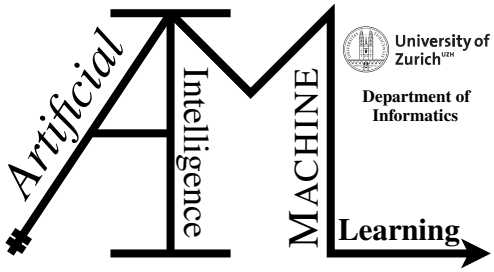
Visualization of Deep Features with Grad-CAM and LOTS

Master Thesis

Maximilian Weber
15-611-130

Submitted on
June 01 2023

Thesis Supervisor
Prof. Dr. Manuel Günther



Master Thesis

Author: Maximilian Weber, maximilian.weber@uzh.ch

Project period: 01.12.2022 - 01.06.2023

Artificial Intelligence and Machine Learning Group
Department of Informatics, University of Zurich

Acknowledgements

First and foremost, I want to express my sincere gratitude to Prof. Dr. Manuel Günther. Your availability, guidance, and valuable feedback have been instrumental in shaping the outcome of this thesis. Your willingness to address my questions and provide insightful suggestions has been invaluable. Thank you for granting me the freedom to explore and express my ideas fully. Our discussions have been stimulating and informative, and I am immensely grateful for the opportunity to learn from you.
Secondly, I would like to thank my girlfriend, family, and friends for their unwavering belief in me. Your support and patience have meant the world to me.
Finally, to all those who have played a part in supporting me during this work, whether through direct assistance, emotional support, or simply being there to listen, I extend my deepest appreciation. Your contributions have made a significant impact on my journey, and I am truly grateful for your presence in my life. Thank you!

Abstract

Deep learning models, particularly Convolutional Neural Networks (CNNs), have achieved remarkable success in image classification tasks. However, their lack of interpretability raises concerns about their trustworthiness, especially in high-risk domains like healthcare. To improve transparency, Explainable Artificial Intelligence (XAI) techniques have been developed. This thesis has a primary focus on expanding the Layerwise Origin Target Synthesis (LOTS) method, which is originally designed as a technique for generating adversarial images, to incorporate visualization capabilities. The aim is to address the limitations observed in current CAM-based visualization techniques that only offer broad area visualizations. The research explores methods for evaluating and comparing visualization techniques in the absence of a standard evaluation metric framework. Additionally, it investigates the applicability of the extended LOTS visualization technique to classes not present in the training dataset. Based on our findings, the LOTS visualization algorithm we propose, generates more focused visualizations that do not require explicit class specification, thereby also serving as a valuable tool for evaluating image quality within a training set. Furthermore, by adjusting the size of the Gaussian blur filter, it is possible to highlight fine locations in an image. Moreover, we demonstrate the potential for extending the LOTS algorithm to classes not included in the training dataset, although further research is required for validation. Lastly, we emphasize the importance of a standardized evaluation metrics framework.

Zusammenfassung

Deep-Learning-Modelle, insbesondere "Convolutional Neural Networks" (CNNs), haben beachtlichen Erfolg bei der Klassifizierung von Bildern erzielt. Jedoch werfen ihre mangelnde Interpretierbarkeit Bedenken hinsichtlich ihrer Verlässlichkeit und Vertrauenswürdigkeit auf, insbesondere in hochriskanten Bereichen wie dem Gesundheitswesen. Um die Transparenz zu verbessern, wurden Techniken der erklärungsfähigen künstlichen Intelligenz (XAI) entwickelt. Diese Arbeit hat einen Schwerpunkt auf der Erweiterung der "Layerwise Origin Target Synthesis" (LOTS) Methode, die ursprünglich als Technik zur Generierung von "adversarial" Bildern entwickelt wurde, um Visualisierungsfähigkeiten. Ziel ist es, die beobachteten Einschränkungen der gängigen CAM-basierten Visualisierungstechniken, die nur allgemeine Flächenvisualisierungen bieten, zu überwinden. Die Forschung untersucht Methoden zur Evaluierung und Vergleich von Visualisierungstechniken in Abwesenheit eines standardisierten Evaluierungsmetriken-Frameworks. Zusätzlich wird die Anwendbarkeit der erweiterten LOTS-Visualisierungstechnik auf Klassen, die nicht im Trainingsdatensatz vorhanden sind, untersucht. Basierend auf unseren Erkenntnissen generiert der von uns vorgeschlagene LOTS-Visualisierungsalgorithmus fokussiertere Visualisierungen, die keine explizite Klassenspezifikation erfordern und daher auch als wertvolles Werkzeug zur Bewertung der Bildqualität innerhalb eines Trainingsdatensatzes dienen. Darüber hinaus ist es möglich, durch Anpassung der Grösse des Gauss'schen Weichzeichnungsfilters den Grad der Pixel-Fokussierung in den Visualisierungen zu erhöhen. Zudem zeigen wir das Potenzial zur Erweiterung des LOTS-Algorithmus auf Klassen, die nicht im Trainingsdatensatz enthalten sind, wobei jedoch weitere Forschung zur Validierung erforderlich ist. Ausserdem betonen wir die Bedeutung eines standardisierten Evaluierungsmetriken-Frameworks.

Contents

Chapter 1

Introduction

Deep networks have greatly improved machine learning and have been particularly effective in image classification tasks. Convolutional Neural Networks (CNNs) have become increasingly popular, particularly due to larger training sets with millions of labeled examples, advancements in GPU technology facilitating the training of large models, and the implementation of better model regularization strategies (Zeiler and Fergus, 2014).

However, the lack of interpretability and understanding of deep learning models is a highly debated topic, especially in high-risk industries such as healthcare. These models are often referred to as Black Boxes due to the difficulty in explaining how they arrive at their conclusions. This raises concerns about the reliability and trustworthiness of their outputs.

In the case of a deep learning model designed to detect cancerous tumors, the model may indicate a 99% likelihood of detecting cancer, but it cannot provide a clear explanation for its decision-making process. This lack of transparency raises critical questions: Was the model's decision based on significant indicators in the MRI scan, or a misinterpretation of irrelevant features? This uncertainty poses a life-or-death situation for the patient, and doctors cannot afford to make errors.

The risks linked to using opaque models in significant decision-making procedures have sparked a trend towards developing more techniques that improve transparency, known as Explainable Artificial Intelligence (XAI). Many of them are based on the Class Activation Mapping (CAM) technique (Zhou et al., 2016). However, most of these methods provide only a coarse region of activation, while it is very likely that deep learning models usually consider much finer regions in images for their classification decision.

On the other hand, very precise localization can be created with the Layerwise Origin Target Synthesis (LOTS) developed by Rozsa et al. (2017), which is actually a technique to generate adversarial images (Szegedy et al., 2014). Particularly, LOTS can be used to explain differences in feature comparison tasks (Rozsa et al., 2017), which other techniques cannot. Our expectation is that the perturbations made to the feature vectors of the original image, as compared to the adversarial image, will target class-specific pixels, thus laying the groundwork for a visualization technique.

The goal of this thesis is to expand the capabilities of the LOTS method to include visualization. This involves exploring methods to evaluate existing visualization techniques and making them comparable to the extended LOTS method. To evaluate the proposed LOTS visualization technique, it is necessary to explore methods for evaluating and comparing existing visualization techniques. One of the major challenges in this field is the lack of a standard evaluation metric for visual explanations. While there are several metrics that have been proposed in the literature, they often focus on specific aspects of the visualizations and may not capture the full range of properties that are important for a good visualization. Additionally, different visualization techniques may have different strengths and weaknesses depending on the dataset, the model, and

the specific task at hand. Moreover, we evaluate whether the LOTS visualization can also be applied to classes which are not present in a dataset. This is in contrast to CAM-based visualization techniques, which are limited to visualizing only classes present in the training dataset and cannot be easily extended to new or unseen classes.

The research focuses on addressing the following questions:

RQ1: How can the layerwise origin-target synthesis (LOTS) method be extended to support visualizations?

- RQ1.1: How can the LOTS method be extended to generate visualizations with larger areas, similar to CAM-based methods?
- RQ1.2: How can the LOTS method be extended to highlight fine locations in an image?

RQ2: How can a visualization be evaluated?

RQ3: Can the extended LOTS visualization technique also be applied to classes which are not present in a training dataset?

The thesis is structured as follows: In Chapter 2, we delve into related research and the current status of Image Classification using CNNs, as well as Visualization Techniques for Feature Visualization and Attribution. Chapter 3 presents background information on the LOTS algorithm and different evaluation metrics, which serves as a crucial foundation for the thesis. Chapter 4 highlights the dataset selection, model choice, and LOTS visualization algorithm. Chapter 5 presents the Quantitative and Qualitative Analysis. In Chapter 6, the findings are examined, revealing that LOTS excels in capturing details qualitatively. The chapter also explores the utilization of LOTS as a quality assessing tool and emphasizes the need for consensus on unified metrics for future research in this area. Chapter 7 concludes the work while outlining potential future directions.

Chapter 2

Related Work

This chapter explores the history and present status of research in image classification. Furthermore, it investigates the background of several visualization methods.

2.1 Image Classification and CNNs

Computer vision involves translating images and videos into signals that machines can comprehend, enabling programmers to control their behavior based on a higher level of understanding (Klette, 2014). Image classification is a fundamental computer vision task, with numerous real-world applications like Google Photo's tagging and AI content moderation, and it paves the way for more advanced vision tasks such as object detection and video understanding.

