is the threshold that selects the top-K% of values in E_i. The tie-breaking rule from Step 2 is applied to ensure B_i contains exactly K% of the total pixels.

Step 4A: Pixel-level Intersection over Union (Mask IoU)

We evaluate the performance of explainability methods using a pixel-wise comparison between the binary prediction mask B_i and the ground-truth mask A_i. The reason of using pixel-level analysis is to measure the direct overlap between the activated pixels and the ground-truth box to see how precisely the object area is captured.

We implement Boolean logic by counting the pixels active in both masks as the intersection, while the union is the count of pixels active in either. Equation 8 shows how Mask Intersection over Union (Mask IoU) is calculated by dividing the shared pixels by the total unique pixels in both predicted and ground-truth masks.

$$\text{Mask IoU} = \frac{\text{Shared Pixels}}{\text{Total Unique Pixels in Both Masks}} \tag{8}$$

Step 4B: Box-level Intersection over Union (Box IoU)

We implement the common technique of calculating Intersection over Union (IoU) by comparing the predicted bounding box with the ground-truth bounding box. First, we extract the predicted bounding box from the binary mask B_i by finding the minimum and maximum x and y coordinates to enclose all top-K% pixels in the predicted box.

Equation 9 shows the calculation of the Box-level Intersection over Union (Box IoU). The area of overlap is divided by the total union area of both boxes.

$$\text{Box IoU} = \frac{\text{Area of Overlap}}{\text{Area of Predicted Box} + \text{Area of Ground-truth Box} - \text{Area of Overlap}} \tag{9}$$

We provide a comprehensive framework of evaluating the explainability methods using the performance metric IoU in both pixel-level and box-level. Mask IoU measures the precise shape and Box IoU checks whether the explanation stays within the annotated boundaries. Both metrics evaluate the spatial alignment between the explainability methods and the ground-truth annotations.

In our study, a low IoU score is not necessarily considered an indicator of poor performance by the explainability methods. In some cases, a low IoU occurs when the highlighted region represents a small yet highly discriminative feature, but the annotated box is large and covers the entire object. Therefore, the explainability methods highlight the target object precisely and remain valid even if they do not cover the entire ground-truth area.

4.2.2 Pointing Game

Pointing Game is also a localization-based metric, meaning it evaluates how well the produced visual explanation aligns with the ground-truth annotations (Zhang et al., 2025). The concept of the Pointing Game metric is straightforward, it checks whether the pixel with the highest intensity falls within the ground-truth annotation. An image is counted as a "hit" when this pixel is inside the ground-truth bounding box. Otherwise, it is noted as a "miss" (Kashefi et al., 2026). The localization accuracy for each class is calculated by using the ratio of hits to the total number of images in that class, as shown in Equation 10 (Zhang et al., 2018).

$$\text{Pointing Game Accuracy} = \frac{\text{Total Hits}}{\text{Total Hits} + \text{Total Misses}} \tag{10}$$

The Pointing Game is based on the assumption that a well-trained classifier relies on features within the annotated object to make a correct prediction (Ventura et al., 2024). Therefore, an explainability method should place its peak attribution within the target object bounding box. A higher Pointing Game accuracy indicates that the method is mostly hitting the correct target and not the background. Evaluation using the Pointing Game metric is stable because it is parameter-free. Therefore, the results do not vary based on the chosen threshold, unlike IoU. This metric effectively checks whether the explainability method to focuses its peak attention on the annotated object, even if the remaining parts of the heatmap contain noise.

4.2.3 Deletion and Insertion Area Under Curve (AUC)

Deletion and Insertion Area Under Curve (AUC) is an evaluation metric like the Pointing Game that does not depend on the choice of the threshold (Rainio et al., 2024). This metric focuses on measuring faithfulness by finding the correlation between pixel importance and changes in model confidence (Zhang et al., 2025). Tests are carried out by perturbing input pixels identified as important by the explainability method to observe changes in the predictions. The objective is to determine whether the highlighted features are truly considered important by the model's output. In our evaluations, we used the Deletion and Insertion Area Under Curve (AUC) metric to measure faithfulness.

- **Deletion Area Under Curve (AUC)**

  Deletion Area Under Curve (AUC) is a metric that works by iteratively removing the pixels from an image to observe how fast the model's confidence in the prediction drops (Kashefi et al., 2026). First, we have an original input image and a baseline image, which is a blurred version of the image using a Gaussian filter with σ=10 to ensure the model cannot recognize the object (Petsiuk et al., 2018). We adopted the strategy of blurring the image instead of directly "blacking out" pixels to prevent introducing sharp edges.

After the saliency map of the explainability method is flattened, all pixels are ranked in descending order using the argsort function from NumPy library. Then, we have identified the most important pixels and can perform the deletion in 100 steps. In each step, approximately 1% of the total pixels (the most important remaining pixels) are substituted with their counterparts from a Gaussian-blurred baseline. We calculate the final Deletion AUC using the trapezoidal rule in Equation 11 after tracking the model's classification probability at each step.

$$\text{AUC} = \sum_{i=1}^{n} \left(\frac{c_{i-1} + c_i}{2} \right) \Delta x_i \tag{11}$$

In Equation 11, c_i represents the model's confidence using softmax probability for the prediction at step i. Δx_i denotes the fraction of pixels modified at each interval, which is a value 0.01 for the 100 steps process.

We visualize the Deletion AUC in red by plotting a faithfulness graph, where the x-axis represents the percentage of pixels removed and the y-axis represents the model's classification probability for the target class. A lower Deletion AUC indicates that the probability declines sharply when the important pixels are removed, thereby proving that those features were the reason for the model's classification (Kashefi et al., 2026).

- **Insertion Area Under Curve (AUC)**

  Insertion Area Under Curve (AUC) follows the same process as Deletion AUC but in reverse order (Kashefi et al., 2026). Instead of replacing the original pixels with a blurred version, the Insertion AUC starts from a completely blurred baseline image. Then, it gradually inserts the original sharp pixels into the image, starting with the highest ranked most important pixels. After all 100 insertion steps, the final Insertion AUC is calculated using the Equation 11.

  The Insertion AUC is visualized as green on the same faithfulness graph as the Deletion AUC. For this curve, the x-axis represents the percentage of pixels inserted from 0% to 100%. A higher Insertion AUC implies that if the explanation is precise, the model's confidence should increase significantly as important features are inserted (Zhang et al., 2025).

The faithfulness graph provides a comprehensive visualization of how the confidence of model changes when the important pixels are perturbed. For a highly faithful saliency map from the explainability method, the Deletion curve (red) should decline rapidly toward the bottom-left of the graph, while the Insertion curve (green) should boost quickly toward the top-right of the graph (Kashefi et al., 2026).

4.2.4 Recall

In our experiment, we use recall rather than standard accuracy to evaluate the performance of the ResNet-18 model on an image classification subset of the ImageNet-1k validation dataset. The evaluation is conducted on a per-class basis, meaning the evaluation of a selected class is completed before proceeding to the next class. Thus, recall is utilized to measure how many samples in that selected class were predicted correctly with that label, regardless of how many other samples were in the dataset. Since the validation dataset is filtered and each class has an imbalanced number of samples, using recall is a more appropriate approach to accurately measure the performance of the model on each specific class (Juba and Le, 2019).

The classification performance is evaluated using recall as shown in Equation 12 (Grandini et al., 2020).

$$\text{Recall} = \frac{TP}{TP + FN} \tag{12}$$

Where TP represents True Positives, denoting the number of correctly predicted instances and FN represents False Negatives, denoting the number of instances belonging to the class that were predicted incorrectly.

Then, an aggregated performance measure is calculated across all classes by dividing the sum of individual recall scores by the total number of classes.

5 Experiments and Results

5.1 Dataset and Preparation

5.1.1 Dataset

ImageNet is a large-scale hierarchical image database that was introduced in 2009 by a computer science team from Princeton University (Deng et al., 2009). This dataset initially consisted of around 3.2 million images across 5247 synsets and with an average of 500–1000 images representing each synset. It was created based on the WordNet structure, where each meaningful concept in WordNet is described by multiple words and grouped into a synonym set, also known as synset. This initial ImageNet dataset featured WordNet's 12 main subtrees: mammal, bird, fish, reptile, amphibian, vehicle, furniture, musical instrument, geological formation, tool, flower and fruit. Because the dataset is huge and based on diverse natural image classes, it is commonly used for tasks including object recognition, image classification and automatic object clustering. As shown in Figure 16, classes such as "husky" and "trimaran" are nested within a broader semantic hierarchy, ranging from specific breeds or boat types up to general categories like "watercraft" or "sailboat".

The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) was first organized in 2010 as a contest for software programs to compete in classifying and

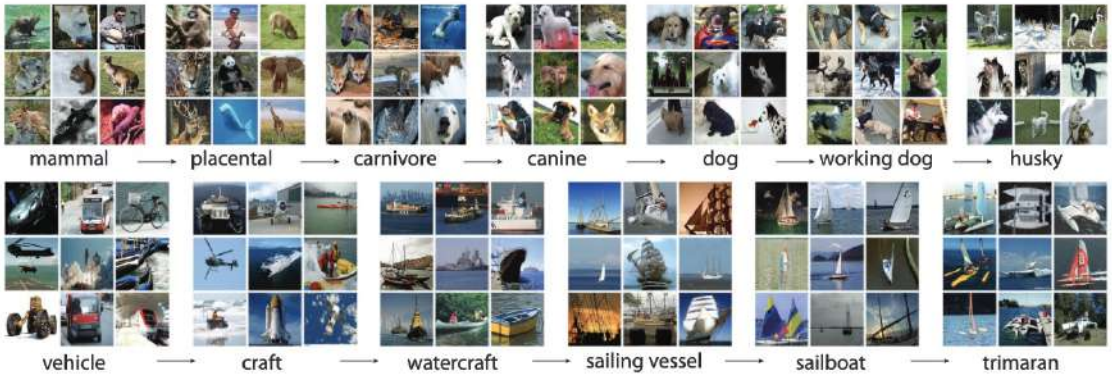
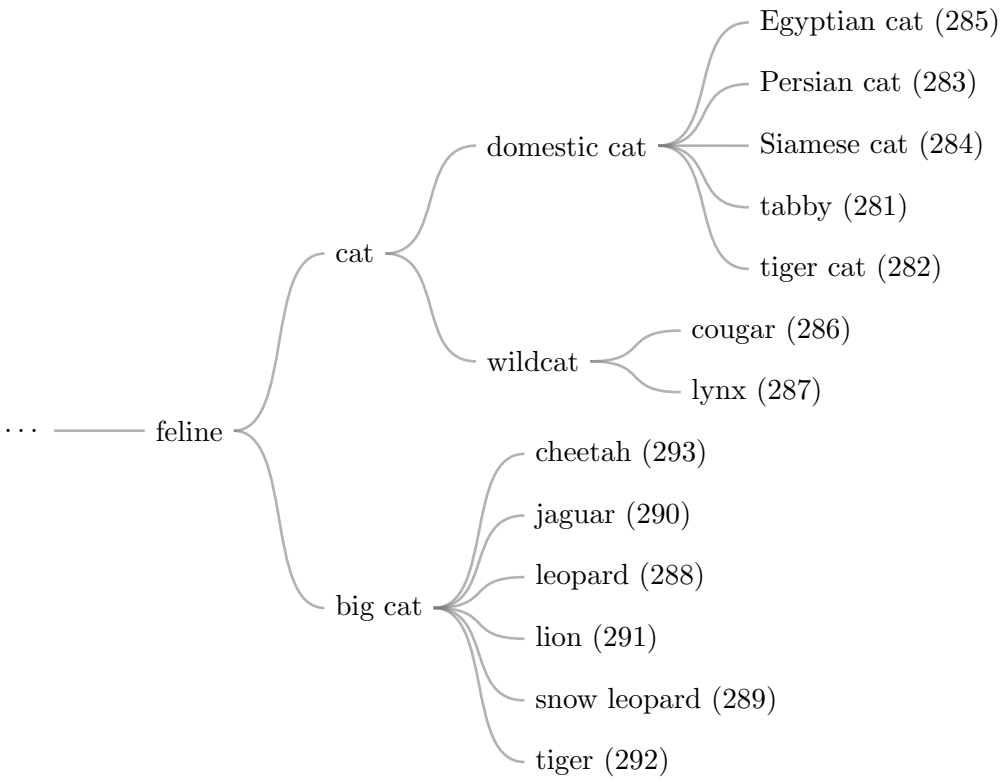


Figure 16: A snapshot of two root-to-leaf branches from ImageNet. The top row follows the mammal subtree to the "husky" class; the bottom row follows the vehicle subtree to the "trimaran" class. Each synset includes nine randomly sampled images (Russakovsky et al., 2015).

detecting objects and scenes correctly (Russakovsky et al., 2015). The ILSVRC results increased the popularity of the ImageNet dataset and it became a foundational benchmark for training computer vision models.

The dataset used was a subset of the original ImageNet dataset from Deng et al. (2009), which is also the same dataset used in our experiment. The 1000 synsets were selected, ensuring that there is no overlap between synsets (Russakovsky et al., 2015). As synsets are part of a larger hierarchy and may have children classes in the original ImageNet dataset, the ILSVRC dataset does not include their child subcategories. Thus, the ILSVRC version of the ImageNet dataset is a more condensed version of the original ImageNet hierarchy.

Bostock (2018) mapped all 1000 classes to their corresponding WordNet nodes to visualize the taxonomic relationships of the ILSVRC dataset using a tree structure. The hierarchy stems from an "entity" root and then splits into major branches like "living thing" (451 classes) and "artifact" (549 classes). The tree below demonstrates what a WordNet hierarchy structure looks like. It illustrates the path from the intermediate node "feline" to specific class-level leaf nodes and draws species that share the "domestic cat" parent node, such as the "Siamese cat", "Egyptian cat", "Persian cat", "tabby" and "tiger cat". This structure allows us to evaluate how explainability methods interpret the model's ability to distinguish between these closely related species within the same taxonomic branch. In this experiment, we selected two to three class-level leaf nodes of the same parent nodes from each of the 12 subtrees. This sampling strategy provides a comprehensive evaluation across the ImageNet dataset and allows for a fair, balanced and generalizable comparison.



The ILSVRC dataset was updated for each year's competitions and ImageNet dataset usually refers to the ILSVRC-2012 version, as the synsets have remained unchanged since then (Russakovsky et al., 2015). This allows researchers to compare models and evaluate them using the same data, which is one of the reasons we selected this dataset. From here onwards, the terms "ImageNet", "ImageNet-1k" and "ILSVRC" refer to the same dataset, which is the ILSVRC-2012 version of the ImageNet dataset. This dataset contains around 1.2 million training images, 50,000 validation images and 100,000 test images across 1000 classes. Each image is labelled with exactly one ground-truth class. In this experiment, we used ImageNet-1k, a dataset that is widely used benchmark dataset in computer vision for image classification tasks. We evaluate only the validation dataset, as the scale is manageable and we focus on interpreting how the models predict, rather than training a model.

The authors identified specific classes that are particularly easy or difficult for image classification based on "optimistic" results (Russakovsky et al., 2015). These aggregated results were based on the best performance achieved by any of the 21 entries from the ILSVRC 2012–2014 competitions, including popular models such as GoogLeNet, VGG and SuperVision (AlexNet). Rather than focusing on a single architecture, this approach identified the maximum accuracy achieved for each class by any participating model in the contest. Under these combined results from the best performing model, 121 labels achieved 100% accuracy, while the most challenging class had a low accuracy with only 59.0%.

As shown in Figure 17, the randomly chosen 10 easiest classes include mammals such as "red fox", "tiger", "hamster", "ibex", "flat-coated retriever" and "Blenheim

spaniel" and animals with distinctive features such as "stingray". We observed that most of the classes are from the mammal subtree or the major branch "living thing". In contrast, the hardest classes in the classification task are from the major branch "artifact" such as man-made things like "water bottle", "spotlight" and "letter opener".