Image classification, also known as Image Recognition, is the process of analyzing an image and assigning it to one of several pre-defined categories (Krishna et al., 2018). These problems have gained popularity due to its wide range of applications. For instance, self-driving cars rely heavily on rapid and accurate image classification as a crucial element (Fujiyoshi et al., 2019). Similarly, social media platforms such as Facebook and Google Photos utilize image classification to provide their users with a personalized and enhanced experience, often by applying transfer-learning approaches instead of end-to-end classification solutions. However, until recently, there has been a lack of research focused on developing algorithms that are capable of accurately handling unknown samples, which is commonly referred to as Open-Set Classification (OSC) (Palechor et al., 2023). OSC addresses the challenge of classifying samples that do not belong to any of the known predefined categories. In traditional image classification tasks, the assumption is that the test samples belong to one of the known categories seen during training. However, in real-world scenarios, there is often a need to identify samples that fall outside the known categories or to have a mechanism to reject ambiguous or novel samples.

The emergence of image classification can be attributed to the development of Artificial Neural Networks (ANN) and, subsequently, CNNs. In 1958, Frank Rosenblatt, a psychologist, developed the Perceptron, which was the first ANN designed to model the human brain's processing of data and its ability to learn object recognition (Rosenblatt, 1958). In 1998, LeNet was introduced, representing a significant milestone in this ongoing research (LeCun et al., 1998). As the first practical implementation of a CNN for image classification, LeNet established a foundation that would inspire future breakthroughs in the field. It used classical CNN techniques such as pooling layers, fully connected layers, padding, and activation layers to extract features and make classifications, achieving 99.05% accuracy on the MNIST test set. CNNs are particularly well-suited for image and speech recognition tasks due to their built-in convolutional layer. This layer reduces the complexity of high-dimensional images while retaining their crucial features, making CNNs an ideal solution for these applications. The convolutional layer is a critical component of CNNs,

as its learnable filters (or kernels) enable the network to extract important features from input data (LeCun et al., 1998). convolutional filters used in CNNs are typically small in terms of their spatial dimensions (e.g., 3x3 or 5x5) but span the entire depth or number of channels in the input. When data passes through a convolutional layer, each filter is convolved across the input's spatial dimensions, generating a 2D activation map. The network learns kernels that detect specific features at different spatial positions in the input by computing the scalar product for each value in the kernel as it moves over the input. These learned features are commonly referred to as activations and play a crucial role in image classification and XAI.

The ImageNet challenge is one of the most widely used benchmarks for evaluating CNNs, especially for image classification tasks. However, there are other datasets and evaluation metrics used to evaluate CNNs, depending on the specific task and application. The challenge has evolved since the introduction of AlexNet by Krizhevsky et al. (2012). AlexNet was revolutionary for several reasons. First, it was a deep neural network with eight layers, which was much deeper than any previous successful image recognition model. Second, it utilized a new activation function called the rectified linear unit (ReLU), which helped to avoid the vanishing gradient problem that had limited the depth of previous neural networks. Third, it used a technique called data augmentation, where the input images were randomly transformed during training, to prevent overfitting and improve generalization performance. Fourth, it was trained on a large dataset of over one million images, which was much larger than any previous image recognition dataset. Subsequently, researchers have introduced several modifications to the architecture, resulting in improved performance. GoogLeNet/Inception, introduced the inception module, which allows efficient use of computation and parameterization by having multiple filter sizes in a single layer (Szegedy et al., 2015). This enables training of much deeper networks. They achieved better performance than AlexNet by significantly reducing the number of parameters involved. In the same year, Simonyan and Zisserman (2015) presented VGGNet, which achieved good performance due to the network's depth. It has a simple structure with a series of small filters that are stacked on top of each other. They concluded that increasing the network depth improves performance in image recognition tasks. Later, He et al. (2016) introduced ResNet, which utilizes skip connections and batch normalization to increase accuracy while maintaining depth. DenseNet and SENet (Squeeze-and-Excitation Networks) are two other notable advancements in the field of CNNs that have made significant contributions to image classification tasks. DenseNet, introduced by Huang et al. (2017), stands out for its dense connectivity pattern. Unlike traditional CNNs where layers are sequentially connected, DenseNet adopts a densely connected structure, where each layer is connected to all preceding layers in a feed-forward manner. However, it is important to note that DenseNet as a whole consists of several dense blocks, and the connectivity pattern is specific to each dense block. This dense connectivity enhances feature reuse and facilitates better gradient propagation, allowing information to flow more efficiently throughout the network. By exploiting these connections, DenseNet achieves better accuracy and parameter efficiency while reducing the risk of overfitting. SENet, proposed by Hu et al. (2018), focuses on the modeling of channel-wise dependencies. It introduces a mechanism that adaptively recalibrates the channel-wise features of a network. By learning to selectively emphasize informative features and suppress less useful ones, SENet improves the representational power of CNNs. This attention mechanism allows the network to dynamically adjust the importance of different channels based on their relevance to the task at hand.

Furthermore, recent advancements in the field of image classification have witnessed the emergence of several notable models that specifically emphasize computational efficiency alongside performance. These models have made significant contributions to addressing the challenges of deploying deep learning models on resource-constrained devices and optimizing computational resources. MobileNetV2, proposed by Sandler et al. (2018), introduced Inverted Residuals and Linear Bottlenecks to enhance efficiency and accuracy in mobile-friendly models. It achieved

a good balance between accuracy and efficiency, suitable for resource-constrained devices. Building upon this, MobileNetV3, proposed by Howard et al. (2019), employed neural architecture search to automatically discover efficient model designs. Another notable approach is EfficientNet and later EfficientNetV2, which focused on designing network design spaces (Tan and Le, 2019, 2021). EfficientNetV2 employed compound scaling to optimize depth, width, and resolution simultaneously, achieving state-of-the-art accuracy while maintaining computational efficiency. RegNet, proposed by Radosavovic et al. (2020), introduced a novel approach to designing models by focusing on improving scalability and efficiency. The key idea behind RegNet is to decouple model design from computational resources, allowing for scalable model architectures that can be efficiently trained across a wide range of resource constraints.

Another noteworthy advancement in the field of image classification is the Vision Transformer (ViT) architecture (Dosovitskiy et al., 2021). Unlike traditional CNNs, ViT utilizes a transformer-based architecture, originally proposed for natural language processing tasks. The transformer architecture in ViT allows for capturing global relationships and long-range dependencies in the image, enabling effective modeling of both local and global information. This approach eliminates the need for handcrafted feature engineering and empowers the model to learn meaningful representations directly from raw image data. Swin Transformer, introduced by Liu et al. (2021), builds upon the success of ViT and proposes a novel hierarchical architecture that combines local and global attention mechanisms. MaxViT, proposed by Tu et al. (2022), further extends the capabilities of ViT by introducing a novel attention mechanism called Max-Attention. MaxViT leverages the strengths of both ViT and CNNs by incorporating max-pooling operations within the self-attention mechanism. Contrasting with the previous approach, ConvNext, by Liu et al. (2022), leverages on Swin Transformers while reintroducing ConvNet principles. This integration of ConvNet priors effectively enables Transformers to serve as a versatile visual backbone, showcasing exceptional performance across a diverse range of visual tasks.

Ongoing research in the field of image classification continues to advance the development of various models. Figure 2.1 provides a comprehensive overview of the performance of different models, including those that have not been specifically addressed in this discussion. In this context, Russakovsky et al. (2015) conducted a study to estimate the classification error made by humans in the ILSVRC dataset, contributing valuable insights to the comparison and evaluation of different model performances.

2.2 Visualization Techniques

Visualization techniques are an essential component of deep learning for image classification, as they provide insight into the behavior and performance of CNNs. Neural networks are often referred to as Black Boxes, and visualization methods are key to understanding how these models make predictions based on image data. CNNs predict by processing the input data through multiple layers with learned weights and non-linear transformations, involving millions of mathematical operations in a single prediction. Humans cannot follow the exact mapping from data input to prediction due to the complex interactions of millions of weights. Visualization techniques, therefore, help understand the decision-making process of a neural network, revealing which features of an image are used to make predictions. A range of visualization methods has been developed to uncover the learned features and decision-making processes of deep learning models, leading to improved performance and a deeper understanding of neural networks.

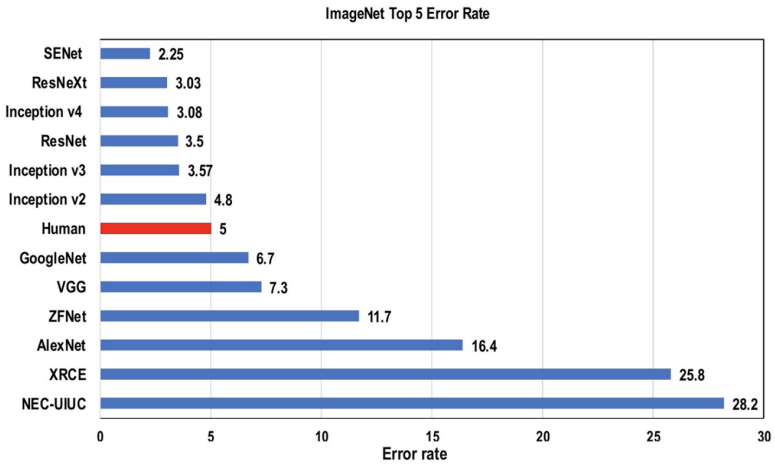


Figure 2.1: DEEP LEARNING PERFORMANCE COMPARED TO HUMAN. *The figure illustrates the performance comparison of 12 deep learning architectures on the y-axis, sorted by their Top 5 error rate. The x-axis represents the Top 5 error rate itself. Notably, the human performance is included in the figure for the purpose of comparison. The figure highlights the remarkable progress made in Deep Learning models, as they have achieved performance levels that exceed human capabilities. Figure from Alzubaidi et al. (2021).*

2.2.1 Feature Visualization

Feature Visualization (or Activation Maximization) is a technique that involves making learned features visible (Olah et al., 2017). In the context of neural networks, it refers to finding the input that results in the highest activation of a given unit, in order to visualize the features learned by that unit. The term unit in the context of neural networks can encompass different elements such as individual neurons, feature maps (also called channels), entire layers, or the final class probability in classification tasks (Schmidhuber, 2015).

Although examining the feature visualization of each neuron would provide the most information, this is not practical as neural networks can have millions of neurons. Instead, feature visualization can be done at the channel level, which is a good compromise between granularity and efficiency. Channels refer to the individual feature maps or activation maps within a convolutional layer of a neural network. Each channel represents a specific learned feature or pattern that the network has detected. By visualizing the channels, we can gain insights into the high-level representations learned by the network. Although Feature Visualization is not the main focus of this master thesis, it is mentioned for the sake of completeness. Further exploration and analysis of this topic will not be included in this thesis.

2.2.2 Attribution

Pixel attribution techniques identify the pixels that play a significant role in determining the classification of an image by a neural network (Olah et al., 2017). Due to its youth as a field, there is no standardized terminology for neural network interpretability. Feature visualization is one of the names used in the literature to refer to attribution. However, current research favors the term pixel attribution to describe this concept. Additionally, there are multiple terms used to refer to pixel attribution, including sensitivity map, saliency map, pixel attribution map, gradient-based attribution methods, feature relevance, feature attribution, and feature contribution. To simplify the range of pixel attribution techniques, it is helpful to note that there are two distinct categories

of attribution methods:

Perturbation or Occlusion-based methods generate attribution visualizations by perturbing the input and observing the changes in a model's output, making them a type of Black Box visualization approach (Ivanovs et al., 2021).

Gradient- or Backpropagation-based methods are considered White Box visualization techniques that build on backpropagation by computing the gradient of the prediction or classification score with respect to the input features. There are several pixel attribution techniques that vary depending on the approach used to calculate the gradient (Gur et al., 2021).

Perturbation-based Methods

Perturbation-based methods aim to explain how the network's predictions change when one or more pixels are perturbed. This makes perturbation-based methods an ideal choice for sensitivity analysis of models (Ancona et al., 2019). The sensitivity analysis of deep neural networks is particularly important when dealing with carefully engineered perturbations, known as Adversarial Perturbations (Szegedy et al., 2014), which are imperceptible to the human eye but can cause catastrophic prediction failures. Unlike gradient-based methods that require access to the model parameters for computing the gradients, perturbation-based methods only require forward passes. However, they are often computationally more expensive since they estimate the importance of a subset of input features, which necessitates multiple inference calls (Zintgraf et al., 2017). Similar to backpropagation-based methods, there are numerous existing techniques, and the following list is not exhaustive. The significant methods mentioned by Ivanovs et al. (2021) served as inspiration.

The occlusion sensitivity maps introduced by Zeiler and Fergus (2014) were among the earliest forms of perturbation-based model explanations. They aimed to investigate a deep model's reasoning by systematically replacing different patches of the input image with a solid grey-colored square and observing the corresponding predicted class label. This allowed them to determine whether the model's prediction was consistent with the object's position in the image, or whether the model relied on the surrounding pixels for context. As more of the object area was occluded, the visualizations of the last convolutional layer's activation maps revealed a progressive degradation in the model's predicted probability for the correct class label.

Perturbation-based explanations for prediction models can be applied to a range of models beyond just deep models. In fact, Ribeiro et al. (2016) proposed a method called Local Interpretable Model-agnostic Explanations (LIME) that can generate an interpretable model for any black-box prediction model. LIME generates image visualizations by perturbing the original image, analyzing the influence of different regions through an interpretable model, and creating a heatmap that visually represents the important areas contributing to the model's prediction.

Randomized Input Sampling for Explanation (RISE) is a black-box explanation technique proposed by Petsiuk et al. (2018) that produces pixelwise saliency maps. The method involves presenting multiple versions of the input image to the prediction model, with each version randomly masked. The resulting scores predicted by the model for a specific class are then used as weights to generate a linear combination of the masked images, which yields the saliency map for that class. To obtain these masked images, the input image is multiplied elementwise with binary masks generated through sampling.

In summary, perturbation-based XAI techniques have the advantage of not relying on a model's parameters. However, a drawback of these methods is their potential for increased computational time due to the need for multiple forward passes through the model. This can pose challenges, particularly when dealing with large and complex models or datasets (Zintgraf et al., 2017).

Gradient-based methods

Gradient-based (or Backpropagation-based) techniques assume accessibility to the parameters of deep neural networks and leverage the network's information flow pathway. Using a forward pass to predict the class label and a backward pass to the input layer, these methods estimate input attributions and create activation maps that reflect the contribution of each pixel to the network's final prediction (Ancona et al., 2019). An advantage of gradient-based methods is that they can produce importance scores for all pixels using only one or a few forward and backward passes.

The saliency map approach, proposed by Simonyan et al. (2014), is a technique used to generate saliency maps that highlight the important regions in an input image for a specific target class. The method utilizes gradient-based methods and can be applied to any layer of a neural network, including convolutional layers. The main idea behind the saliency map approach is to compute the gradient of the loss function with respect to the input pixels. The loss function is typically designed to measure the discrepancy between the predicted output of the network and the desired output (e.g., cross-entropy loss). By calculating the gradient of the loss function, the method aims to understand how changes in the input pixels affect the output prediction. To generate a saliency map (see Figure 2.2), the following steps are typically involved:

Forward pass: The input image is fed forward through the neural network, resulting in an output prediction for the target class of interest.

Backpropagation: The gradient of the loss function with respect to the output prediction is calculated using backpropagation. This gradient represents the sensitivity of the output to changes in the prediction scores.

Gradient computation: The calculated gradient is then backpropagated through the network, propagating it backward to the input layer. During this process, the gradient is applied to each layer and each neuron in a specific way depending on the gradient-based method used.

Pixel-wise gradient visualization: Once the gradients are computed, they are visualized to generate the saliency map. The saliency map represents the importance or relevance of each pixel in the input image for the prediction of the target class. The values in the saliency map correspond to the magnitudes and signs of the gradients, indicating the impact of each pixel on the final prediction.

The resulting saliency map provides insights into which regions of the input image are most relevant for the target class prediction. It highlights the regions that have the highest influence on the output, aiding in understanding the model's decision-making process.

In their paper, Springenberg et al. (2015) expanded upon the concept of saliency maps and introduced the guided backpropagation algorithm. This algorithm improves upon standard backpropagation by incorporating additional guidance signals from higher layers of the neural network, resulting in more informative and accurate saliency maps (see Figure 2.3).

At a similar time frame, when interactive visualization became increasingly popular, Yosinski et al. (2015) devised a software application that allows users to interactively view every neuron's reaction in a trained CNN as they input an image or video.

Zhou et al. (2016) proposed an alternative method called class activation map (CAM) as a solution. Their research showed that the Global Average Pooling (GAP) layer had a dual function of not only regularizing the CNN structure, but also preserving its localization ability up to the final layer.

Figure 2.2: SALIENCY MAPS SPECIFIC TO EACH IMAGE WERE GENERATED FOR THE TOP-1 PREDICTED CLASS IN THE ILSVRC-2013 TEST IMAGES. *These maps were obtained by performing a single back-propagation pass through a classification ConvNet. Figure from Simonyan et al. (2014).*

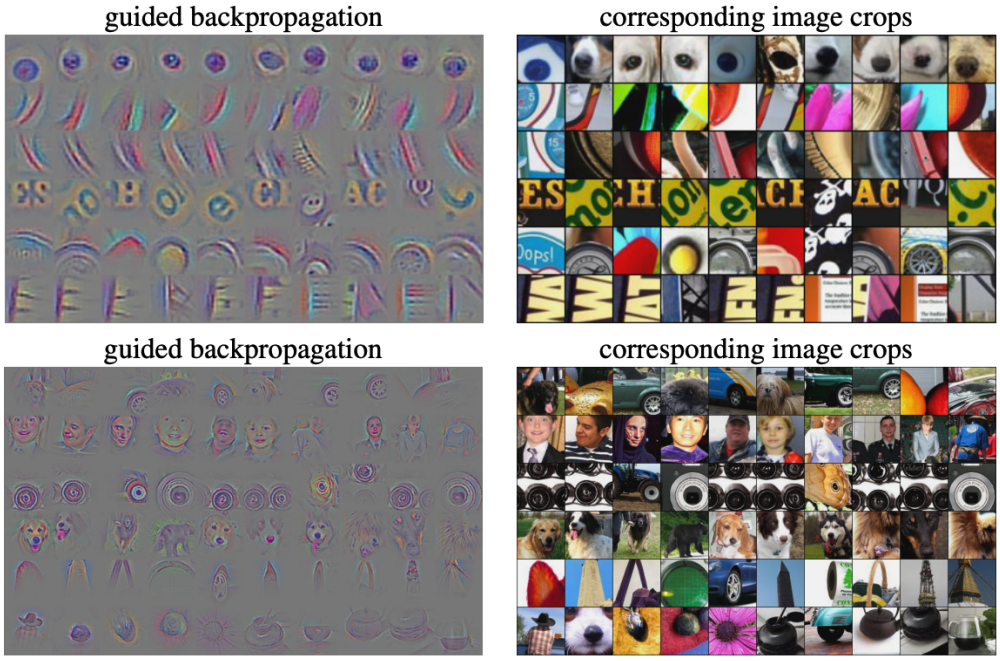


Figure 2.3: GUIDED BACKPROPAGATION VISUALIZATION. *The visualization showcases the patterns learned by two layers of the network trained on ImageNet: conv6 (top) and conv9 (bottom). Each row in the visualization corresponds to a specific filter. The visualization is generated using the guided backpropagation technique and showcases the top 10 image patches from the ImageNet dataset that activate each filter the most. Figure from Springenberg et al. (2015).*