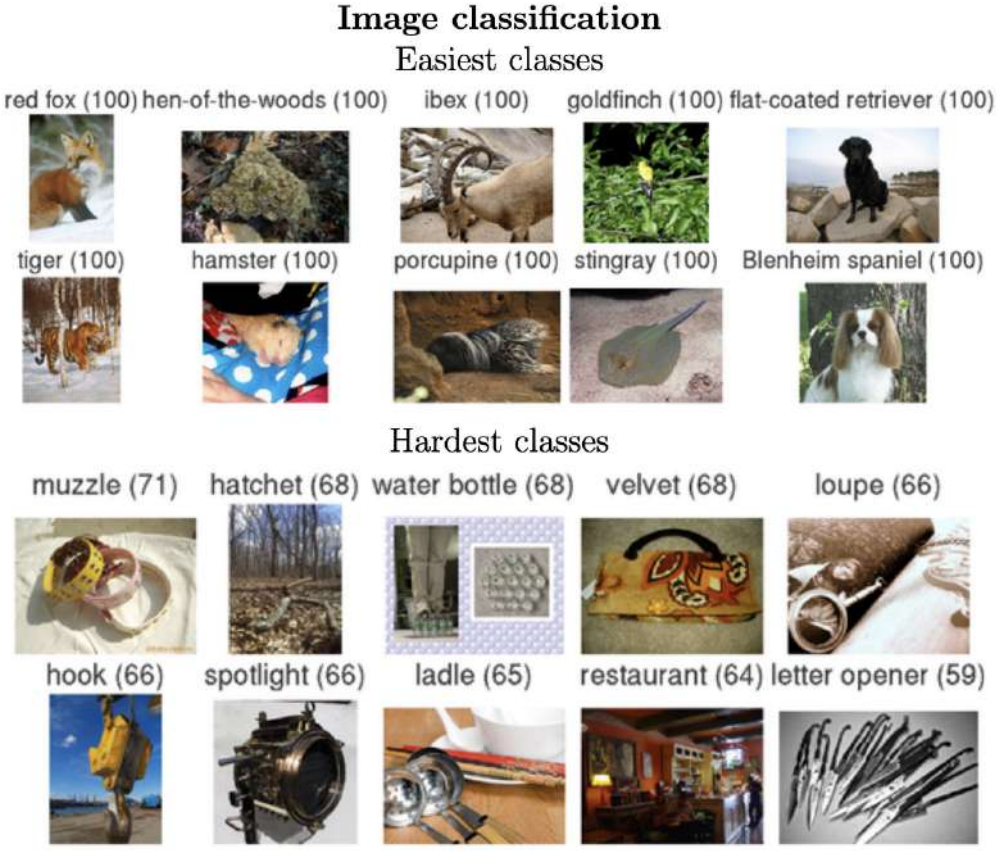


Figure 17: The 10 easiest and hardest classes for the image classification task aggregate results from best performing models from the particular class. The values in parentheses represent classification accuracy. 10 Easiest classes are randomly chosen from among 121 classes that all had 100% accuracy (Russakovsky et al., 2015).

5.1.2 Dataset Preparation

Before the experiment, we performed data filtering to focus solely on single-object images containing only one bounding box. These criteria were applied to ensure the images used in the evaluation were clearly visible and well-proportioned. There were approximately 27,000 images remaining after the dataset filtering process. Any instance where the bounding box was larger than 80% or smaller than 10% of the image area was excluded to ensure each image contained a single, well-proportioned object. We also removed any samples with ambiguously incorrect ground-truth

labels. We carried out the data filtering process according to the sequence below, along with the reasons:

- **Multiple objects or bounding boxes:**

  In ImageNet, some of the images may contain more than one object. This may lead to multiple valid labels being selected or even cases where the annotated label does not match the most prominent object in the image (Tsipras et al., 2020). Sometimes, not all objects are annotated with a ground-truth bounding box even though they belong to the same class. Therefore, in some cases, when explainability methods correctly identify one of the objects that has no ground-truth bounding box, the performance metrics do not align with showing how well that particular method interprets the model's prediction. For example, the IoU metric will be low even if the method is correct, because the method highlights an object without a ground-truth bounding box rather than the one that is annotated. This data filtering criterion effectively prevents punishing the methods for correctly identifying a single object when multiple exist.

  - **Scenario 1:** An image contains multiple objects, but the explainability method highlights only one of them that is not annotated with a ground-truth bounding box. Figure 18 shows an example of two cats of the same species, "Siamese cat", but only one is annotated with a ground-truth bounding box. Another example in Figure 19 shows a more extreme case where there are numerous "pretzel" objects, but the explainability method does not focus on the one with the annotated ground-truth bounding box at all.

    Problem: Low IoU despite correct localization of an object that is not annotated with a ground-truth bounding box.
  - **Scenario 2:** An image contains multiple objects of different classes. As the ImageNet dataset only has one ground truth label per image, only one object is identified and annotated. Figure 20 shows an example for this scenario.

    Problem: The model's predictions or explanation methods may highlight an object that is not annotated. This can result in incorrect predictions or cause explanation methods to focus on the wrong features.

- **Too high bounding box coverage ($> 80\%$):**

  - **Scenario:** If an image has a ground-truth bounding box that covers more than 80% of the total image area, the object occupies almost the entire image and only a little background context is left. Therefore, we excluded these images from our experiment.
  - **Problem:** If an explainability method generates a heatmap covering the entire image, we cannot observe how the model differentiates an object from a complex background. In such cases, the evaluation is less meaningful because these images are too easy to achieve a high IoU.

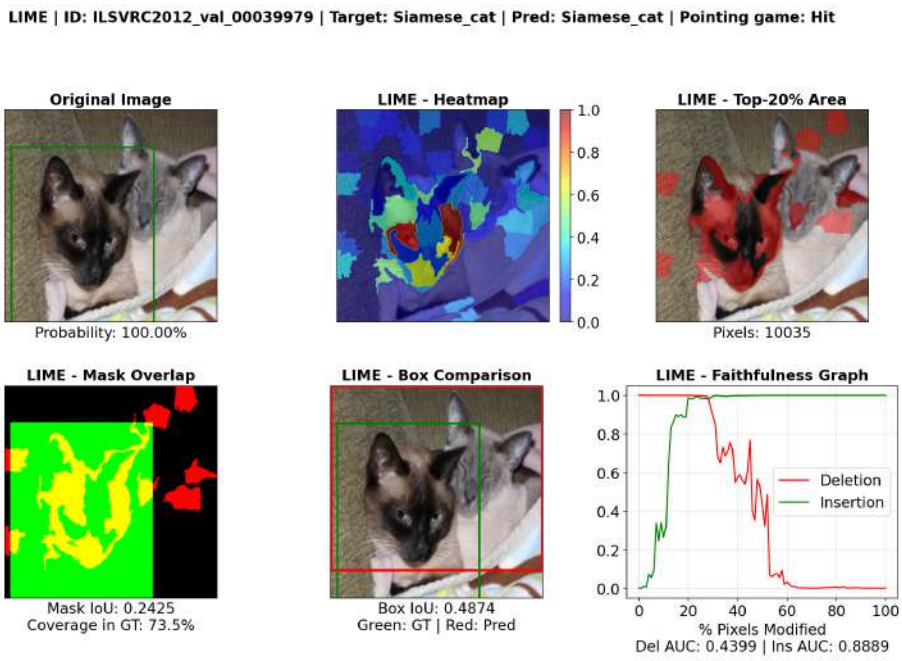


Figure 18: A sample with multiple objects that share the same class "Siamese cat", but only one cat is annotated with a ground-truth bounding box in green. We refer to the annotated cat as Cat 1 and the unannotated cat as Cat 2. Although the model makes a correct prediction and the LIME explainability method focuses primarily on Cat 1, it also captures some features of Cat 2, resulting in a larger predicted bounding box based on the top 20% area of the image. This makes the box IoU low, which is 0.5134. This shows that even when LIME successfully captures the features of the "Siamese cat", the box IoU is relatively low.

- **Too low bounding box coverage ($< 10\%$):**
  - **Scenario:** The object covers less than 10% of the image area, making it potentially unrecognizable in the image content.
  - **Problem:** The model may not focus on this object when classifying the image. In other words, the model may focus on the background context instead of the tiny object when making a correct prediction.
- **Ambiguous wrong label**

  The ImageNet samples were collected from the internet and according to the WordNet hierarchy and annotated by humans. Hence, it is common for issues to exist in any human-annotated real-world dataset (Kisel et al., 2024). One reason for this is the difficulty in distinguishing between similar animal species (Tsipras et al., 2020). For instance, there are 24 distinct terrier breeds in ImageNet. Therefore, it is unreasonable to expect non-experts to differentiate all of them correctly.

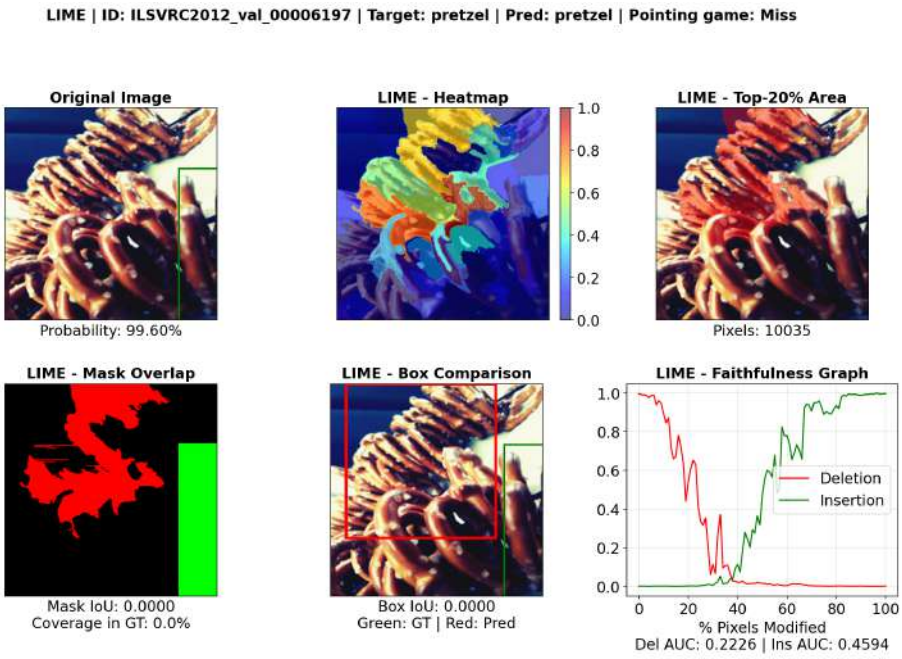


Figure 19: A sample with multiple objects that share the same class "pretzel", but only one of them, located at the bottom rightmost of the image, is annotated with a ground-truth bounding box in green. The LIME method does not highlight any region of the annotated pretzel, but instead focuses primarily on those located in the middle of the image. This results in a zero IoU, even though the method correctly identifies pretzel objects.

After data filtering, we applied the standard ImageNet transformation. First, images were resized to 256 x 256 pixels and then center-cropped to 224 x 224 pixels. After that, we normalized the images using the ImageNet mean and standard deviation. This preprocessing pipeline is consistent with the ResNet-18 model training procedure.

5.2 Pretrained Model

In this experiment, we utilized a pretrained ResNet-18 model trained on ImageNet-1k from the PyTorch Image Models `timm` open-source library (Wightman, 2019).

Residual Network (ResNet) architecture was originally introduced by He et al. (2016). The authors observed that deeper neural networks are more difficult to train as the degradation problem emerges. Their experiments showed that gradients were not vanishing, but the networks showed higher training error when going deeper. They proposed a model based on residual learning framework to address this issue. Figure 21 shows the residual learning building block. A plain network will learn the desired underlying mapping $H(x)$ directly. In contrast, ResNet blocks are designed to learn only the difference, also referred to as residual, between the input and output, which is defined as $F(x) := H(x) - x$. By using "shortcut connections"

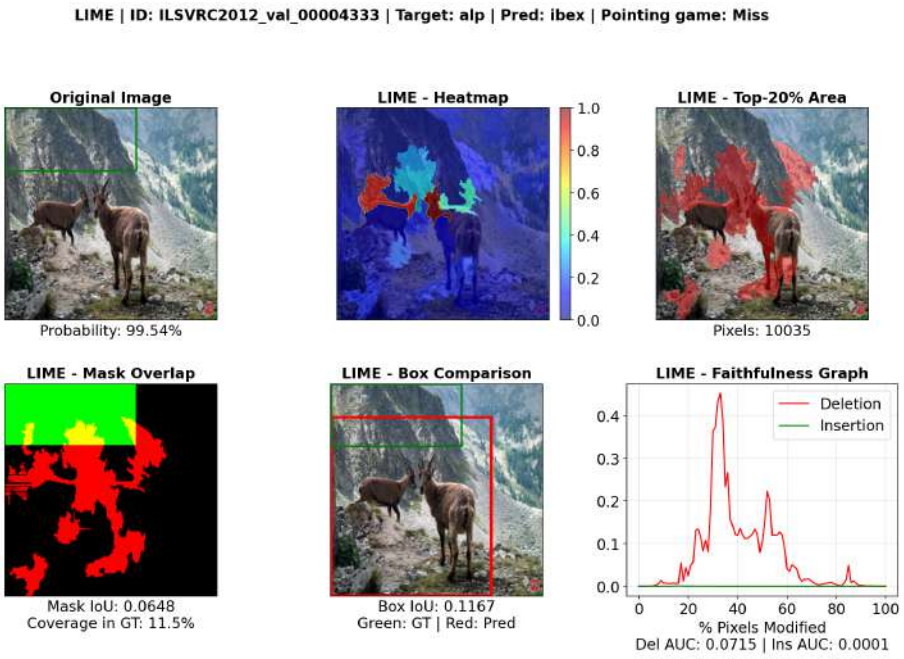


Figure 20: A sample with multiple objects from different ImageNet classes. This image is labelled as the class "alp", which appears in the background of the entire image, while two other "ibex" class objects are located in the middle of the image. The model classifies this image as "ibex" because it focuses more on the middle part of the image, as shown by the LIME method. This demonstrates why excluding images with multiple objects, especially those from different classes, is important in our evaluation.

that skip one or more layers, the network can perform an identity mapping by adding the input x to the output of the stacked layers, formulated as $H(x) = F(x)+x$. This optimization process is simplified as the network layers only need to adjust small residual values rather than the entire transformation. This architecture benefits the models with very deep layers to learn identity mappings more easily, thereby solving the degradation issue.

ResNet is built with residual blocks, a convolutional layer, four max-pooling layers, an average pooling layer and a SoftMax output layer (Tsirtsakis et al., 2025). In ResNet-18, each residual block consists of two 3×3 convolutional layers, where the input is added to the output with a "shortcut connection". In the past few years, there have been many variations and improvements to the original ResNet architecture. Table 1 is a list of specifications and the performance metrics of the ResNet-18 model implemented in this experiment.

The ResNet-18 model has a 7×7 convolution as its initial layer (He et al., 2016). Then, the ResNet-B architecture, a slightly modified version of the original ResNet design, is employed to address the spatial information loss issue. The change occurs in the downsampling process, which reduces the feature map size by half, by moving it from the first 1×1 convolution of the residual block to the second 3×3 convolution,

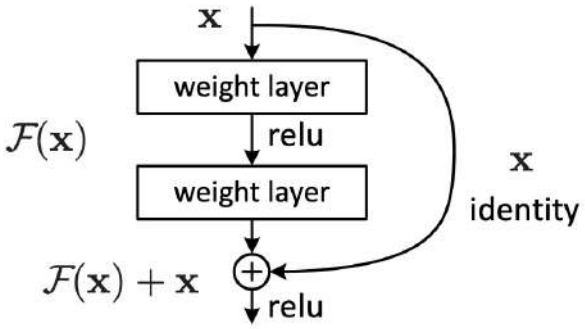


Figure 21: A residual learning building block is shown (He et al., 2016). Each block consists of an identity mapping x, a residual mapping $F(x)$ and the output of the block after the "shortcut connection", which is defined as $H(x) = F(x) + x$.

Table 1: Specifications and performance metrics of the `resnet18.a1_in1k` model (Wightman, 2019; Wightman et al., 2021).