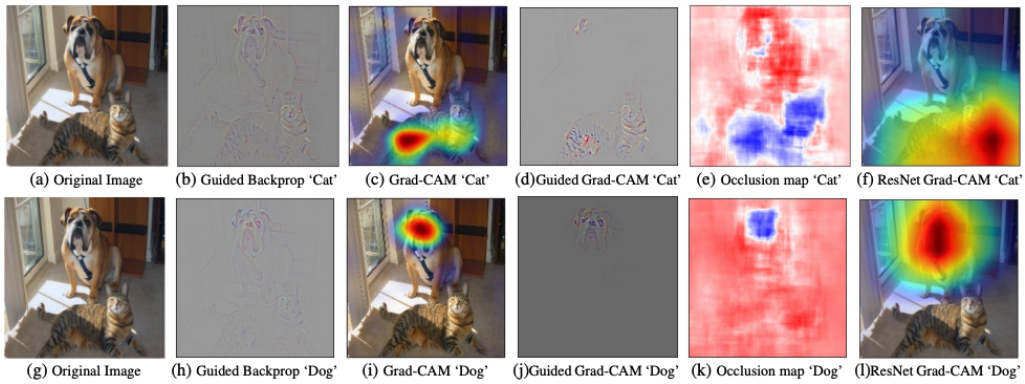


Figure 2.4: GRAD-CAM VISUALIZATIONS. *The original image contains both a cat and a dog, while the subsequent images (b-f) demonstrate the support for the cat category using various visualizations for VGG-16 and ResNet. Guided Backpropagation (b) highlights all contributing features, while Grad-CAM (c, f) localizes class-discriminative regions. Combining (b) and (c) produces Guided Grad-CAM, which yields high-resolution class-discriminative visualizations. Additionally, (f, l) depict Grad-CAM visualizations for the ResNet-18 layer. In (c, f, i, l), the red regions indicate high scores for the class, while blue in (e, k) corresponds to evidence for the class. Figure from Selvaraju et al. (2017).*

This discovery led to the possibility of identifying discriminative regions in a single forward pass and generating class-specific feature maps. We will refer to these visualizations as activation maps. One limitation of CAMs is that the GAP layer must directly follow the convolutional layer being visualized. Without a GAP layer or fully connected layers, CAMs are unsuccessful because the class-wise weights for each activation unit are undefined. To address this issue, Selvaraju et al. (2017) proposed Grad-CAM, which combines feature maps using the gradient signal as a solution. Localization of the target class was accomplished in a single pass with Grad-CAM, requiring only one forward and a partial backward pass per image. The importance score of a neuron is calculated by considering the gradients of the class's logits with respect to the feature activation maps of the final convolutional layer. Logits refer to the raw output values generated by the neural network before applying the final activation function (such as softmax) to obtain class probabilities. To obtain features that positively influence the target class, a ReLU nonlinearity is applied to the weighted combination of the forward activation map. Since Grad-CAM generates only rough visualizations, the authors of the method combined it with guided-backpropagation (Springenberg et al., 2015) to propose a new approach known as Guided Grad-CAM (Selvaraju et al., 2017). They obtained a fine-grained and class-discriminative visualization by taking the element-wise product of guided-backpropagation visualization and Grad-CAM's visualization. Despite its effectiveness, the Guided Grad-CAM method is subject to the limitations of guided backpropagation, which is caused by the elimination of negative gradients during backpropagation (see Figure 2.4).

Over the years, many new visualization techniques were developed to improve the interpretability of CNNs. Smilkov et al. (2017) introduced SmoothGrad, which smoothed out gradients using a Gaussian kernel to reduce noise and improve coherency. Chattopadhyay et al. (2018) proposed Grad-CAM++, a generalized approach that calculates higher-order derivatives for exponential and softmax activation functions. Srinivas and Fleuret (2019) developed Full-Grad, which assigns dual importance scores to input features and individual neurons. Wang et al. (2020) introduced Score-CAM, which encodes activation map significance by global contribution of associated input features. Other methods like Ablation-CAM (Desai and Ramaswamy, 2020),

XGrad-CAM (Fu et al., 2020), Eigen-CAM (Muhammad and Yeasin, 2020), HiRes-CAM (Draelos and Carin, 2020) and Layer-CAM (Jiang et al., 2021) were also developed to determine feature map importance, maintain linearity of feature maps, visualize principal components, highlight specific locations, and generate fine-grained object localization information from activation maps. Although it is not feasible to discuss every newly emerging method, these visualization techniques serve as a valuable foundation for enhancing the interpretability and comprehension of CNNs.

Chapter 3

Background

This chapter includes the dataset selection, existing evaluation metrics and, most importantly, the LOTS paper (Rozsa et al., 2017) that forms the basis of our thesis.

3.1 Dataset

The full ImageNet dataset (Deng et al., 2009) is an extensive collection of over 14 million labeled images spanning more than 21,000 object categories, encompassing a diverse range of object classes such as various animal species, different types of food, and numerous vehicle types. Additionally, the full dataset includes more annotations, such as object detection and segmentation masks, making it a valuable resource for many computer vision tasks, including image captioning, image retrieval, and object detection. However, the dataset's vast size and complexity make training models on it challenging and time-consuming.

The ImageNet 1k dataset, also known as ILSVRC 2012, is a widely recognized benchmark dataset in computer vision, particularly for image classification tasks. The dataset contains over 1.2 million labeled images belonging to 1,000 distinct object categories, gathered from sources like Flickr and Bing. The dataset is divided into training, testing and validation sets, with 1.28 million images used for training and 50,000 for validation. By using this subset, researchers can train and test their models more efficiently while still leveraging the full dataset's diversity and scale.

The experiments conducted in the LOTS paper (Rozsa et al., 2017) initially utilized the VGG Face dataset. The choice of this dataset was motivated by the authors' objective of generating adversarial examples specifically for attacking face recognition networks. Given the nature of the task, which involved targeting face recognition systems, the authors deemed the VGG Face dataset to be suitable and relevant for their research. In the context of this thesis, comparing the newly developed LOTS visualization method with the ImageNet dataset is more relevant. Nonetheless, there is potential for future research utilizing other datasets.

3.2 Layerwise object-target synthesis (LOTS)

This section explains the LOTS technique proposed by Rozsa et al. (2017). The fundamental concept of the LOTS technique builds the basis of this work.

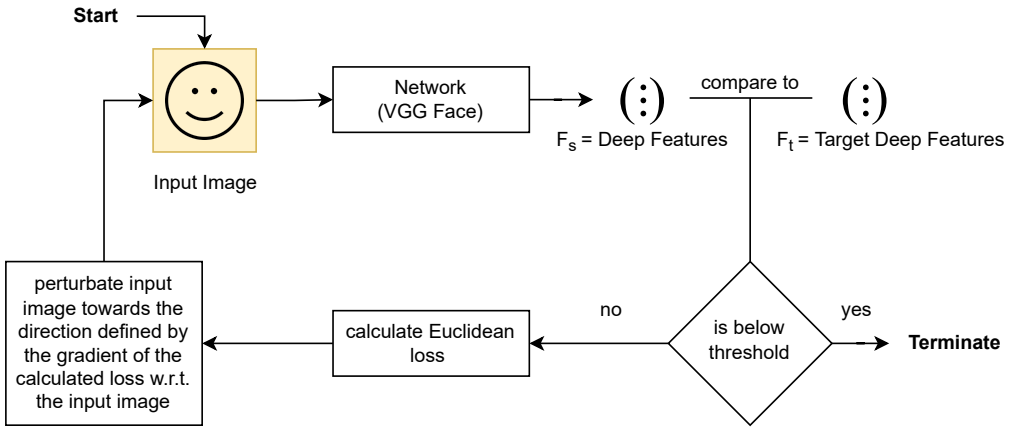


Figure 3.1: LOTS TECHNIQUE. *Visualization of the iterative LOTS technique. Adapted from Chavannes (2022).*

3.2.1 Context

DNNs are the latest state-of-the-art machine learning models, but they are unexpectedly vulnerable to adversarial images. These malicious inputs are formed by adding small perturbations to correctly recognized data, causing the models to misclassify the input. DNNs are believed to be resilient to minor changes in their input data. However, the presence of adversarial images has cast doubt on the efficacy of such vulnerable models and raises questions about their applications. Szegedy et al. (2014) and Goodfellow et al. (2015) were the first to write about adversarial images. Su et al. (2019) demonstrated that a Deep Neural Network can be tricked by just perturbing a single carefully selected pixel in the input image. Later, it was proposed to improve the training process of models by including adversarial examples in the dataset, thus encouraging the development of more robust models. Rozsa et al. (2017) have presented a generic algorithm which can be used to generate adversarial examples for both end-to-end classification networks and for systems which use deep features extracted from DNNs.

3.2.2 Approach

The publicly available VGG Face dataset, which contains 2.6 million images of 2'622 identities, was used as data source. This dataset was primarily selected for its suitability in attacking face recognition systems, which was the main focus of the research. The decision to use this dataset was driven by the need for a face-specific dataset to evaluate and target face recognition networks accurately.

The LOTS technique, depicted in Figure 3.1, involves several steps. Initially, an input image of a person is fed into the VGG Face network to extract deep features (F_s). A target (F_t) is selected to mimic, using the extracted deep features. F_s is then compared to F_t using either cosine distance or Euclidean distance, depending on the experiment. If the distance value is below a defined threshold (τ), the process concludes, and an adversarial image is generated, imitating the target's deep features. However, if the distance value exceeds τ, the Euclidean loss ($\mathcal{L}_2$) between F_s and F_t is calculated. The input image is slightly modified in the direction specified by the gradient of the loss with respect to the image ($\alpha \cdot \nabla_I \mathcal{L}$), where α is the step width taken. Typically, gradients are

employed alongside an optimizer to adjust network weights, aiming for a closer match between the expected and actual results of a forward pass with a specific input. In this case, though, the authors utilize the gradient to modify the pixel values of the input image, aiming to bring F_s closer to F_t, with fixed network weights. The entire process is repeated until the comparative distance between F_s and F_t falls below the threshold. To clarify the process further, we provide a line-by-line explanation of Algorithm 1.

Algorithm 1 Original LOTS Algorithm

```
1:  function LOTS(image_init, F_t, iter, τ, α)
2:      image_adv = image_init
3:      for range(iter) do
4:          F_s = model.get_features(image_adv)
5:          if
6:              distance (F_s, F_t) > τ then
7:              loss = L_2(F_s, F_t)
8:              gradient = loss.backpropagation().get_gradient(image_adv)
9:              gradient_step = gradient × (α/max(abs(gradient)))
10:             image_adv = clamp(image_adv - gradient_step)
11:         else
12:             return image_adv
13:         end if
14:     end for
15:     return image_adv
16: end function
```

- Line (1): The LOTS function is defined as follows: The input image ($image_{init}$) to be synthesized into an adversarial image. The target features (F_t) that the adversarial image should mimic. The number of iterations ($iter$). The threshold (τ) sets an upper limit for the distance between the features extracted from the adversarial image and F_t. To achieve a successful attack, the distance between these features must be less than or equal to τ. Both the cosine distance and the Euclidean distance can serve as viable distance metrics. The step width (α) represents the magnitude of each incremental step taken during an iterative optimization process.

- Line (2): The variable $image_{adv}$ is initially assigned as a copy of the original $image_{init}$. Throughout the algorithm, $image_{adv}$ undergoes synthesis to become the final adversarial image, while still representing the original image before any modifications were made.

- Line (3): Iterates over a range of iterations for a specified amount of times.

- Line (4): Feature extraction for $image_{adv}$. On the first iteration, this is equal to extracting features from $image_{init}$.

- Line (5-6): If clause to check whether the distance between F_s and F_t is above the predefined threshold τ.

- Line (7): The Euclidean loss ($\mathcal{L}_2$) is computed by comparing F_s with F_t. The formal definition of the Euclidean loss is presented in Equation 3.1.

- Line (8): During the backpropagation step, the gradients are computed. The gradient of the loss with respect to $image_{adv}$ is stored in the gradient variable. The formal definition of the gradient is provided in Equation 3.2.
- Line (9): The result, denoted as $gradient_{step}$, is the scaled gradient that will be used to update the parameters or variables being optimized. By normalizing the gradient in this way, it helps to control the magnitude of the updates and stabilize the optimization process.
- Line (10): The variable $image_{adv}$ is updated by subtracting the $gradient_{step}$ from it. The result is then clamped, which means that any values below the minimum allowed value or above the maximum allowed value are adjusted to the nearest boundary. This ensures that the updated $image_{adv}$ remains within the desired range or constraints.
- Line (11-14): If the distance between F_s and F_t is below the threshold τ, the algorithm has converged and the adversarial image $image_{adv}$ is returned.
- Line (15): Current adversarial image $image_{adv}$, where the distance between F_s and F_t did not fall below a certain threshold τ.
- Line (16): End of the LOTS function.

$$\mathcal{L}_2(F_s, F_t) = \frac{1}{2}||F_t - F_s||^2 \tag{3.1}$$

$$gradient(F_s, F_t) = \nabla_{image_{adv}}(\mathcal{L}_2(F_s, F_t)) \tag{3.2}$$

3.2.3 Visualizing Perturbations

An adversarial example is created by adding perturbations p to the original input image i, which results in the Adversarial image a. This relationship can be expressed as the Equation 3.3:

$$a = i + p \tag{3.3}$$

Consequently, we can also express p as the difference between the Adversarial image a and the original image i, that is, $p = a - i$.

The important pixels of the input image that had to be adjusted to achieve an adversarial image can be observed in the perturbation located at the center, as depicted in Figure 3.2. Similar to other techniques, shown in subsection 2.2.2, adversarial methods, including LOTS, could serve as a valuable means of illustrating network decisions for image classification. Figure 3.3 demonstrates the feasibility of LOTS perturbation visualization, as shown by Rozsa et al. (2017). There are other adversarial image generation methods that can be used for visualization. Goodfellow et al. (2015) introduced the fast gradient sign (FGS) method. Rozsa et al. (2016) extended the FGS method by considering a scaled version of the raw loss gradient instead of using only the sign of the gradient, and developed the fast gradient value (FGV) method. Rozsa et al. (2016) also introduced the hot/cold approach, which took the Image Inverting method as motivation (Mahendran and Vedaldi, 2015).

By analyzing these perturbations, researchers gain insights into which regions of the input image the network is relying on to make its decision. Visualizing these perturbations highlight the most important pixels that the network uses to make its prediction. Moreover, by identifying which specific parts of the image are used for the network's decision, it may be possible to generate more robust and interpretable models. Therefore, perturbation visualization can serve as a valuable tool for understanding deep networks that is distinct from the saliency maps discussed in subsection 2.2.2 in ways that are not limited to class-specific visualizations.

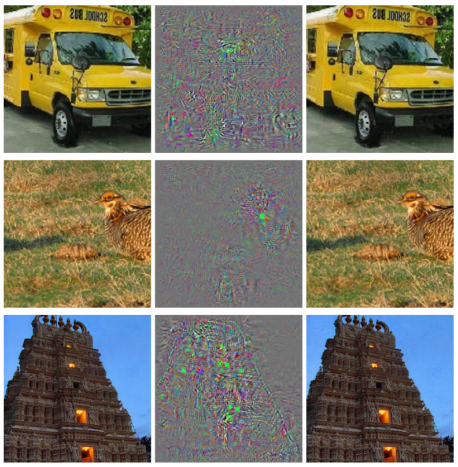


Figure 3.2: ADVERSARIAL EXAMPLES. *Adversarial examples generated for AlexNet. (Left) is correctly predicted sample, (center) difference between correct image, and image predicted incorrectly magnified by 10x (values shifted by 128 and clamped), (right) adversarial example. All images in the right column are predicted to be an ostrich, Struthio camelus. Figure from Szegedy et al. (2014).*

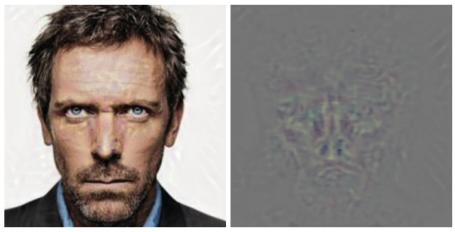


Figure 3.3: LOTS PERTURBATION VISUALIZATION. *The left part of this illustration displays an adversarial example, while the right part visualizes the perturbations that led to the creation of the adversarial image. Figure from Rozsa et al. (2017).*

3.3 Evaluation Metrics

Since there are numerous visualization techniques available, determining the best one can be challenging. Evaluating those techniques is not straightforward due to the absence of a universally agreed-upon evaluation metric. Different visualization methods may emphasize different aspects of the underlying model's behavior, making it difficult to define a single metric that captures all relevant criteria. Additionally, the choice of evaluation metric often depends on the specific application or research question at hand, further contributing to the complexity of selecting the best technique. Therefore, current research suggests a combination of metrics (Gildenblat and contributors, 2021). Hedström et al. (2023) have created a toolbox called Quantus, which serves as a comprehensive tool for quantitative evaluation of visualization techniques. They propose that XAI metrics should be classified into one of six categories. These categories include faithfulness, robustness, localization, complexity, randomization, and axiomatic metrics.

Faithfulness is a measure of the extent to which visualization techniques align with the predictive behavior of the model.

Robustness evaluates the stability of visualization techniques under slight input perturba-

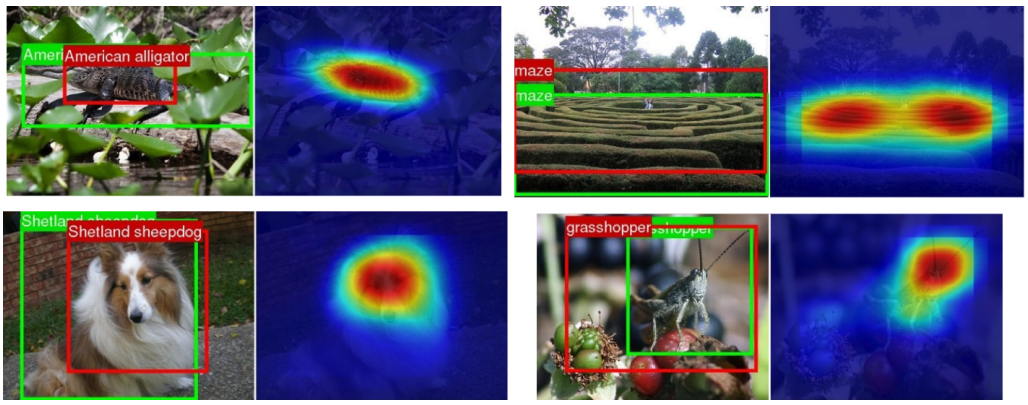


Figure 3.4: EVALUATING CLASS ACTIVATION MAPS BY USING THEM FOR LOCALIZATION. *The predicted bounding boxes from the class activation map are in red, while the ground-truth boxes are in green. Figure from Zhou et al. (2016).*

tions, assuming that the model output remains approximately unchanged.