Specification	Value
Model Variant	`resnet18.a1_in1k`
Architecture	ResNet-B
Parameters (Millions)	~ 11.7
Computational Cost (GMACs)	~ 1.8
Activations (Millions)	~ 2.5
Input Size (Train / Test)	224×224 / 288×288
Top-1 Validation Accuracy	$\sim 71.49\%$
Top-5 Validation Accuracy	$\sim 90.07\%$
Activation Function	ReLU
Stem (Initial Layer)	7×7 Conv + Max Pooling
ResNet-B Downsample	3×3 Convolution
Shortcut Downsample	1×1 Convolution

while integrating 1×1 projection shortcuts for dimension alignment (Wightman, 2019).

The `resnet18.a1_in1k` model variant is trained using Recipe A1, which is an improved training procedure (Wightman et al., 2021). For example, the original ResNet was trained using a standard step-wise learning rate schedule (He et al., 2016).

Recipe A1 improves the training procedure by utilizing a longer, cosine learning rate schedule with a warm-up phase (Wightman et al., 2021). In their study, after employing the Recipe A1, which uses 600-epoch cosine learning rate schedule and the LAMB optimizer for large batch sizes like 2048, they improved the Top-1 accuracy of ResNet-50 from 75.3% to 80.4%. The list below highlights the key features of Recipe A1 for the `resnet18.a1_in1k` model trained on ImageNet-1k.

- **Recipe:** ResNet Strikes Back A1
- **Optimizer:** LAMB optimizer
- **Loss Function:** Binary Cross-Entropy (BCE) loss
- **Learning Rate Schedule:** Cosine schedule with linear warm-up

5.3 Experimental Results

In this experiment, a pretrained ResNet-18 architecture was chosen as our "black-box" model (Wightman, 2019). This model was utilized to perform image classification tasks on the filtered ImageNet-1k validation dataset (Russakovsky et al., 2015). Then, explainability methods, including Grad-CAM by Selvaraju et al. (2017) and LIME by (Ribeiro et al., 2016), were employed on the same input image to evaluate the model's decisions. In this study, we evaluated these explainability methods both qualitatively and quantitatively.

The evaluation was performed by comparing all input images of the selected class before proceeding to the next class instead of processing the entire validation set at once. In this approach, the comparison of outputs from different instances of the same class is possible, in order to have a clearer picture of whether the model consistently focused on the same features across multiple samples. We then extended this analysis to other very similar classes to observe the shared characteristics of the objects, how the model distinguishes between them or even why it makes wrong predictions. Thus, we can observe the specific scenarios in which the model becomes confused. These visual explanations illustrate whether the model focuses on the right area in the image when making a prediction or if it learned the background features during the training. The ImageNet dataset consists of 12 main WordNet subtrees (Deng et al., 2009). In this evaluation, at least two classes from each subtree are represented.

5.3.1 Qualitative Evaluation

Case Study 1 presents a full visual evaluation for Grad-CAM and LIME. The remaining figures illustrate specific interesting insights, with findings provided either before the visuals or in the captions.

Case Study 1: Siamese Cat - Figure 22 (Grad-CAM) Figure 23 (LIME)

The model predicted the image correctly and assigned a high probability to the class "Siamese cat". The green square on the original image represents the ground truth bounding box. Both methods focus on the object successfully instead of relying on the background cues. According to the Pointing Game metric, both methods achieve a "hit" within the ground truth bounding box, meaning that their highest focus is correctly placed on the target object.

While both methods identify the face as the primary region of interest, they offer different levels of granularity. In Grad-CAM, the entire head structure was covered by a high intensity "red blob", indicating it as the primary feature contributing to the prediction. In contrast, LIME provides more specific insights, identifying the eyes as the highest and second-highest influential segments. Thus, LIME enables the identification of discriminative, fine-grained features within an object, whereas Grad-CAM provides a more generalized spatial region.

When observing the Faithfulness Graphs, the deletion and insertion curves intersect approximately at the top 20% pixel mark, identifying this threshold as the minimal sufficient explanation where the most discriminative information was concentrated. Therefore, using the top 20% of pixels to compute the Intersection over Union (IoU) is a reasonable approach as it filters out background noise to focus strictly on the regions that drive the model's confidence. Both methods demonstrate high insertion AUC and low deletion AUC, indicating that they provided highly accurate and faithful explanations by identifying the most influential features for the model's decision.

The Box IoU scores of both methods are higher than their Mask IoU scores. Moreover, the LIME Box IoU was significantly higher than Grad-CAM. However, this does not indicate that LIME has better explainability or that Grad-CAM has failed. In fact, Grad-CAM demonstrate a more precise focus on the object. Because the provided ground truth bounding box does not tightly enclose the cat, LIME's higher score was an artifact of its top 20% of pixels including the background rug. In contrast, Grad-CAM remained accurately concentrated focus on the cat's body. This is further confirmed by the fact that 100% of Grad-CAM top 20% pixels fell within the ground truth bounding box.

Across all samples of correct "Siamese cat" predictions, both methods consistently highlight the eyes as an important feature. Given the Siamese cat's distinctive blue eye color, it is highly likely that the model utilizes this specific trait to distinguish Siamese cats from other breeds.

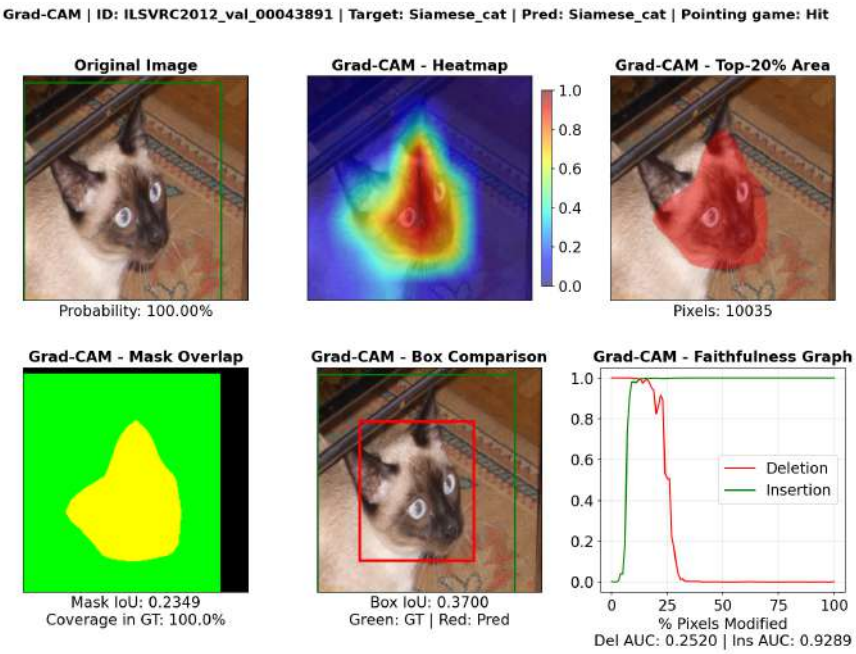


Figure 22: Grad-CAM explanation for correct prediction: "Siamese Cat".

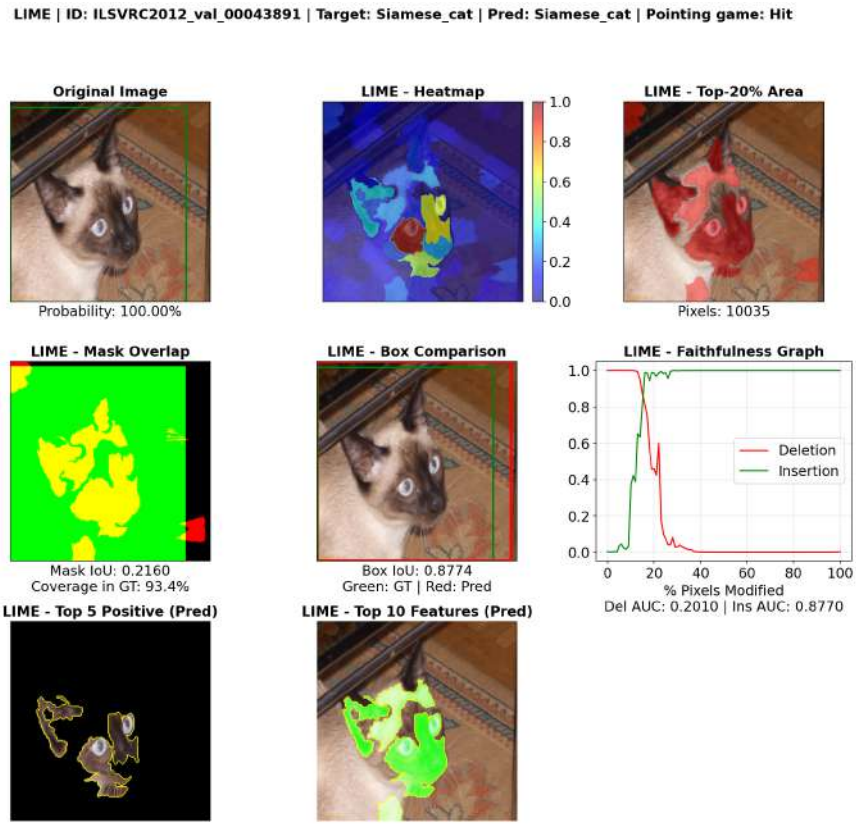


Figure 23: LIME explanation for correct prediction: "Siamese Cat".

Case Study 2: Siamese Cat Misclassification - Figure 24 (Grad-CAM) Figure 25 (LIME)

The model misclassified the image as "Siberian Husky", despite the correct class being "Siamese Cat". However, the probability was less than 50%, indicating low confidence and uncertainty in the classification.

As shown in the figures, the specific trait of Siamese cat, which is the blue eyes, cannot not be captured because the input image is in grayscale. Grad-CAM show its highest focus on areas of black intensity, including the paws, the left ear and the background. Furthermore, Grad-CAM fails to localize the object effectively as the high intensity "red blobs" are spread across the left and right sides of the image. The model focuses on the high contrast edges, such as the boundary between the black paws and the white rug, rather than the actual shape of the cat.

In contrast, LIME localizes the object correctly. However, the top segments for the predicted class demonstrate that the model focuses these features and attribute them to the "Siberian Husky" class due to similarities in fur color and texture. These top segments are also difficult to recognize as distinct feline features. Even though the top 10 LIME segments capture most of the cat's body, the model fails to identify the object as a cat. Thus, these results confirm that the model relies heavily on the distinctive blue eye characteristics to identify the Siamese breed.

In this misclassification case, the Faithfulness Graphs exhibit irregular behavior compared to the correct predictions in Case Study 1. The deletion curve for both methods begins at a low baseline, reaches a peak as important pixels were removed and then drops significantly. This suggests that the methods initially highlighted not the most important features, but negative evidence or background noise that suppressed the model's confidence. As these distracting components are deleted, the model's probability for "Siberian Husky" increases and then declines when the influential pixels were removed. The Insertion curve for both methods also does not show a gradual increase as important pixels were added.

This pattern, combined with the low overall AUC scores, indicates low explanatory faithfulness and reflects the model's high uncertainty.

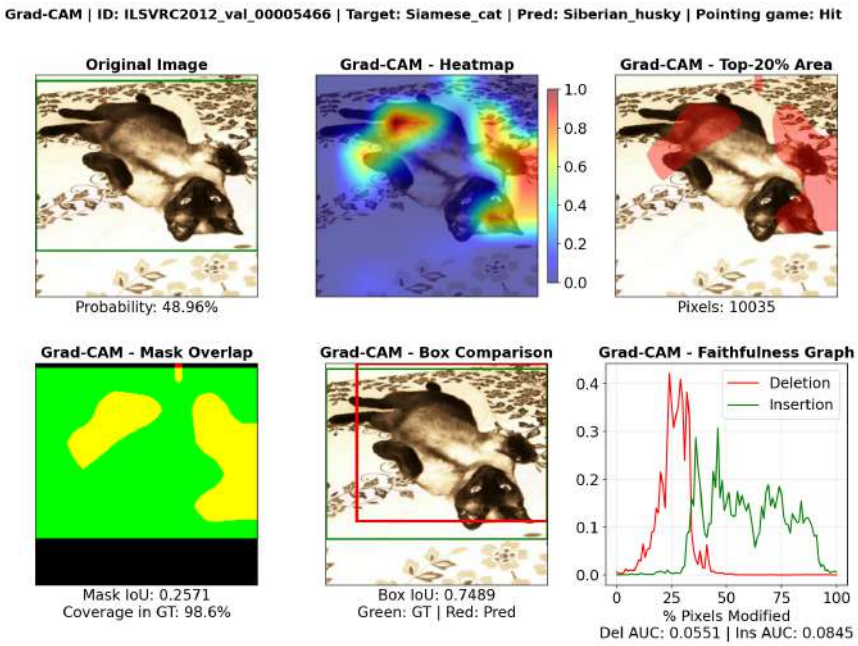


Figure 24: Grad-CAM: Predicted "Siberian Husky" vs. True Label "Siamese Cat"

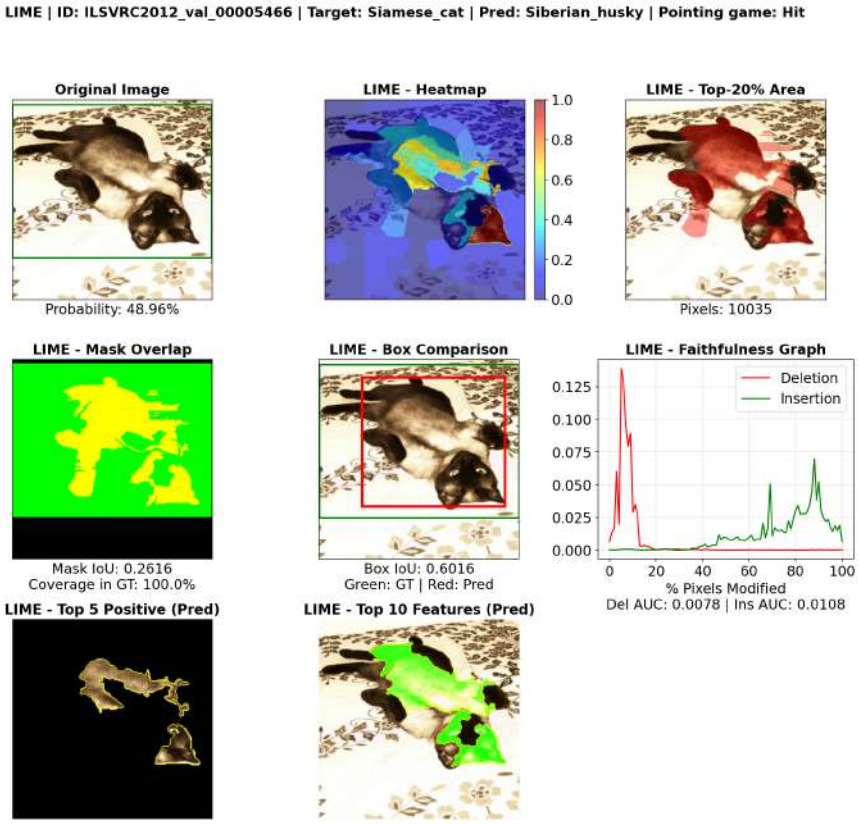


Figure 25: LIME: Predicted "Siberian Husky" vs. True Label "Siamese Cat"

Case Study 3: Flamingo - Figure 26 (Grad-CAM) Figure 27 (LIME)

The model predicted the image correctly and assigned a high probability to the "Flamingo" class. Both methods show that the model relies on the color and the body shape to identify the flamingo.

The primary focus in Grad-CAM lies on the body and the neck as the high intensity "red blob" is concentrated on those regions. The leg regions also receive a small amount of focus. LIME confirms these features as important. This shows that the model prioritizes the pink color and the main body of the bird, rather than small details like the shape of its legs or beak, to make a confident prediction for the "Flamingo" class.

Interestingly, the model also shows signs of background confusion as the Grad-CAM slightly extends to incorporate the water reflection beneath the bird and the LIME top 10 features include segments of water as well, suggesting that the model associates the aquatic environment with the "Flamingo" class.

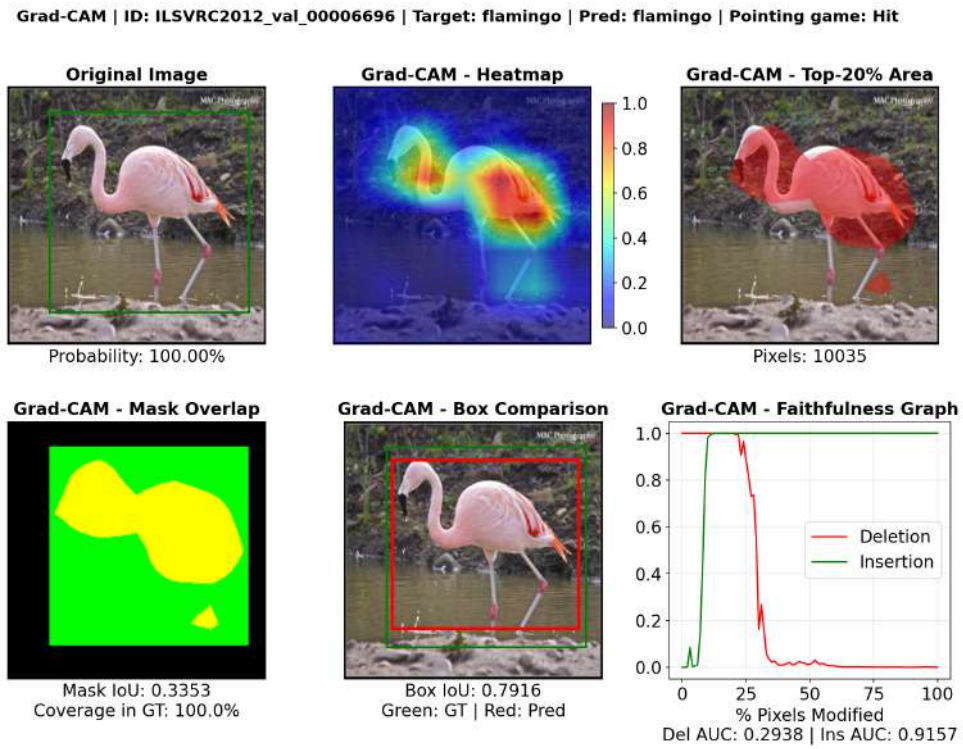


Figure 26: Grad-CAM explanation for correct prediction: "Flamingo".

Case Study 4: Flamingo Misclassification - Figure 28 (Grad-CAM)

The model misclassified the image as a "Spoonbill" with probability of around 80%. Both method focus on the object correctly and highlight the entire body structure. However, the Faithfulness Graph for this case follow the same pattern as Case Study 2, showing low AUC scores.

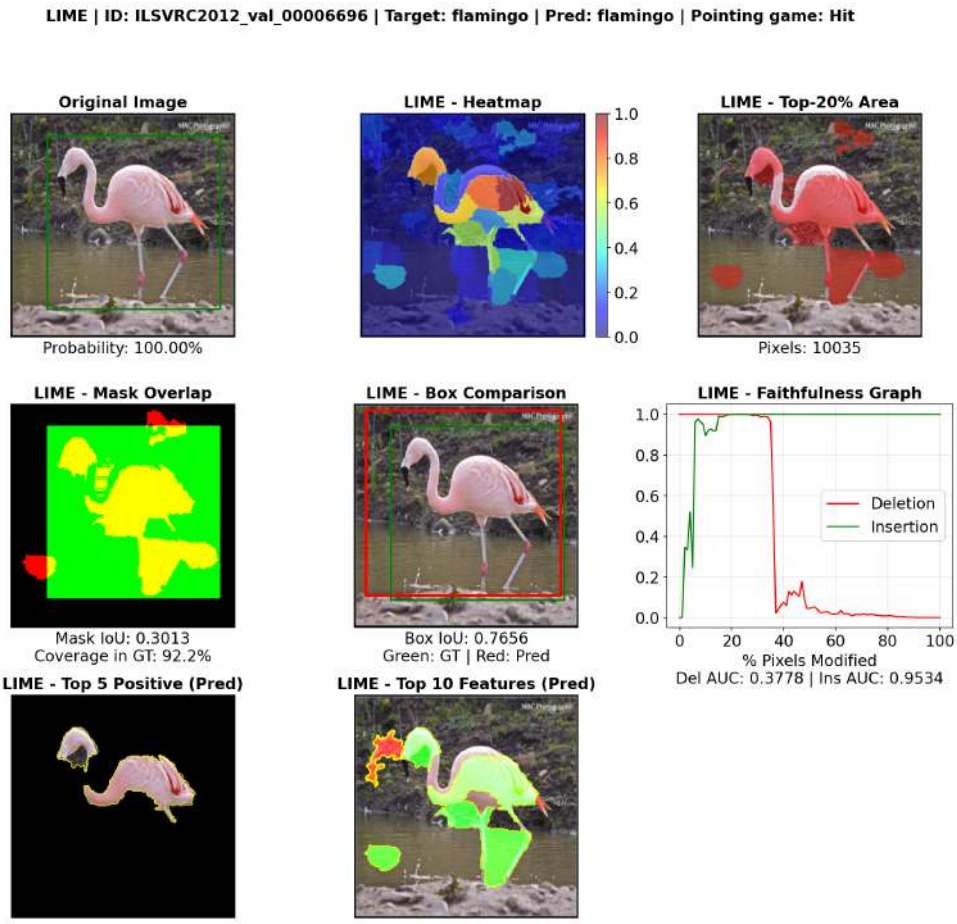


Figure 27: LIME explanation for correct prediction: "Flamingo".

Figure 29 shows a sample of a correctly predicted "Spoonbill". The Flamingo and Spoonbill are both birds that share highly similar characteristics, including pink body, long legs and a slim neck.

Out of 22 validation samples, there are three images featuring a Flamingo with its wings wide open and all of them were classified as "Spoonbill". Two of these samples featured the Flamingo with the sky as a background, while the third misclassification featured a Flamingo with its wings wide open.

This suggests that the model learned the general shape of the body instead of the specific features of the "Flamingo", including a shorter curved beak and S-shaped neck. The model also appears to prioritize the "pose" of the bird, which relates the standing still position to "Flamingo" and wings wide open as "Spoonbill".

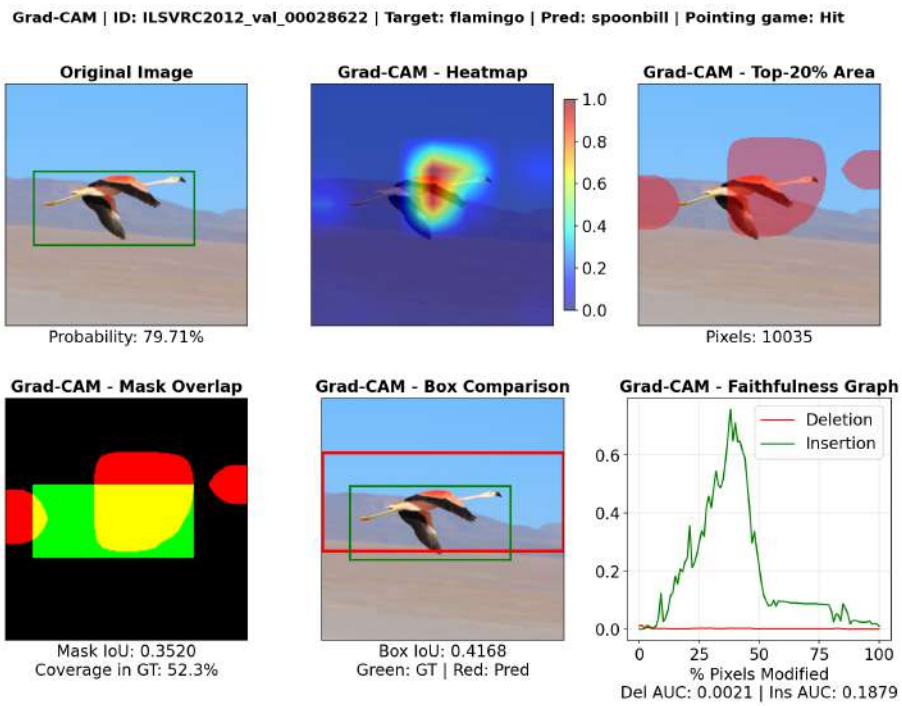


Figure 28: Grad-CAM: Predicted "Spoonbill" vs. True Label "Flamingo"

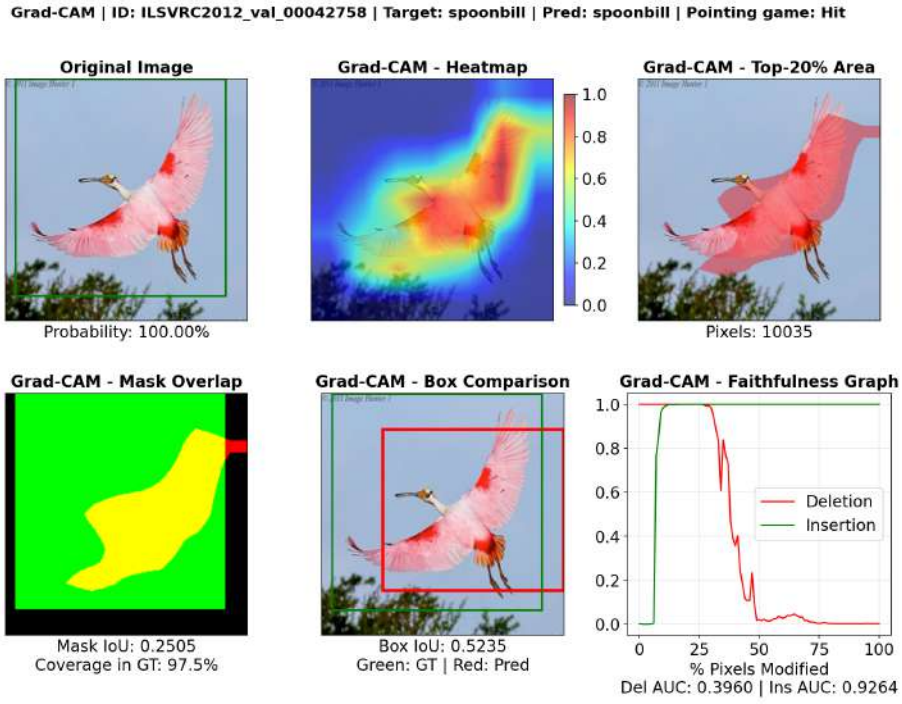


Figure 29: Grad-CAM explanation for correct prediction: "Spoonbill". The red regions highlight the wings and body as the most influential features. While a Spoonbill has a longer, wider beak and shorter legs than a Flamingo, the model relies on global shape. All 37 filtered samples were correctly predicted, 8 of them featured the "wings wide open" pose.

Case Study 5: Hammerhead Misclassification - Figure 30 (Grad-CAM) Figure 31 (LIME)

The model misclassified the image as "Tiger Shark". The Faithfulness Graph for this case follows the same pattern as Case Study 2. The low AUC scores indicate that the explanation is not faithful. Even though the model focused on the fish, the specific regions highlighted are insufficient to lead to a correct classification.

Furthermore, while both methods localized the object correctly, they do not highlight the same regions. Grad-CAM focus on the dorsal fin only. The dorsal fin of the Hammerhead is a distinct feature as it is taller than other shark species. However, the dorsal fins of other sharks may appear similarly tall, when seen from certain angles. Thus, relying solely on this feature is insufficient for correct classification without considering the Hammerhead's most important feature, which is the T-shaped head.

LIME focus mainly on the middle of the body, which appear to have high contrast stripes. However, these are actually water reflections. Figure 32 shows an example of a Tiger Shark for comparison. Even when LIME highlight the T-shaped head, it prioritizes the body features over the head structure. Thus, it categorizes the head structure as a negative segment contributing to the "Tiger Shark" prediction, despite capturing the Hammerhead's most important feature.

This case study demonstrates that the model fails to establish a proper hierarchy of features on fish, prioritizing ambiguous traits like fin height and environmental noise over the definitive anatomical markers necessary for accurate species identification.

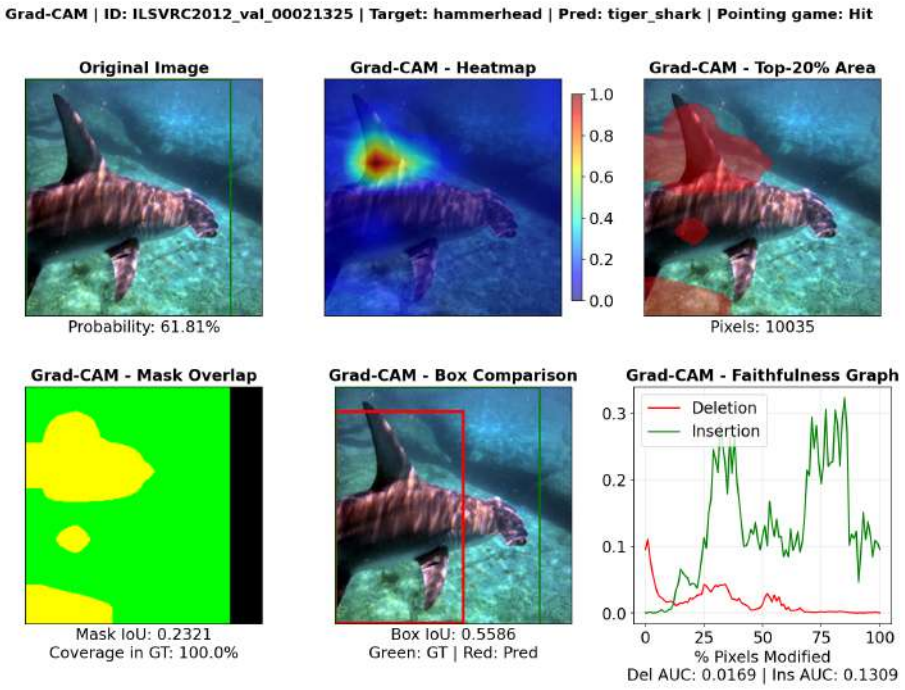


Figure 30: Grad-CAM: Predicted "Tiger Shark" vs. True Label "Hammerhead"

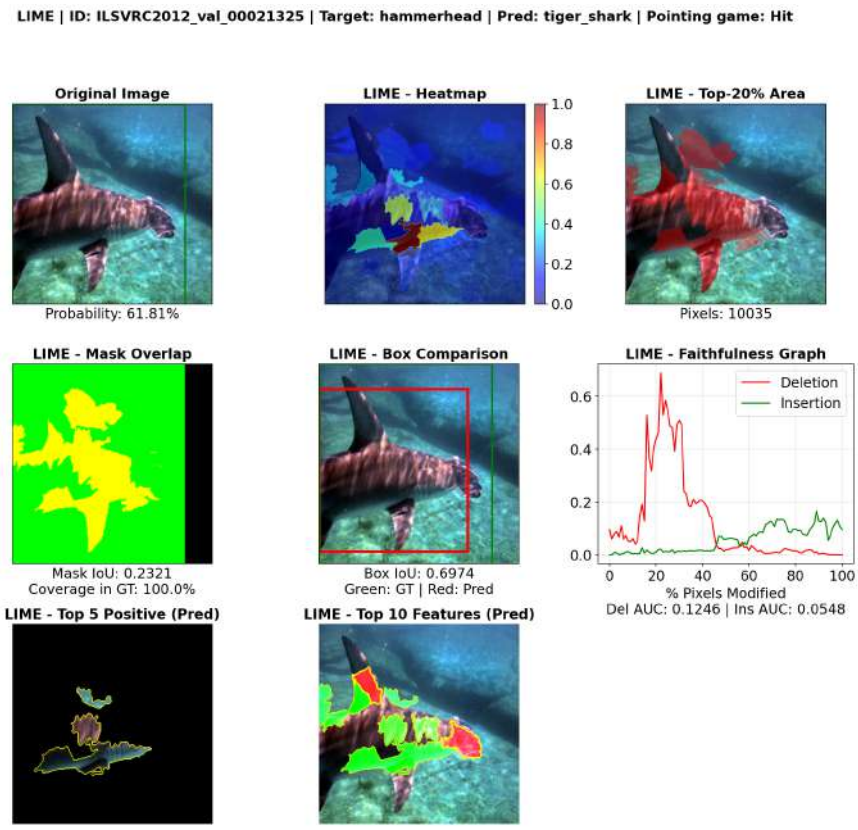


Figure 31: LIME: Predicted "Tiger Shark" vs. True Label "Hammerhead"