Localization evaluates whether the explainable evidence is concentrated on a specific region of interest (ROI), which could be defined using a bounding box, a segmentation mask, or a cell within a grid, typically centered around an object.

Complexity measures the extent to which visualization techniques are concise, denoting that they use a minimal number of features to explain a model's prediction.

Randomization evaluates the extent to which visualization techniques worsen in quality as the input image becomes increasingly randomized.

Axiomatic assesses whether visualization techniques satisfy fundamental principles specified by Hedström et al. (2023).

However, Hedström et al. (2023) points out that the evaluation metrics for XAI methods are often based on empirical interpretations or translations of qualities that some researchers have deemed essential for visualization methods. Consequently, there might be a gap between what the author intends to measure with the proposed metric and what is actually measured. Regrettably, although Quantus has a repository on GitHub, it has not been thoroughly tested yet. Nevertheless, it is worth considering for future evaluations.

The subsequent sections will provide a more detailed analysis of some chosen metrics.

Using Visualization methods for Localization

The proposed method in the CAM paper by Zhou et al. (2016) involves assessing visualization methods by generating bounding boxes from them and comparing them to the bounding boxes in the ILSVRC dataset, which contains annotations for objects in ImageNet (see Figure 3.4). The underlying idea is that a good and accurate explanation should have an overlap with the object it represents. To create a bounding box, the CAM algorithm selects the top 20% highest pixels and identifies the largest connected component.

It is important to note that activation and localization are not equivalent (Dabkowski and Gal, 2017). As an example, when it comes to humans identifying a dog, usually only the sight of its head is sufficient, and the information conveyed by its legs and body may not be necessary.

Consequently, a good activation map for a dog will highlight only its head, whereas a localization box will encompass the entire dog, including non-salient details like legs and tail.

The quality of visualizations can be assessed using different evaluation metrics. One such metric is the Pointing Game, introduced by Zhang et al. (2018). It is a human evaluation metric that scores a hit when the highest activation mapping point falls within the bounding box of an object annotated by a human. A miss is counted if the highest activation mapping point falls outside the human-annotated bounding box. The accuracy is then calculated as $Accuracy = \frac{\#Hits}{\#Hits+\#Misses}$, where $\#$ denotes number of.

On the other hand, the Score-CAM paper by Wang et al. (2020) suggests an alternative approach for evaluating localization, which involves computing the total sum of CAM pixels within the corresponding bounding box.

Image Perturbation for Evaluating Visualization Techniques through Prediction

The subsequent Grad-CAM++ paper by Chattopadhyay et al. (2018) introduced popular metrics that are still in use today. The method involves multiplying the image by the generated activation map, resulting in the visibility of only the high-scoring regions (see Figure 3.5), which is commonly known as the explanation map. The next step involves running the explanation map through the model and examining the new prediction scores. The proposed metrics are as follows:

$$\textbf{Average Drop} : \sum_{i=1}^{N} \frac{\max(0, Y_i^c - O_i^c)}{Y_i^c} \times 100, \quad \textbf{Average Increase} : \sum_{i=1}^{N} \frac{Sign(Y_i^c < O_i^c)}{N} \times 100, \tag{3.4}$$

where Y_i^c is the predicted score for class c on image i and O_i^c is the predicted score for class c with the explanation map as input. $Sign$ presents an indicator function that returns 1 if input is True. The average drop calculates the percentage reduction in confidence (or 0 if the confidence increased). Since we have solely removed insignificant portions of the image, the optimal value for this metric would be as low as possible. The average increase (or increase in confidence) indicates the number of instances where the confidence increased. Our aim is to enhance the model's confidence in its predictions by eliminating irrelevant pixels. As a result, we desire the metric to achieve the highest possible value.

Insertion and Deletion

One aspect of visualization methods that should be quantifiable is the activation map's fidelity: A measure of fidelity should capture how well the visualization method assigns relevance values to the input pixels. Therefore, Petsiuk et al. (2018) presented a metric called Insertion/Deletion, which is twofold (see Figure 3.6).

The Deletion metric assesses the decrease in class probability as significant pixels, identified by the activation map, are systematically removed from the image. The rate of removal is determined by the number of steps taken, which indicates the division of the activation map into chunks, gradually replacing pixels in the original image. A significant drop and a small area under the probability curve (AUC) indicate a good visualization technique. On the other hand, the Insertion metric assesses pixel importance by measuring the increase in the probability of the target class when pixels are added based on the generated activation map. There are multiple techniques to remove pixel values from an image, including setting pixel intensities to zero, blurring the region, or introducing noise (Fong and Vedaldi, 2017).

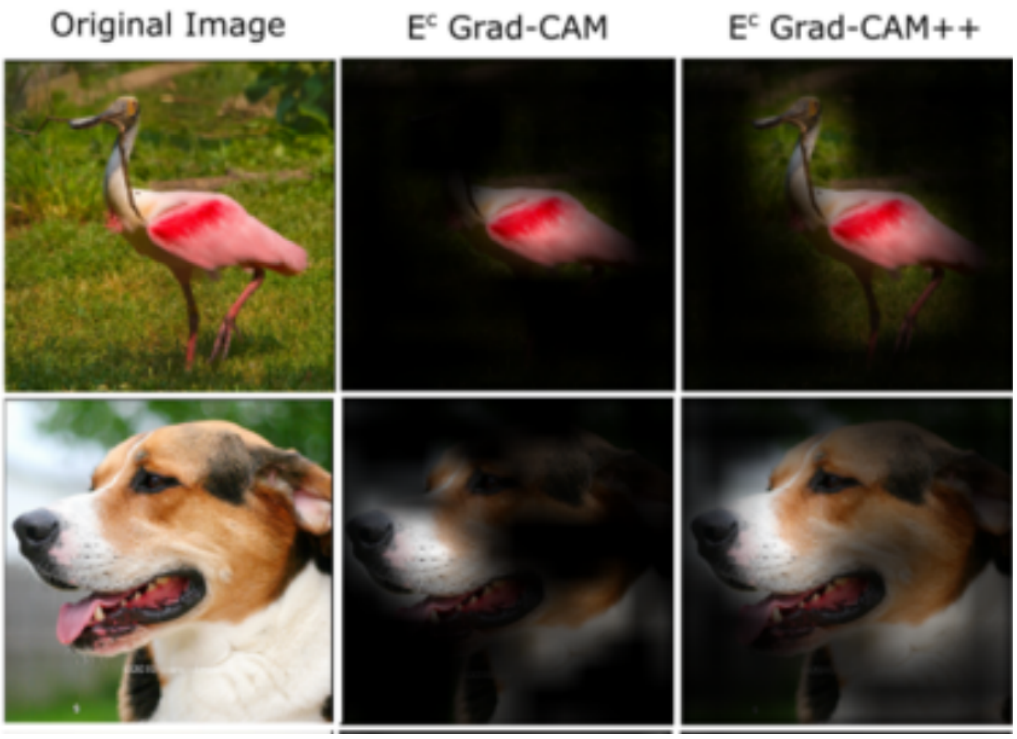


Figure 3.5: GRAD-CAM++ EVALUATION. *By multiplying the original image with the computed activation map, an explanation map is generated, which is a masked image that excludes unimportant areas of the initial image. Figure from Chattopadhyay et al. (2018).*

Maximum Coherency

It is important for visualization techniques to include all the relevant pixels from the input image that contribute to a prediction, while masking irrelevant pixels in a coherent manner. This means that the activation map of one image should be equal to that of the explanation map calculated with the same visualization techniques (Poppi et al., 2021). Consequently, when given an input image x and a specific class of interest c, the activation map CAM should remain unchanged when conditioning x on the activation map CAM itself. Formally,

$$CAM_c(x \odot CAM_c(x)) = CAM_c(x). \tag{3.5}$$

Drawing from previous research on the comparison of activation maps, Poppi et al. (2021) utilize the Pearson Correlation Coefficient to measure the similarity between the two activation maps mentioned in Equation 3.5:

$$Coherency(x) = \frac{Cov(CAM_c(x \odot CAM_c(x)), CAM_c(x))}{\sigma_{CAM_c(x \odot CAM_c(x))}\sigma_{CAM_c(x)}}, \tag{3.6}$$

In the given equation, the symbol Cov represents the covariance between two maps, and σ denotes the standard deviation. To ensure that the coherency score falls within the range of 0 to 1, the authors normalize it, since the Pearson Correlation Coefficient ranges from -1 to 1. Consistent