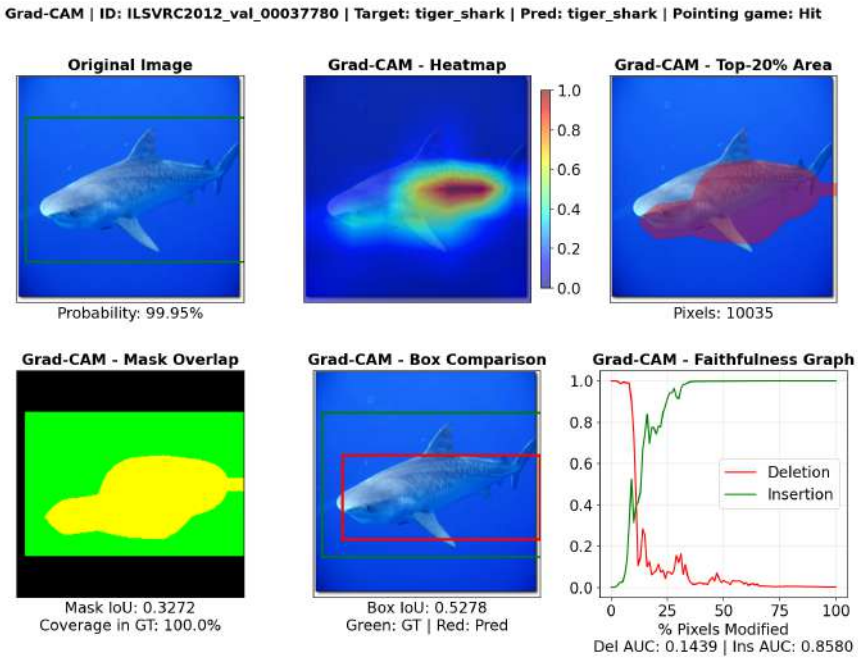


Figure 32: Grad-CAM explanation for correct prediction: "Tiger Shark". The model shows correct focus on the faint stripes on gray body, which are a distinct trait of the Tiger Shark.

Case Study 6: Cliff Misclassification - Figure 35 (Grad-CAM) Figure 33 (LIME)

The model misclassified the image as a "Valley" with a high probability of nearly 100%. Unlike previous cases, both methods failed the Pointing Game as the heatmaps do not focus on the actual cliff structure. The Faithfulness Graphs for this case follow the same pattern as Case Study 2, showing low AUC scores.

This suggests that the model prioritizes a V-shaped outline over the structural features of a mountain top or a cliff edge. Because the model associates this specific V-shaped geometry with a "Valley", it ignores the immediate vertical drop that associates with a "Cliff" However, the Cliff featured in this image is not ambiguous as the edge exhibits a consistent stone-like structure that continues vertically to the bottom.

Furthermore, the LIME important segments are spread across the image rather than being concentrated on the cliff edge or the V-shaped outline like Grad-CAM. A comparison with a correctly predicted "Valley" in Figure 34 demonstrates a similar pattern, where the top 10 important segments are spread out. The model show correct focus on the two clusters that form the V-shaped outline of the landscape and spread out textural cues. This confirms that the model relies on distributed environmental textures rather than specific object boundaries to make its decision.

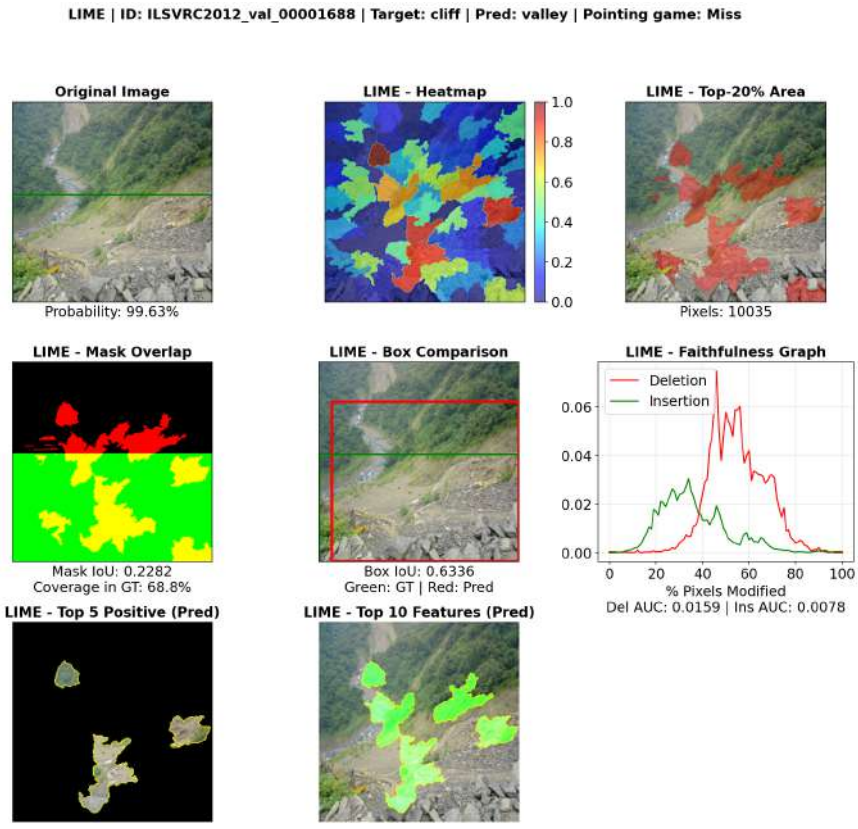


Figure 33: LIME: Predicted "Valley" vs. True Label "Cliff"

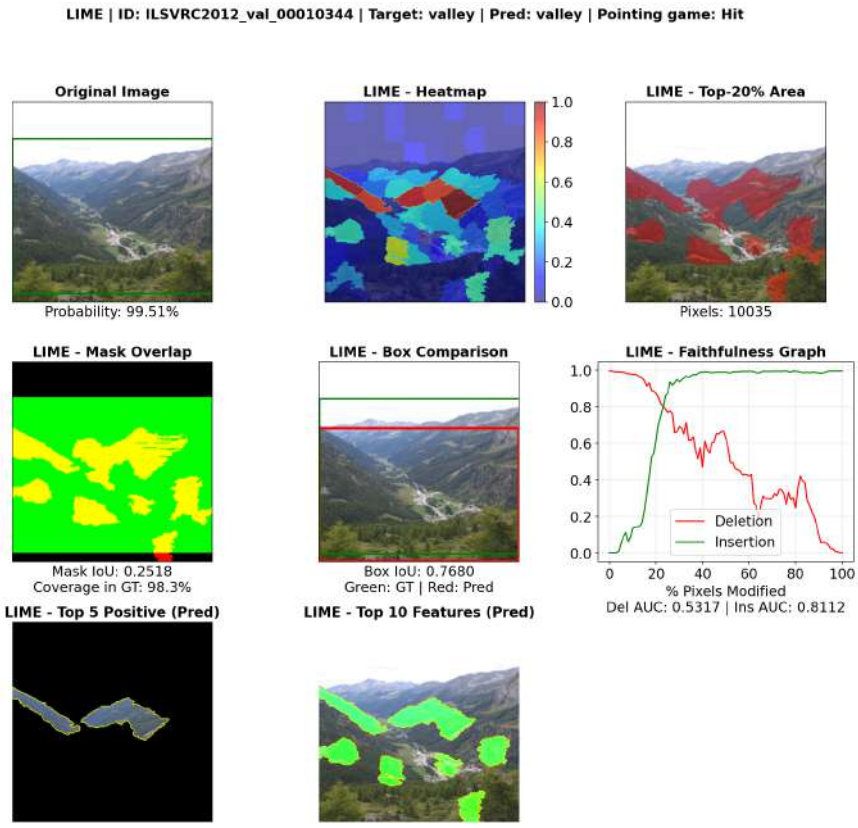


Figure 34: LIME explanation for correct prediction: "Valley".

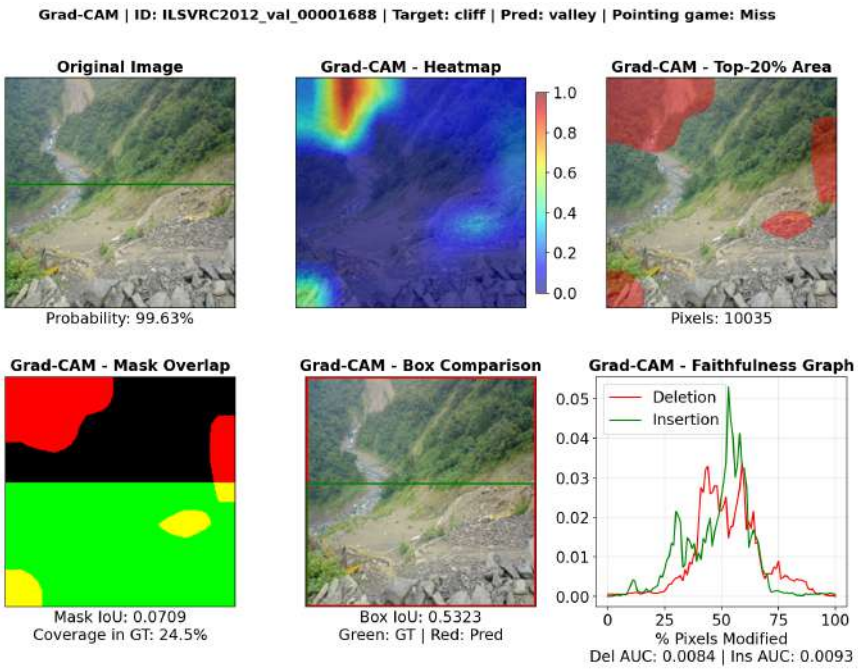


Figure 35: Grad-CAM: Predicted "Valley" vs. True Label "Cliff"

Case Study 7: Dining Table Classification - Figure 36 (Grad-CAM) Figure 37 (LIME)

The model predicted the image correctly and assigned a high probability to the "Dining Table" class. In this class, it is difficult to keep a single object featured in the image as dining tables usually have surrounding chairs. However, there are only three chair labels in the ImageNet-1k dataset, including "Barber Chair", "Folding Chair" and "Rocking Chair". None of these type are featured in the samples.

The two explainability methods highlight distinct regions of interest. Grad-CAM demonstrates a primary focus on the edges of the table, with moderate attention distributed across the surrounding chairs and the tabletop surface. In contrast, LIME shows a "miss" in the Pointing Game metric. Its top five positive segments focus only on the surrounding chairs. When visualizing the top 10 features, LIME only highlights table related features, which are the same corners and edges captured by Grad-CAM.

This suggests that the model uses the presence of multiple chairs as a strong contextual cue for the "Dining Table". The faithfulness graph shows that the deletion AUC and insertion AUC metrics for LIME are better, indicating that LIME interprets the model's decision-making logic more accurately than Grad-CAM.

This hypothesis is further supported by the sample in Figure 38, where a "Desk" is misclassified as a "Dining Table". All of the samples in the "Desk" class that are correctly predicted feature only one chair or none. Thus, the model appears to have learned a pattern where a "Desk" only has one chair at most.

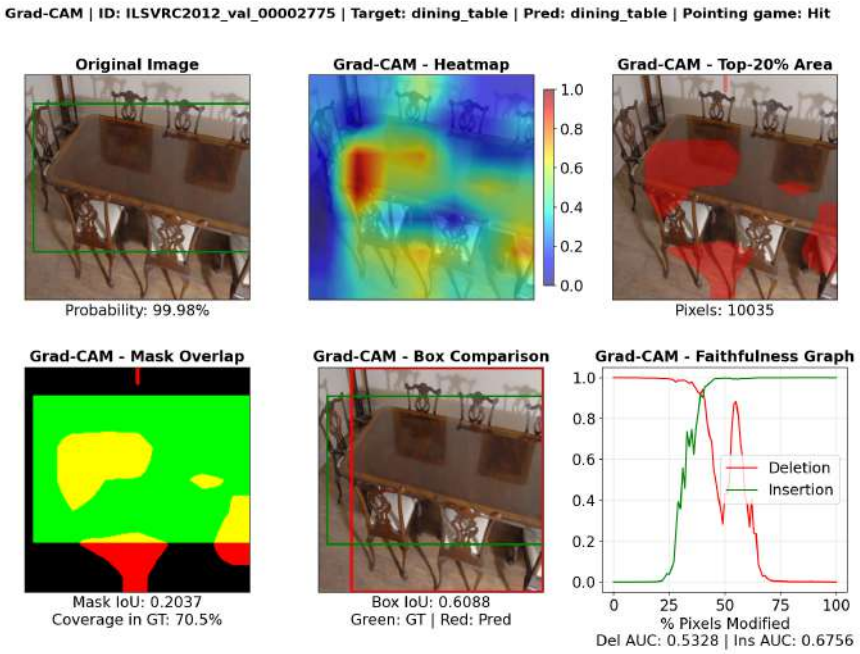


Figure 36: Grad-CAM explanation for correct prediction: "Dining Table".

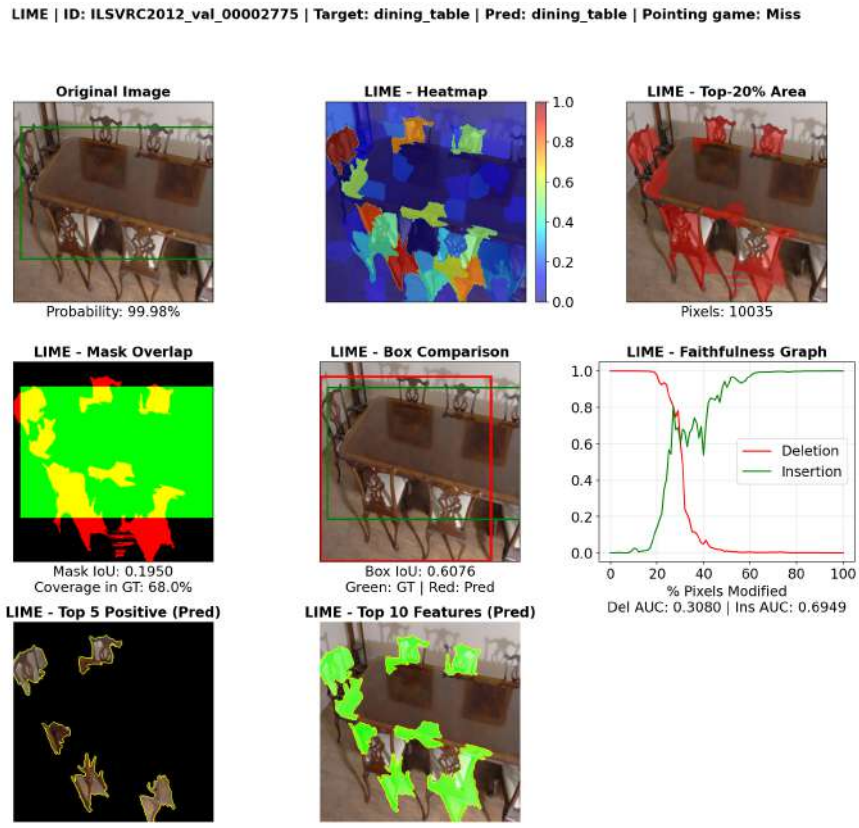


Figure 37: LIME explanation for correct prediction: "Dining Table".

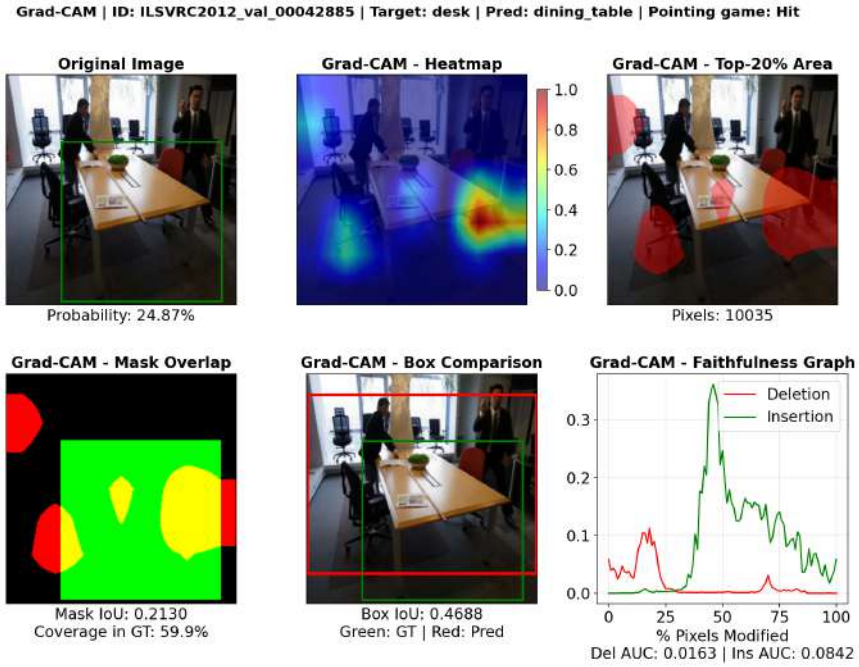


Figure 38: Grad-CAM: Predicted "Dining Table" vs. True Label "Desk"

Case Study 8: Rapeseed Misclassification - Figure 39 (Grad-CAM) Figure 40 (LIME)

The model misclassified the image as a "Container Ship" with a high probability of nearly 100%. This result provides an insight into model shortcuts and the inconsistency of explainability methods. Although the image contains no ship, both Grad-CAM and LIME provide different interpretations.

Grad-CAM highlights the yellow rapeseed field, resulting in a high Mask IoU and a Pointing Game "hit". However, this "success" is qualitatively a failure. It reveals that the model is using a color shortcut, where large areas of yellow are associated with features of a "Container Ship". This may be due to the visual similarities between colorful containers with the yellow of the rapeseed.

In contrast, LIME focuses on the containers, leading to a Pointing Game "Miss" and an extremely low Mask IoU. Only 0.4% of the top 20 % most important pixels lie within the ground truth bounding box. The focus on containers seems reasonable, but the Faithfulness Graph for LIME shows an extremely low Insertion AUC as well, indicating that these container segments do not actually drive the model's high confidence.

This comparison demonstrates that a high quantitative metrics can be misleading if the model is focusing on the correct area for the wrong reasons. It confirms that the model relies on correlations, specifically color, rather than a semantic understanding of the objects. The structural elements of the containers are insufficient to drive the model's high confidence prediction without the yellow background shortcut.

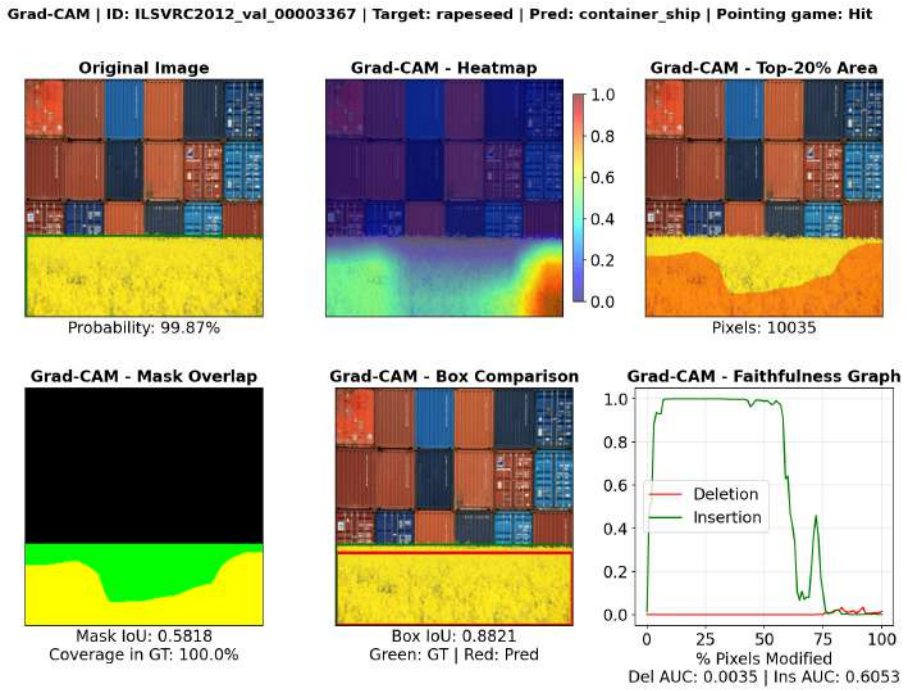


Figure 39: Grad-CAM: Predicted "Container Ship" vs. True Label "Rapeseed"

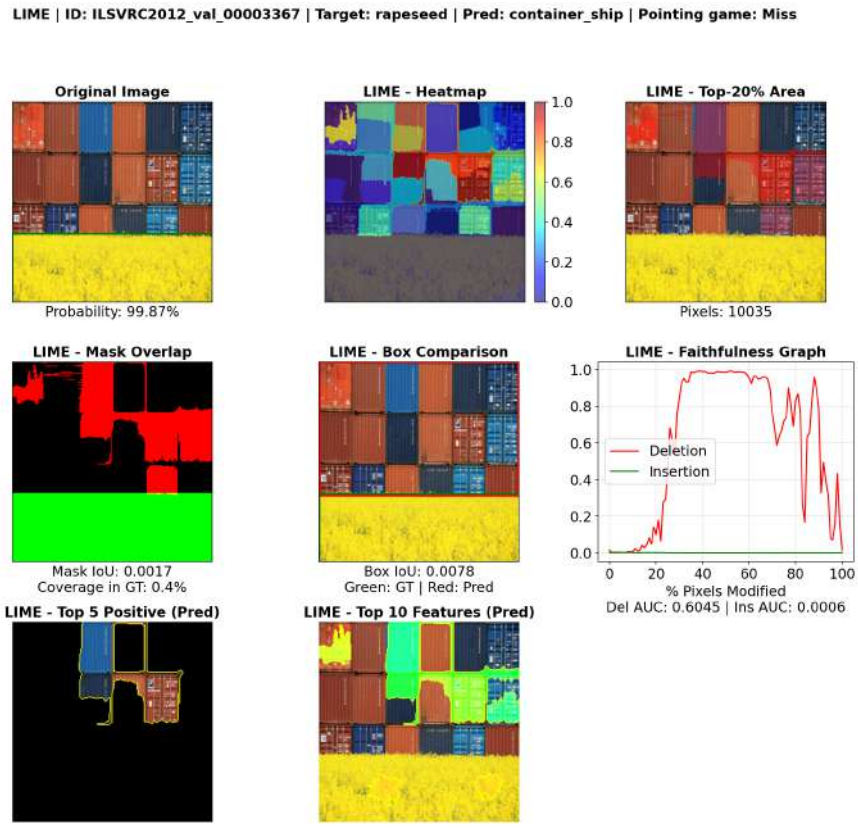


Figure 40: LIME: Predicted "Container Ship" vs. True Label "Rapeseed"

Case Study 9: Container Ship - Figure 41 (Grad-CAM) Figure 42 (LIME)

The model correctly predicted the class "Container Ship" with high confidence in Figure 42 and 43. However, a qualitative analysis of both Grad-CAM and LIME reveals a semantic shortcut. The model does not recognize the ship, but rather the stacked containers.

In Figure 42, LIME's top five positive segments focus solely on the rectangular blocks of the containers. This reliance on the containers as identification for the ship is further supported by Figure 43. In this sample, the shape of the ship is invisible from a top-view angle, only the containers are featured. Despite the absence of any visible ship structure, the model assigns a high probability to the "Container Ship" class.

This suggests the model has learned a shortcut, where the presence of the containers is sufficient evidence to predict a "Container Ship". While this leads to high quantitative accuracy, it indicates a lack of true semantic understanding of the target object. Furthermore, the Faithfulness Graphs for these correct predictions show high Insertion AUC scores, confirming that the containers are the main features contributing to the model's high confidence predictions.

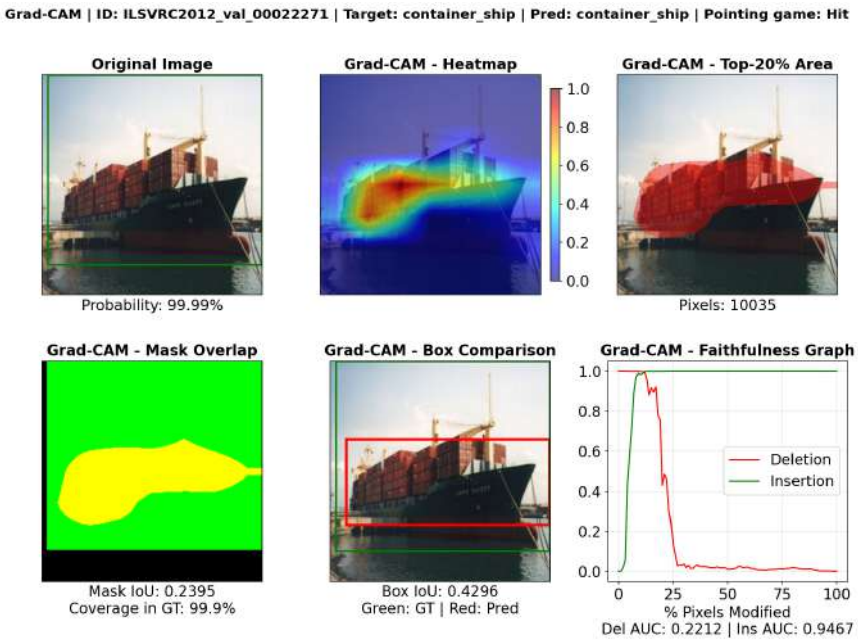


Figure 41: Grad-CAM explanation for correct prediction: "Container Ship".

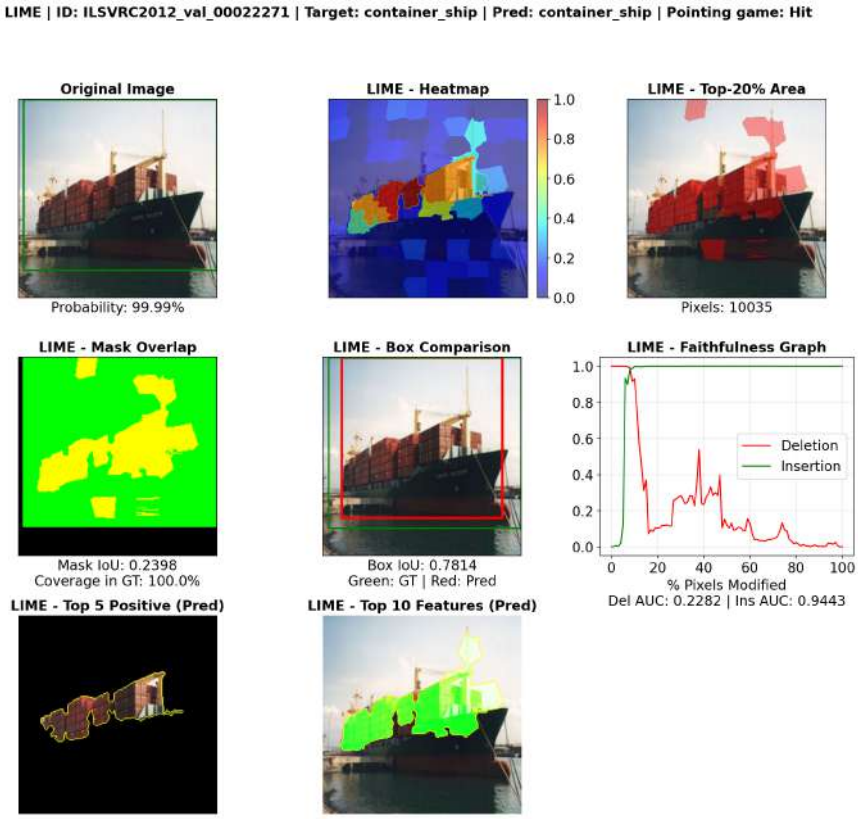


Figure 42: LIME explanation for correct prediction: "Container Ship".

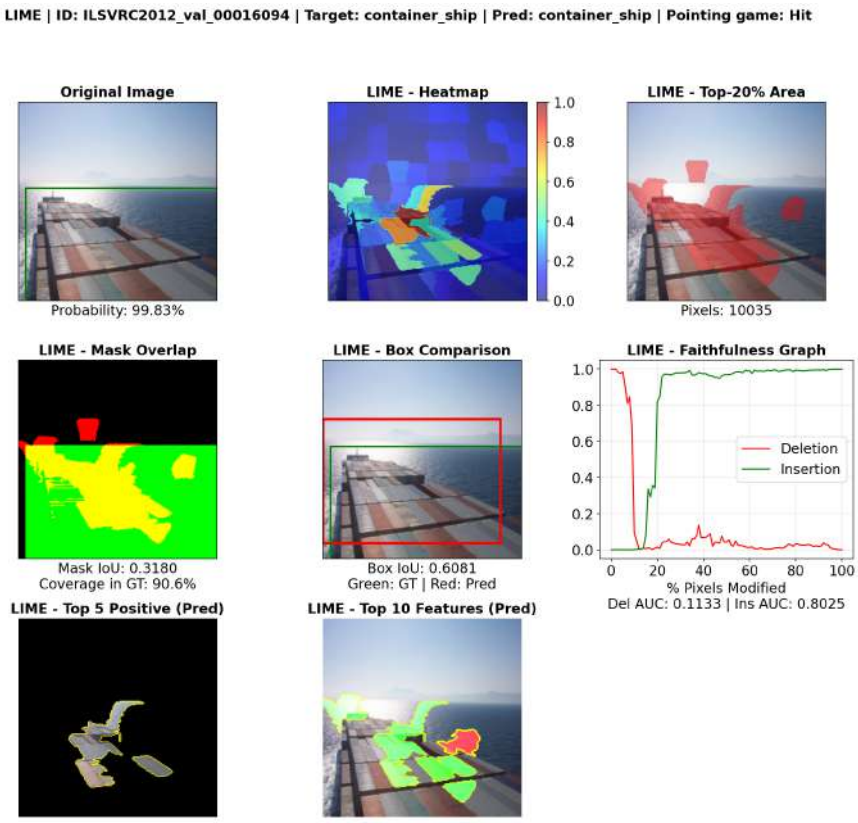


Figure 43: LIME explanation for correct prediction: "Container Ship". However, the ship is not visible in the image.

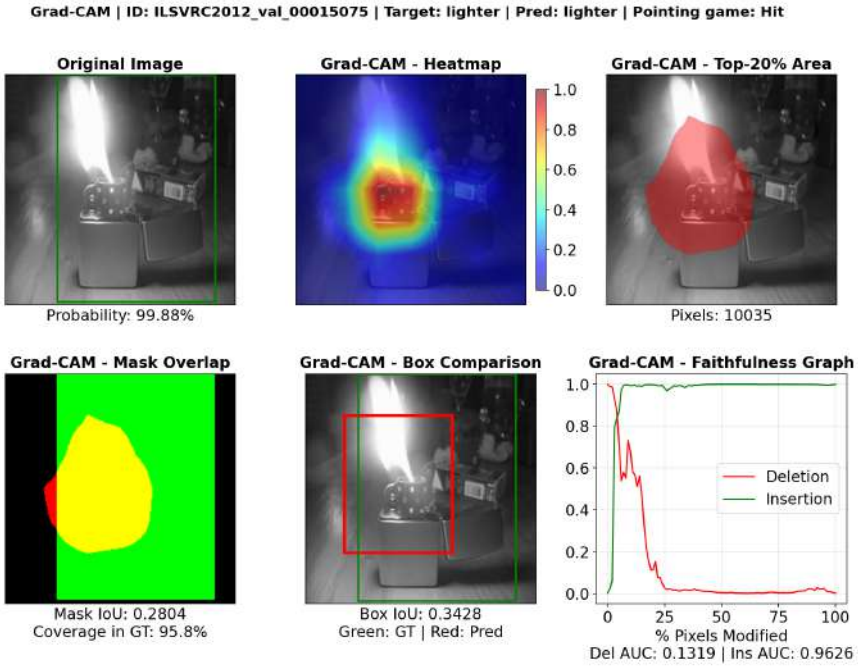


Figure 44: Grad-CAM highlights the sparkwheel as the most distinct feature of lighter.