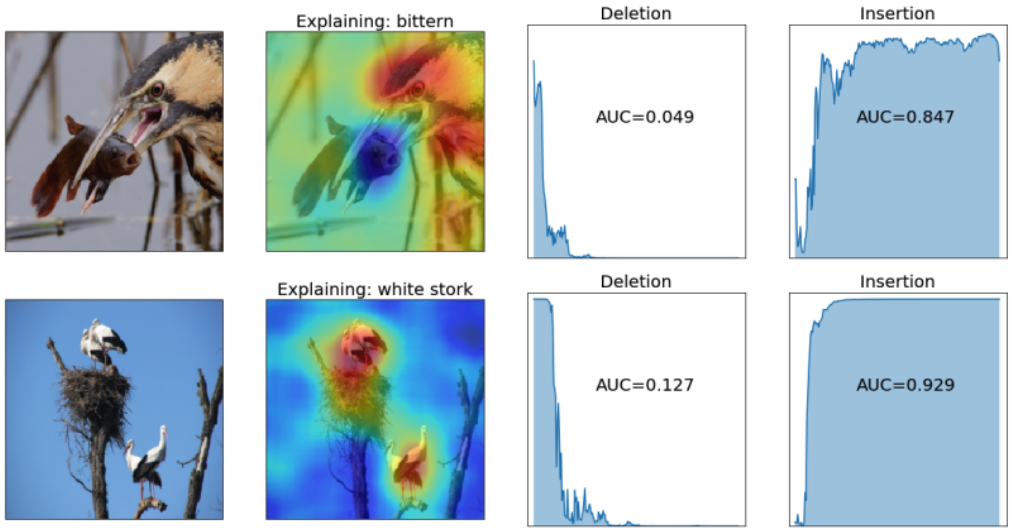


Figure 3.6: DELETION/INSERTION METRIC. *The second column displays the activation heatmaps generated by RISE (Petsiuk et al., 2018) for two representative images shown in the first column. The third column presents the Deletion curves, showcasing the drop in class probability as pixels are progressively removed based on the activation map. Conversely, the fourth column illustrates the Insertion curves, representing the rise in class probability as pixels are added according to the activation map. Figure from (Petsiuk et al., 2018).*

with existing metrics, the coherency score is defined as a percentage. It is worth noting that the coherency is maximized when the attribution method remains unaffected by changes in the input image.

Conclusion

It has become common to use multiple evaluation metrics instead of relying on one. It is also recommended to use a Random-CAM as a benchmark for comparison. Gildenblat and contributors (2021) define their Random-CAM implementation as a visualization method that returns random weights for the activation maps based on the shape of the gradients provided as input. The random weights are uniformly sampled from the range -1 to 1. Effective visualization methods should perform better than Random-CAM on average. Nevertheless, it is important to interpret the results with care and verify them against benchmarks such as Random-CAM before making any substantial conclusions.

Chapter 4

Approach

The objective of this thesis is to enhance the LOTS algorithm to generate interpretable visualizations for image recognition tasks. This chapter presents a detailed description of the process involved in extending the LOTS algorithm to achieve this. The approach involves modifying the existing LOTS algorithm to incorporate interpretability by developing techniques for creating visual explanations of the decision-making process in image recognition tasks. In addition, this work seeks to answer the research questions of how visualizations can be evaluated and whether the extended LOTS visualization technique can be applied to classes not present in the dataset. Through this contribution, we aim to advance the research focused on enhancing the transparency and interpretability of deep learning algorithms in computer vision.

4.1 Dataset

As detailed in chapter 3, the ImageNet 1k dataset was selected as the underlying dataset for this research. With over a million images belonging to a thousand distinct classes, the dataset provides a comprehensive and diverse source of data for our research. Furthermore, the standardized evaluation protocol offered by the ImageNet dataset played a significant role in its selection. The publicly available dataset is divided into two subsets, namely the training and validation sets. The training set is utilized to train the model, while the validation set is used to optimize hyperparameters and select the best model. Such a protocol ensures uniform evaluation of models and facilitates fair comparison of different models. Moreover, pre-trained models specifically for ImageNet have become widely accessible and openly available. The availability of these pre-trained models has accelerated the development of computer vision applications and enabled researchers to explore new frontiers in the field. Finally, the ImageNet dataset was chosen as it is a widely recognized benchmark for evaluating computer vision models. Its diverse range of images, with complex scenes and lighting conditions, pose a challenging task for computer vision models to accurately classify and detect objects. The dataset's challenging nature further highlights its significance in advancing the state-of-the-art in computer vision. Overall, the ImageNet dataset is an essential resource for researchers in the field of computer vision, given its size, diversity, standardized evaluation protocol, and challenging nature, all of which render it an ideal dataset for training and evaluating deep neural networks.

4.2 Target selection for LOTS

In a deep network, as the depth of the layer increases, the features become increasingly abstract, enabling the network to capture more sophisticated concepts such as object categories, scenes, or textures. For example, layers in the middle of the network may detect object parts, while the deepest layers may identify high-level concepts such as object classes. Therefore, the deeper the layer in the network, the more abstract the features that it captures, and the more suitable they are for higher-level tasks like object recognition and scene understanding. Given that we have selected an object detection task, we made the decision to focus on the last layer of each CNN.

As depicted in Figure 3.1, for the LOTS algorithm, it is necessary to select a target deep feature F_t to enable the convergence of the initial image deep features F_s. The selection of F_t requires careful consideration, and various attempts were made to determine an optimal choice. We came up with three alternatives to elaborate on.

The first alternative is to have a F_t which consists of the zero vector of size F_s. The motivation behind that, is the meaning of the zero vector in the deep feature space. In the deep feature space of a neural network's last layer, the zero vector corresponds to an image that is very unlikely to belong to any class in the dataset. This is because the deep features of the zero vector indicate that the image does not contain any discernible patterns or structures that the network has learned to recognize as belonging to a specific class. In other words, the zero vector represents an image that has no meaningful representation in the feature space of the neural network. Therefore, it is unlikely that this image would be classified as belonging to any particular class by the network.

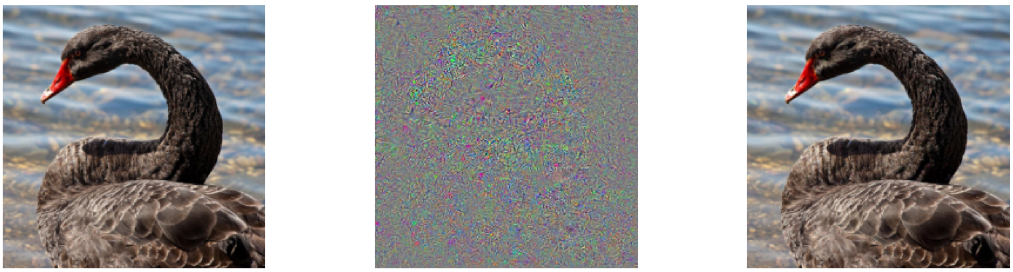
In the context of generating adversarial examples and their corresponding perturbations, the LOTS algorithm aims to bring the deep features of the original image F_s closer to the target deep features F_t. This results in an adversarial image with perturbations that affect the pixels in a way that the model's prediction is shifted away from any specific class. In this regard, the perturbations can be interpreted as the specific parts of the image that were responsible for influencing the model's decision in favor of a particular class or classes. Therefore, the perturbations with zero vector F_t can provide valuable insights into how the model processes and interprets images. A potential limitation of using zero F_t is that the presence of multiple object classes in a single image may result in the activation of all class-specific pixels. This differs from CAM techniques, which only highlight the pixels corresponding to the highest-scoring object class when no specific target is indicated. However, by construction, the ImageNet dataset should not include images that contain several known objects of different classes.

The second and third alternative of F_t are interconnected. Specifically, the second approach involves selecting a random image from each class in the training set and computing its deep feature vector F_t^c, where c corresponds to the specific class. When we input an image of a class to LOTS, we extract the deep features of the image corresponding to a class F_s^c and use the corresponding F_t^c to converge to. The underlying objective of the second approach is to generate perturbations for each image that could help identify which pixel values in the image should be altered to produce a comparable deep feature space to F_t.

As the third and final concept, we explore using the mean feature vector as F_t. The approach is similar to the second idea, but involves computing the average deep feature vector of each class across the entire training dataset of ImageNet. This introduces an additional pre-calculation step with the expectation of achieving more equitable deep features F_t.

4.3 LOTS visualization extension

In chapter 3, we introduced the LOTS algorithm and explained the motivation behind expanding it to create XAI visualizations. The initial LOTS paper had already demonstrated the use



(a) Original (b) Perturbations (c) Adversarial image

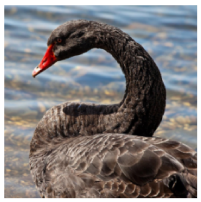
Figure 4.1: PERTURBATIONS VISUALIZED AS DIFFERENCE BETWEEN ORIGINAL AND ADVERSARIAL IMAGE. *Image 4.1(a) shows the original image, 4.1(c) shows the generated adversarial image through LOTS and 4.1(b) shows the difference between original image and adversarial image magnified by 10x (values shifted by 128 and clamped).*

of visualization (see Figure 3.3). Our main aim in this thesis is to extend the LOTS algorithm for visualizations. However, ongoing debate surrounds the comparability among visualization methods. To address this issue, we developed a visualization technique that can be compared to other CAM-based methods. We aimed to generate activation maps from adversarial perturbations, which can be obtained using Equation 3.3. By calculating the difference between the adversarial and initial images across the RGB color channel, we can visualize the perturbations (see Figure 4.1).

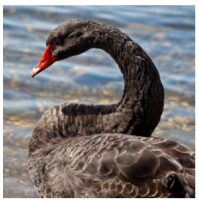
In order to maintain comparability with existing CAM-based methods and leverage selected metrics, we developed a technique to manipulate the provided perturbations, enabling us to generate a visualization that can be compared to CAM-based visualization techniques.