5.3.2 Quantitative Evaluation

Table 2 and 3 show the results of quantitative evaluations on both Grad-CAM and LIME explainability methods. Table 2 shows the localization performance and Table 3 shows the classification performance of the model and faithfulness metrics of the visual explainability methods. The total mean is calculated as the sum of the values from each class divided by the total number of classes.

Table 2: Localization Accuracy Metrics

ID	Class Name (Samples)	Box IoU		Mask IoU		Pointing G.	
		G	L	G	L	G	L
1	Goldfish (17)	0.5461	0.4807	0.4188	0.2647	0.9412	0.8235
2	Great white shark (38)	0.6022	0.6343	0.3190	0.2900	1.0000	0.9474
3	Tiger shark (30)	0.6116	0.6098	0.3409	0.2845	1.0000	0.9000
4	Hammerhead (31)	0.6271	0.5345	0.4300	0.2968	1.0000	0.9032
92	Bee eater (43)	0.5418	0.4960	0.4211	0.2885	1.0000	0.7674
93	Hornbill (37)	0.5405	0.5715	0.3505	0.2786	0.9730	0.7838
129	Spoonbill (37)	0.6379	0.5969	0.3964	0.3060	1.0000	0.9459
130	Flamingo (22)	0.6572	0.6827	0.3001	0.2668	1.0000	0.9545
207	Golden retriever (32)	0.5690	0.6295	0.3386	0.3016	1.0000	0.9375
235	German shepherd (29)	0.5811	0.6203	0.3572	0.3089	1.0000	0.9655
250	Siberian husky (29)	0.5718	0.5840	0.3456	0.2850	1.0000	1.0000
270	White wolf (33)	0.5671	0.6118	0.3708	0.3192	1.0000	0.9697
281	Tabby (30)	0.5953	0.6873	0.2863	0.2696	1.0000	0.9667
283	Persian cat (29)	0.5134	0.6877	0.2686	0.2492	1.0000	1.0000
284	Siamese cat (22)	0.5300	0.6505	0.3027	0.2800	1.0000	0.9545
403	Aircraft carrier (37)	0.5696	0.6233	0.3404	0.2859	0.9730	0.8649
437	Beacon (17)	0.5693	0.3001	0.5274	0.2686	0.9412	0.7059
510	Container ship (33)	0.5711	0.5448	0.4164	0.2955	1.0000	0.9697
511	Convertible (37)	0.6449	0.6775	0.3119	0.3062	0.9459	0.9189
513	Cornet (14)	0.6085	0.4340	0.4959	0.3347	1.0000	0.7857
526	Desk (32)	0.5587	0.5318	0.2992	0.2325	0.9375	0.5625
532	Dining table (31)	0.5491	0.5805	0.3090	0.2415	0.9677	0.5806
566	French horn (18)	0.5093	0.4613	0.3975	0.3060	1.0000	1.0000
587	Hammer (21)	0.6123	0.6225	0.3643	0.2837	1.0000	0.9048
626	Lighter (27)	0.5646	0.4604	0.3920	0.2808	0.9630	0.7407
628	Liner (37)	0.5990	0.5927	0.3710	0.3212	1.0000	0.9459
671	Mountain bike (29)	0.5291	0.4495	0.3927	0.2716	0.8621	0.8276
698	Palace (29)	0.5811	0.5857	0.3514	0.3055	0.9655	1.0000
736	Pool table (30)	0.5983	0.4892	0.3822	0.2514	1.0000	0.8667
740	Power drill (24)	0.4911	0.4497	0.4013	0.2498	0.9167	0.8333
784	Screwdriver (10)	0.6149	0.4879	0.3633	0.2491	1.0000	0.7778
817	Sports car (33)	0.6203	0.5785	0.3866	0.3270	0.9697	0.8788
875	Trombone (27)	0.5937	0.5586	0.4057	0.3154	1.0000	0.9259
889	Violin (18)	0.5348	0.4206	0.4599	0.2843	1.0000	0.8889
949	Strawberry (8)	0.7431	0.5370	0.4049	0.3091	1.0000	1.0000
950	Orange (11)	0.6723	0.5107	0.4209	0.3438	1.0000	1.0000
951	Lemon (7)	0.7242	0.5804	0.5961	0.4647	1.0000	0.8571

Table 2 – Continued from previous page

ID	Class Name (Samples)	Box IoU		Mask IoU		Pointing G.	
		G	L	G	L	G	L
970	Alp (13)	0.6274	0.5398	0.3346	0.2057	1.0000	0.6154
972	Cliff (18)	0.6116	0.5757	0.3032	0.2779	0.9444	0.7778
979	Valley (28)	0.5468	0.5386	0.2176	0.2257	0.6071	0.5714
980	Volcano (38)	0.5180	0.4175	0.3956	0.2583	0.9211	0.7632
984	Rapeseed (25)	0.7863	0.5375	0.4761	0.2817	1.0000	0.5600
985	Daisy (11)	0.6227	0.6229	0.3667	0.3232	1.0000	1.0000
Total Mean		**0.5922**	**0.5532**	**0.3751**	**0.2881**	**0.9728**	**0.8591**

Note: G = Grad-CAM, L = LIME, Pointing G. = Pointing Game

Table 3: Classification Performance and Faithfulness Metrics

ID	Class Name (Samples)	Recall	Del AUC ↓		Ins AUC ↑	
			G	L	G	L
1	Goldfish (17)	0.7647	0.1943	0.2567	0.6799	0.6139
2	Great white shark (38)	0.7368	0.2715	0.3483	0.7014	0.6879
3	Tiger shark (30)	0.8000	0.1000	0.1155	0.5683	0.5203
4	Hammerhead (31)	0.8065	0.1776	0.3730	0.7011	0.6992
92	Bee eater (43)	0.9767	0.1570	0.1959	0.8894	0.8842
93	Hornbill (37)	0.8108	0.0995	0.1062	0.7402	0.5967
129	Spoonbill (37)	1.0000	0.1953	0.2681	0.8665	0.8235
130	Flamingo (22)	0.8636	0.2557	0.2730	0.7780	0.7110
207	Golden retriever (32)	0.7812	0.1215	0.1018	0.6191	0.5991
235	German shepherd (29)	0.8621	0.0953	0.0739	0.6627	0.6433
250	Siberian husky (29)	0.4138	0.0487	0.0624	0.2427	0.2389
270	White wolf (33)	0.8788	0.1431	0.1232	0.6481	0.6338
281	Tabby (30)	0.6667	0.0518	0.0454	0.3426	0.3124
283	Persian cat (29)	0.8621	0.1061	0.0869	0.6839	0.6562
284	Siamese cat (22)	0.9091	0.1788	0.1615	0.8380	0.7827
403	Aircraft carrier (37)	0.7568	0.1158	0.3225	0.6429	0.6490
437	Beacon (17)	0.8235	0.2050	0.5612	0.8178	0.7991
510	Container ship (33)	0.9697	0.1928	0.3163	0.9003	0.8421
511	Convertible (37)	0.6216	0.1033	0.1567	0.3884	0.4394
513	Cornet (14)	0.7143	0.0597	0.0719	0.5383	0.4620
526	Desk (32)	0.6250	0.1123	0.1335	0.3406	0.3543
532	Dining table (31)	0.8387	0.1836	0.1810	0.5374	0.4778
566	French horn (18)	0.8889	0.1103	0.0943	0.7182	0.6046
587	Hammer (21)	0.7143	0.0465	0.0335	0.3755	0.2455
626	Lighter (27)	0.7037	0.0913	0.1073	0.4750	0.4208
628	Liner (37)	0.8919	0.1909	0.2597	0.7640	0.7875
671	Mountain bike (29)	0.8276	0.1281	0.1374	0.6600	0.5868
698	Palace (29)	0.7931	0.1052	0.1274	0.5912	0.5618
736	Pool table (30)	1.0000	0.3290	0.4443	0.8782	0.8645

Table 3 – Continued from previous page

ID	Class Name (Samples)	Recall	Del AUC ↓		Ins AUC ↑	
			G	L	G	L
740	Power drill (24)	0.6250	0.0360	0.0576	0.4290	0.3112
784	Screwdriver (10)	0.4000	0.0140	0.0239	0.1950	0.1550
817	Sports car (33)	0.5152	0.1756	0.2716	0.5325	0.4889
875	Trombone (27)	0.7407	0.0435	0.0515	0.4712	0.4045
889	Violin (18)	0.8333	0.0393	0.0651	0.6471	0.5897
949	Strawberry (8)	1.0000	0.2945	0.2793	0.8710	0.8783
950	Orange (11)	0.9091	0.1795	0.1872	0.7573	0.7053
951	Lemon (7)	1.0000	0.1965	0.1364	0.9081	0.8814
970	Alp (13)	0.6154	0.2621	0.4344	0.5734	0.4994
972	Cliff (18)	0.7778	0.0993	0.1381	0.4209	0.4201
979	Valley (28)	0.7143	0.1554	0.2352	0.4458	0.3810
980	Volcano (38)	0.7632	0.1700	0.2521	0.6527	0.5980
984	Rapeseed (25)	0.9200	0.7429	0.8462	0.9420	0.8871
985	Daisy (11)	1.0000	0.3431	0.5004	0.9517	0.9154
Total Mean		**0.7934**	**0.1610**	**0.2097**	**0.6369**	**0.5957**

Note: G = Grad-CAM, L = LIME

Based on the results in Table 3, the model achieved a total mean recall of approximately 80% across 43 selected classes, showing that the model has a robust performance even though the dataset is imbalanced. When comparing the explainability methods in localization and faithfulness aspects, Grad-CAM generally performed better across all metrics.

Grad-CAM's mean Box IoU (0.5922) and Mask IoU (0.3751) are slightly better than LIME, which achieved scores of 0.5532 and 0.2881 respectively. This indicates that highlighted regions by Grad-CAM have a greater overlap with the ground truth bounding box. Thus, Grad-CAM focuses on the correct spatial region more accurately than LIME. However, LIME achieved higher IoU scores in most of the "living thing" classes. Thus, LIME is more effective in capturing the complex, non-rigid features characteristic of biological subjects.

For the Pointing Game metrics, Grad-CAM achieved an almost perfect score of 0.9728, which is also better than LIME with 0.8951. This metric checks whether the highest-weighted pixel lies within the annotated bounding box. Interestingly, both methods received low scores in the "Valley" class. This may be due to the images having many V-shaped outlines, since valleys exist between two mountains, this is the important feature for identifying a valley. This low score suggests that both methods did not highlight the annotated V-shape, but rather another part of the image.

In faithfulness metrics, a lower deletion AUC and a higher insertion AUC score are better. Grad-CAM achieved better performance in both metrics. This suggests

Grad-CAM reveals the model's actual decision-making process more accurately than LIME. Overall, Grad-CAM performs better localization and faithfulness in interpreting the model's decisions.

6 Discussion

In this work, visual explainability methods are evaluated using both qualitative and quantitative approaches to ensure a comprehensive evaluation.

Grad-CAM visualizes the explanation with a continuous color-coded heatmap overlaid on the input image. In contrast, LIME provides more precise attention by showing meaningful segments with clear boundaries. A clear physical part of an object that contributes toward the model's prediction can be identified. Thus, LIME enables the identification of fine-grained features, while Grad-CAM provides more generalized spatial regions. A LIME visualization is more intuitive when compared to Grad-CAM. However, the stability of a LIME explanation relies heavily on the chosen number of segments and the number of perturbed samples fed into the model.

In qualitative evaluations, when observing the visual results across all samples, the Grad-CAM and LIME methods are able to focus on the correct region in most scenarios. For mammal entities, both methods showed that the model has learned biological hierarchies and considers the face to be the most discriminative part. For bird and fish entities, the body shape and texture provide more information.

Color is an important feature in distinguishing between different species or breeds. This is because certain traits and characteristics can be found in only specific species. For example, the blue eyes of Siamese cat, the pink body of Flamingo.

Within "artifact" classes, both methods typically emphasize the most functional or structurally unique components or parts that are usually gripped. For instance, in the "Lighter" class, the sparkwheel is consistently identified as a key feature.

Sometimes, the model learns shortcuts. For example, all "Lighter" samples featuring a flame were correctly predicted. However, shortcuts may lead to incorrect predictions. For example, the model identifies a V-shape as a "Valley" because it focuses heavily on the converging lines of the background that it completely ignores the foreground context, specifically the vertical drop-off and stony texture of the cliff—which explains the 100% confidence in the wrong answer. Furthermore, pink birds in a "wings wide open" pose are consistently misidentified as "Spoonbills". The model has also only learned to recognize containers, leading it to classify them as a "Container Ship" even when there is no ship present, only flowers. Finally, the "Desk" class is expected to have at most one chair, which leads to misclassification as a "dining table" when more are present.

When the model assigns a high probability score, Grad-CAM shows a concentrated, high intensity "red blob" on the object, while LIME highlights influential segments in green directly on the object. In contrast, when the model has a low probability score, Grad-CAM may exhibit fragmented, high attribution areas. These spread

regions often incorporate the background into the classification and fail to localize the object correctly. Thus, we may assume that if Grad-CAM has "red blobs" spread across different areas, the model is specifically finding high contrast edges instead of focusing on the object. However, in misclassification cases, LIME may be able to capture the object features and focus on the object correctly, but it still fails to identify the object. Thus, the performance of these explainability methods is dependent on the model's confidence and the presence of distinctive features.

Moreover, the spatial composition of the data plays a significant role. If an image contains significant context, the model tends to focus on the center. However, target objects are not always centered and images containing multiple objects from different classes can lead to model misclassification or the misinterpretation of explainability results. Thus, the approach of focusing on single object images can prevent penalizing correct focus on correct class but the annotated class in another. These findings also reflect that the quality of data used is important when training or testing the models.

In quantitative analysis, Grad-CAM consistently outperforms LIME across all localization and faithfulness metrics. This suggests that a model-specific method is more effective in explanation compared to a model-agnostic method. Grad-CAM interprets the model based on the internal gradients and feature maps of the target class. Thus, Grad-CAM captures the exact information the model used. However, this is also its limitation as it is required to access to the model's internal architecture. In contrast, LIME is a model-agnostic approach and can be applied to any model. However, it is quantitatively less precise.

In our study, a low IoU score is not necessarily considered as poor performance. This reflects a mismatch between human-defined ground-truth box and the model's actual focus. While a human-annotated box covers the entire object, the model often relies on discriminative parts, such as the "sparkwheel" of a lighter or the face" of a mammal to make a prediction. Grad-CAM and LIME faithfully highlight these specific features rather than the whole object, causing the overlapping with the ground truth is reduced. Moreover, the coarse nature of Grad-CAM heatmaps and the superpixel boundaries in LIME often fail to align perfectly with the edges of the objects, further lowering the quantitative IoU even when the focus is qualitatively correct. Thus, we also include the Pointing Game metrics to check if the method hits the correct spot, rather than if it fills the entire mask.

Consequently, the evaluation of the interpretability of explainability methods should consider both qualitative and quantitative approaches to provide a complete picture of model behavior.

7 Conclusion

This work provided a comprehensive comparative evaluation of Grad-CAM and LIME to determine their effectiveness in explaining model decisions. We introduced an evaluation framework by integrating budget attention into IoU metrics, allowing for a more controlled and fair comparison between coarse heatmaps and segmented explanations.

The analysis of explainability methods across 42 ImageNet classes covered different subtrees shows that Grad-CAM and LIME are able to focus on the correct region in most scenarios. Both methods highlighted faces of mammals or functional parts as the most important features. However, our study also exposed the model learned shortcuts, where it learned the class "Container Ship" without learning the ship, but rather just the containers.

Grad-CAM consistently outperforms LIME across all metrics, achieving a significantly higher Pointing Game score (97.28%) and a lower Deletion AUC (16.10%). This suggests that model-specific methods using internal gradients are more faithful than model-agnostic approaches like LIME.

Future study could explore other model-agnostic methods which have better interpretability. By investigating these methods through the evaluation framework, researchers could determine if they perform better than model-specific methods like Grad-CAM.

In conclusion, the evaluation of the interpretability of explainability methods should consider both qualitative and quantitative approaches. Quantitative metrics provide a numerical comparison, while the qualitative analysis reveals whether the model is focusing on the correct features or using a shortcut. If different explainability methods highlight the same regions, it allows us to gain more trust in the model's prediction, ensuring that the decision is based on the actual object.

Bibliography

Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34 (11):2274–2282, November 2012. ISSN 1939-3539. doi: 10.1109/TPAMI.2012.120. URL https://ieeexplore.ieee.org/document/6205760/.

Abdurrahim Akgündoğdu and Şerife Çelikbaş. Explainable deep learning framework for brain tumor detection: Integrating LIME, Grad-CAM, and SHAP for enhanced accuracy. *Medical Engineering & Physics*, 144:104405, October 2025. ISSN 13504533. doi: 10.1016/j.medengphy.2025.104405. URL https://linkinghub.elsevier.com/retrieve/pii/S1350453325001249.

Anna Arias-Duart, Ferran Parés, Dario Garcia-Gasulla, and Victor Giménez-Ábalos. Focus! Rating XAI Methods and Finding Biases. In *2022 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–8, July 2022. doi: 10.1109/FUZZ-IEEE55066.2022.9882821. URL https://ieeexplore.ieee.org/document/9882821/.

Mike Bostock. ImageNet Hierarchy. https://observablehq.com/@mbostock/imagenet-hierarchy, 2018.

Quoc Hung Cao, Truong Thanh Hung Nguyen, Vo Thanh Khang Nguyen, and Xuan Phong Nguyen. A Novel Explainable Artificial Intelligence Model in Image Classification problem, July 2023. URL http://arxiv.org/abs/2307.04137. arXiv:2307.04137 [cs].

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009. doi: 10.1109/CVPR.2009.5206848. URL https://ieeexplore.ieee.org/document/5206848/.

Keerthi Devireddy. A Comparative Study of Explainable AI Methods: Model-Agnostic vs. Model-Specific Approaches, April 2025. URL http://arxiv.org/abs/2504.04276. arXiv:2504.04276 [cs].

Andrei Dugăeșescu and Adina Magda Florea. Evaluation and analysis of visual methods for CNN explainability: a novel approach and experimental study. *Neural Computing and Applications*, 37(20):14935–14970, July 2025. ISSN 1433-3058. doi: 10.1007/s00521-025-11282-7. URL https://doi.org/10.1007/s00521-025-11282-7.

Abdul Muiz Fayyaz, Said Jadid Abdulkadir, Noureen Talpur, Safwan Mahmood Al-Selwi, Shahab Ul Hassan, and Ebrahim Hamid Sumiea. Grad-CAM (Gradient-weighted Class Activation Mapping): A systematic literature review. *Computers in Biology and Medicine*, 198:111200, November 2025. ISSN 00104825. doi:

10.1016/j.compbiomed.2025.111200. URL https://linkinghub.elsevier.com/retrieve/pii/S0010482525015537.

Damien Garreau and Ulrike Luxburg. Explaining the Explainer: A First Theoretical Analysis of LIME. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, pages 1287–1296. PMLR, June 2020. URL https://proceedings.mlr.press/v108/garreau20a.html.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. IMAGENET-TRAINED CNNS ARE BIASED TOWARDS TEXTURE; INCREASING SHAPE BIAS IMPROVES ACCURACY AND ROBUSTNESS. 2019.

Margherita Grandini, Enrico Bagli, and Giorgio Visani. Metrics for Multi-Class Classification: an Overview, August 2020. URL http://arxiv.org/abs/2008.05756. arXiv:2008.05756 [stat].

Lav Kumar Gupta, Deepika Koundal, and Shweta Mongia. Explainable Methods for Image-Based Deep Learning: A Review. *Archives of Computational Methods in Engineering*, 30(4):2651–2666, May 2023. ISSN 1886-1784. doi: 10.1007/s11831-023-09881-5. URL https://doi.org/10.1007/s11831-023-09881-5.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, NV, USA, June 2016. IEEE. ISBN 978-1-4673-8851-1. doi: 10.1109/CVPR.2016.90. URL http://ieeexplore.ieee.org/document/7780459/.

Mirka Henninger and Carolin Strobl. Interpreting machine learning predictions with LIME and Shapley values: theoretical insights, challenges, and meaningful interpretations. *Behaviormetrika*, 52(1):45–75, January 2025. ISSN 1349-6964. doi: 10.1007/s41237-024-00253-2. URL https://doi.org/10.1007/s41237-024-00253-2.

Brendan Juba and Hai S. Le. Precision-Recall versus Accuracy and the Role of Large Data Sets. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):4039–4048, July 2019. ISSN 2374-3468. doi: 10.1609/aaai.v33i01.33014039. URL https://ojs.aaai.org/index.php/AAAI/article/view/5193.

Rojina Kashefi, Leili Barekatain, Mohammad Sabokrou, and Fatemeh Aghaeipoor. Explainability of vision transformers: a comprehensive review and new perspectives. *Multimedia Tools and Applications*, 85(2):115, February 2026. ISSN 1573-7721. doi: 10.1007/s11042-026-21313-7. URL https://doi.org/10.1007/s11042-026-21313-7.

Nikita Kisel, Illia Volkov, Katerina Hanzelkova, Klara Janouskova, and Jiri Matas. Flaws of ImageNet, Computer Vision's Favourite Dataset, November 2024. URL http://arxiv.org/abs/2412.00076. arXiv:2412.00076 [cs].

Hari Kishan Kondaveeti and Chinna Gopi Simhadri. Evaluation of deep learning models using explainable AI with qualitative and quantitative analysis for rice leaf disease detection. *Scientific Reports*, 15(1):31850, August 2025. ISSN 2045-2322. doi: 10.1038/s41598-025-14306-3. URL https://www.nature.com/articles/s41598-025-14306-3.

B. La Rosa, G. Blasilli, R. Bourqui, D. Auber, G. Santucci, R. Capobianco, E. Bertini, R. Giot, and M. Angelini. State of the Art of Visual Analytics for eXplainable Deep Learning. *Computer Graphics Forum*, 42(1):319–355, 2023. ISSN 1467-8659. doi: 10.1111/cgf.14733. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14733. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.14733.

Felipe Tomazelli Lima and Vinicius M.A. Souza. A Large Comparison of Normalization Methods on Time Series. *Big Data Research*, 34:100407, November 2023. ISSN 22145796. doi: 10.1016/j.bdr.2023.100407. URL https://linkinghub.elsevier.com/retrieve/pii/S2214579623000400.

Scott M Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html.

Antonio Mastroianni and Sibylle D Sager-Müller. Validation of ml models from the field of xai for computer vision in autonomous driving. In *xAI (Late-breaking Work, Demos, Doctoral Consortium)*, pages 185–192, 2024.

Christoph Molnar. *Interpretable machine learning: a guide for making black box models explainable*. Christoph Molnar, Munich, Germany, third edition edition, 2025. ISBN 978-3-911578-03-5. URL https://christophm.github.io/interpretable-ml-book. OCLC: 1518801363.

Christoph Molnar, Giuseppe Casalicchio, and Bernd Bischl. Interpretable Machine Learning – A Brief History, State-of-the-Art and Challenges. volume 1323, pages 417–431. 2020. doi: 10.1007/978-3-030-65965-3_28. URL http://arxiv.org/abs/2010.09337. arXiv:2010.09337 [stat].

Christoph Molnar, Gunnar König, Bernd Bischl, and Giuseppe Casalicchio. Model-agnostic Feature Importance and Effects with Dependent Features – A Conditional Subgroup Approach. *Data Mining and Knowledge Discovery*, 38(5):2903–2941, September 2024. ISSN 1384-5810, 1573-756X. doi: 10.1007/s10618-022-00901-9. URL http://arxiv.org/abs/2006.04628. arXiv:2006.04628 [stat].

Md Imran Nazir, Afsana Akter, Md Anwar Hussen Wadud, and Md Ashraf Uddin. Utilizing customized CNN for brain tumor prediction with explainable AI. *Heliyon*, 10(20):e38997, October 2024. ISSN 24058440. doi: 10.1016/j.heliyon.2024.e38997. URL https://linkinghub.elsevier.com/retrieve/pii/S2405844024150281.

Chung Hou Ng, Hussain Sadiq Abuwala, and Chern Hong Lim. Towards More Stable LIME For Explainable AI. In *2022 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, pages 1–4, November 2022. doi: 10.1109/ISPACS57703.2022.10082810. URL https://ieeexplore.ieee.org/document/10082810/.

Vitali Petsiuk, Abir Das, and Kate Saenko. RISE: Randomized Input Sampling for Explanation of Black-box Models, September 2018. URL http://arxiv.org/abs/1806.07421. arXiv:1806.07421 [cs].

Oona Rainio, Jarmo Teuho, and Riku Klén. Evaluation metrics and statistical tests for machine learning. *Scientific Reports*, 14(1):6086, March 2024. ISSN 2045-2322. doi: 10.1038/s41598-024-56706-x. URL https://www.nature.com/articles/s41598-024-56706-x.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144, San Francisco California USA, August 2016. ACM. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939778. URL https://dl.acm.org/doi/10.1145/2939672.2939778.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge, January 2015. URL http://arxiv.org/abs/1409.0575. arXiv:1409.0575 [cs].

Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization. pages 618–626, 2017. URL https://openaccess.thecvf.com/content_iccv_2017/html/Selvaraju_Grad-CAM_Visual_Explanations_ICCV_2017_paper.html.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for Simplicity: The All Convolutional Net, April 2015. URL http://arxiv.org/abs/1412.6806. arXiv:1412.6806 [cs].

Huawei Sun, Lorenzo Servadei, Hao Feng, Michael Stephan, Avik Santra, and Robert Wille. Utilizing Explainable AI for improving the Performance of Neural Networks. In *2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1775–1782, Nassau, Bahamas, December 2022. IEEE. ISBN 978-1-6654-6283-9. doi: 10.1109/ICMLA55696.2022.00271. URL https://ieeexplore.ieee.org/document/10069558/.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic Attribution for Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3319–3328. PMLR, July 2017. URL https://proceedings.mlr.press/v70/sundararajan17a.html.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Andrew Ilyas, and Aleksander Madry. From ImageNet to Image Classification: Contextualizing Progress on Benchmarks, May 2020. URL http://arxiv.org/abs/2005.11295. arXiv:2005.11295 [cs].

Paschalis Tsirtsakis, Georgios Zacharis, George S. Maraslidis, and George F. Fragulis. Deep learning for object recognition: A comprehensive review of models and algorithms. *International Journal of Cognitive Computing in Engineering*, 6: 298–312, December 2025. ISSN 26663074. doi: 10.1016/j.ijcce.2025.01.004. URL https://linkinghub.elsevier.com/retrieve/pii/S266630742500004X.

Francisca Chibugo Udegbe, Ogochukwu Roseline Ebulue, Charles Chukwudalu Ebulue, and Chukwunonso Sylvester Ekesiobi. Ai's impact on personalized medicine: Tailoring treatments for improved health outcomes. *Engineering Science & Technology Journal*, 5(4):1386–1394, 2024.

Francesco Ventura, Salvatore Greco, Daniele Apiletti, and Tania Cerquitelli. Explaining deep convolutional models by measuring the influence of interpretable features in image classification. *Data Mining and Knowledge Discovery*, 38(5): 3169–3226, September 2024. ISSN 1573-756X. doi: 10.1007/s10618-023-00915-x. URL https://doi.org/10.1007/s10618-023-00915-x.

Qiyang Wan, Chengzhi Gao, Ruiping Wang, and Xilin Chen. A Survey on Interpretability in Visual Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–20, 2026. ISSN 1939-3539. doi: 10.1109/TPAMI.2026.3672629. URL https://ieeexplore.ieee.org/document/11427331/.

Ross Wightman. Pytorch image models. https://github.com/huggingface/pytorch-image-models, 2019.

Ross Wightman, Hugo Touvron, and Hervé Jégou. ResNet strikes back: An improved training procedure in timm, October 2021. URL http://arxiv.org/abs/2110.00476. arXiv:2110.00476 [cs].

Yongjun Xu, Xin Liu, Xin Cao, Changping Huang, Enke Liu, Sen Qian, Xingchen Liu, Yanjun Wu, Fengliang Dong, Cheng-Wei Qiu, Junjun Qiu, Keqin Hua, Wentao Su, Jian Wu, Huiyu Xu, Yong Han, Chenguang Fu, Zhigang Yin, Miao Liu, Ronald Roepman, Sabine Dietmann, Marko Virta, Fredrick Kengara, Ze Zhang, Lifu Zhang, Taolan Zhao, Ji Dai, Jialiang Yang, Liang Lan, Ming Luo, Zhaofeng Liu, Tao An, Bin Zhang, Xiao He, Shan Cong, Xiaohong Liu, Wei Zhang, James P. Lewis, James M. Tiedje, Qi Wang, Zhulin An, Fei Wang, Libo Zhang, Tao Huang, Chuan Lu, Zhipeng Cai, Fang Wang, and Jiabao Zhang. Artificial intelligence: A powerful paradigm for scientific research. *The Innovation*, 2(4): 100179, November 2021. ISSN 26666758. doi: 10.1016/j.xinn.2021.100179. URL https://linkinghub.elsevier.com/retrieve/pii/S2666675821001041.

Han Chel Yoon and Lih Poh Lin. Brain Tumor Classification in MRI: Insights From LIME and Grad-CAM Explainable AI Techniques. *IEEE Access*, 13:154172–154202, 2025. ISSN 2169-3536. doi: 10.1109/ACCESS.2025.3603272. URL https://ieeexplore.ieee.org/document/11142756/.

Jianming Zhang, Sarah Adel Bargal, Zhe Lin, Jonathan Brandt, Xiaohui Shen, and Stan Sclaroff. Top-Down Neural Attention by Excitation Backprop. *International Journal of Computer Vision*, 126(10):1084–1102, October 2018. ISSN 1573-1405. doi: 10.1007/s11263-017-1059-x. URL https://doi.org/10.1007/s11263-017-1059-x.

Yifei Zhang, James Song, Siyi Gu, Tianxu Jiang, Bo Pan, Guangji Bai, and Liang Zhao. Saliency-Bench: A Comprehensive Benchmark for Evaluating Visual Explanations. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pages 5924–5935, August 2025. doi: 10.1145/3711896.3737414. URL http://arxiv.org/abs/2310.08537. arXiv:2310.08537 [cs].

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning Deep Features for Discriminative Localization. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2921–2929, Las Vegas, NV, USA, June 2016. IEEE. ISBN 978-1-4673-8851-1. doi: 10.1109/CVPR.2016.319. URL http://ieeexplore.ieee.org/document/7780688/.

Declaration of Authorship

Ich erkläre hiermit gemäß §9 Abs. 12 APO, dass ich die vorstehende Abschlussarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Des Weiteren erkläre ich, dass die digitale Fassung der gedruckten Ausfertigung der Abschlussarbeit ausnahmslos in Inhalt und Wortlaut entspricht und zur Kenntnis genommen wurde, dass diese digitale Fassung einer durch Software unterstützten, anonymisierten Prüfung auf Plagiate unterzogen werden kann.

BAMBERG, 31.03.2026
Place, Date


Signature