The proposed LOTS visualization extension is outlined in Figure 4.2 and detailed in Algorithm 2. The process starts with an original RGB image 4.2(a) and the adversarial image 4.2(b) created through LOTS. We then convert both the original image 4.2(c) and the adversarial image 4.2(d) to grayscale. The perturbations 4.2(e) are obtained by taking the absolute difference between 4.2(c) and 4.2(d), as only absolute pixel changes are relevant. Additionally, To make the perturbations visible, the obtained values are min-max normalized. To achieve a comparable activation map with CAM methods, a crucial step is to use a Gaussian blurring filter. A Gaussian blur filter applies a weighted average to each pixel of an image, resulting in a smoothing effect by reducing high-frequency details and reducing image noise. Following an additional min-max normalization process, our LOTS activation map, referenced in 4.2(f), is generated. Multiplying the original image 4.2(a) with the created activation map 4.2(f), we arrive at an explanation map 4.2(g). From the activation map 4.2(f), we can easily generate an activation heatmap and lay it over the initial image to see the focus of LOTS 4.2(g). In the context of the Gaussian filter, the choice of kernel or filter size, as indicated in Equation 4.2, introduces an additional parameter. Increasing the size of the kernel leads to a greater degree of blurring, causing a more pronounced reduction in high-frequency details within the image. Conversely, reducing the kernel size results in less blurring, thereby preserving finer image features. As a result, our method aims to visualize broader regions by employing a larger filter, while using a smaller filter to emphasize and capture the details in specific regions.

The implementation of the LOTS visualization extension utilizes PyTorch and slightly deviates from Algorithm 2. To clarify, we go through the algorithm line-by-line.



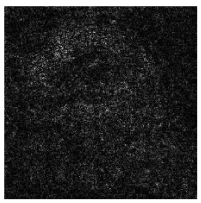
(a) Original image



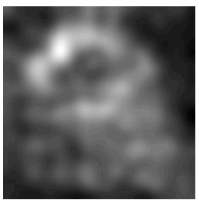
(b) Adversarial image

(c) Original image grayscale

(d) Adversarial image grayscale

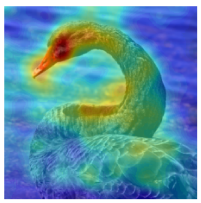


(e) Absolute difference grayscale



(f) Activation map

(g) Explanation map



(h) Activation heatmap over original image

Figure 4.2: LOTS VISUALIZATION PROCEDURE. *4.2(a) shows the original image, 4.2(b) displays the adversarial example generated through LOTS ($iter = 500$, $\tau = 0.1$, $\alpha = \frac{1}{255}$ and $F_t = zero\ vector$), 4.2(c) and 4.2(d) show the respective image in grayscale, 4.2(e) visualizes the absolute difference of 4.2(c) and 4.2(d), 4.2(f) after applying the Gaussian blur filter ($filter\ size = 49 \times 49$) on 4.2(e), the explanation and acitvation heatmap are displayed in 4.2(g) and 4.2(h).*

- Line (1): The visualizeLOTS function is defined. The function takes three parameters: $image_{init}$ representing the initial image or input data for visualization, $image_{adv}$ representing an adversarial image or perturbed version of the initial image, and $filterSize$ representing the size or dimension of the Gaussian blur filter used in the visualization process.
- Line (2): The input image $image_{init}$ is converted from RGB to grayscale.
- Line (3): The adversarial image $image_{adv}$ is converted from RGB to grayscale.
- Line (4): The perturbations are calculated using the absolute difference between the initial image in grayscale $image_{initGray}$ and the adversarial image in grayscale $image_{advGray}$.
- Line (5): $perturbations_{norm}$ represents the normalized $perturbations$ calculated through Equation 4.1. By subtracting the minimum value of $perturbations$ and dividing it by the difference between the maximum and minimum values, the $perturbations$ are scaled to the range of 0 to 1. This makes the minor perturbations visible.
- Line (6): The given line applies a Gaussian blur filter to the $perturbations_{norm}$ using a specific filter size. The filter itself is a gaussian kernel, detailed in Equation 4.2, where x represents the distance from the origin along the horizontal axis, y represents the distance from the origin along the vertical axis, and σ is the standard deviation of the Gaussian distribution. The function GaussianBlur, which applies the image convolution operation using the generated kernel, is called with two parameters: $perturbations_{norm}$ which represents

the normalized data to be blurred, and $(filterSize, filterSize)$ a tuple specifying the dimensions of the square Gaussian blur filter to be applied. The operation blurs the image and reduces high-frequency details or noise. The size of the filter determines the extent of blurring. A larger filter size results in a stronger blur effect.

- Line (7): Again, $norm_{min-max}$ detailed in Equation 4.1 is applied on $perturbations_{blurred}$ to generate the $activationMap$. Normalizing the blurred data after applying Gaussian blurring guarantees a consistent range, facilitates comparison, and enhances the visibility of patterns.
- Line (8): Return the $activationMap$
- Line (9): End of the visualizeLOTS function.

Algorithm 2 LOTS Visualization Extension

1: **function** VISUALIZELOTS($image_{init}$, $image_{adv}$, $filterSize$)
2: $image_{advGray}$ = $image_{adv}$.toGrayscale()
3: $image_{initGray}$ = $image_{init}$.toGrayscale()
4: $perturbations$ = $abs(image_{advGray} - image_{initGray})$
5: $perturbations_{norm}$ = $normalize_{min-max}(perturbations)$
6: $perturbations_{blurred}$ = GaussianBlur($perturbations_{norm}$, ($filterSize$, $filterSize$))
7: $activationMap$ = $norm_{min-max}(perturbations_{blurred})$
8: **return** $activationMap$
9: **end function**

$$perturbations_{norm}(pixel_{values}) = \frac{pixel_{values} - pixel_{values_{min}}}{pixel_{values_{max}} - pixel_{values_{min}}} \tag{4.1}$$

$$\text{GaussianKernel}(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{4.2}$$

Chapter 5

Experiments and Results

In Chapter 4, we presented the approach used to enhance the LOTS algorithm to perform visualizations. In this chapter, we demonstrate the quantitative and qualitative evaluation of the proposed method through various experiments.

5.1 Quantitative Analysis

This quantitative section focuses on addressing research question RQ2. The evaluation of our proposed LOTS visualization extension involved two main experiments. Firstly, we conducted an experiment to determine a suitable target F_t from the three options presented in section 4.2. Subsequently, we compared selected CAM-based visualization techniques (including Random-CAM) with our LOTS method. Both experiments utilized a combination of predefined metrics.

To enhance computational efficiency and streamline our analysis process, we selected a subset of the ImageNet ILSVRC2012 validation set. To ensure diversity within the selected subset, we chose 1000 images, equivalent to one image per class. In order to achieve replicable results, we selected the first image per class from the validation set to be part of our chosen subset. Unless specified otherwise, we consistently utilized this dataset subset for all our experiments.

We employed a set of five metrics from the evaluation part outlined in section 3.3 to assess the visualization techniques. The Average Drop and Average Increase metrics were utilized to evaluate the visualization technique's ability to selectively mask pixels that have an influence, or no influence, on the model prediction, resulting in increased or slightly decreased confidence levels. To verify if the model was indeed not concentrating on newly introduced black pixels when calculating the confidence increase/decrease, indicating distraction caused by the explanation map, we employed the Coherency metric. This assessment enabled us to determine the similarity between the activation map of the initial image and the activation map of the explanation map. When there is a strong alignment between these maps, it leads to a high Coherency score, indicating that the attention did not shift towards the newly masked pixels. Finally, the Insertion and Deletion metrics were employed. To minimize the introduction of artifacts, especially when applying convolution filters, the pixels that needed replacement were substituted by blurring the corresponding pixels in the original image rather than being set to zero (black). This approach was chosen to maintain a connection with the original data, reducing the likelihood of undesired distortions. Unlike the original metric, we adopted the highest class probability as our prediction, considering that LOTS with a zero F_t does not have a specified class for visualization purposes. Further, parameter decisions were relevant to the Deletion/Insertion metric, which required a setting of the number of steps to gradually replace the original pixel with blurred ones. To this end, we employed the suggested configuration proposed by the authors of the metric (Petsiuk

	ResNet-50				
Target	Avg Drop ↓	Avg Increase ↑	Coherency ↑	Insertion ↑	Deletion ↓
Zero F_t	**11.31**	**43.80**	**90.09**	**32.20**	**18.92**
Class F_t	12.66	41.70	75.17	31.11	21.29
Mean F_t	11.66	43.60	80.68	31.23	21.13

Table 5.1: COMPARING LOTS WITH THREE DIFFERENT TARGETS F_t. *The ResNet-50 architecture with the latest ImageNet 1k v2 pretrained weights was used to evaluate the five metrics. The metrics of Average Drop, Average Increase, Coherency, Insertion, and Deletion are compared for three LOTS targets F_t. An upward-pointing arrow indicates that the metric should be maximized, while a downward-pointing arrow suggests the metric should be minimized. All the results presented in the table are expressed in percentage.*

et al., 2018), which proposed using 224 steps for the ImageNet image size of 224 x 224.

Before starting the experiments, several parameters needed to be configured for the LOTS method. The threshold τ was set to 0.1, and a maximum of 500 iterations $iter$ were established, with a step width α of $1/255$. The size of the filter for the Gaussian blur (see Algorithm 2) directly influences the level of detail captured by the activation map (see section 4.3). In light of this, we configured the Gaussian filter to have dimensions of 49 x 49 pixels.

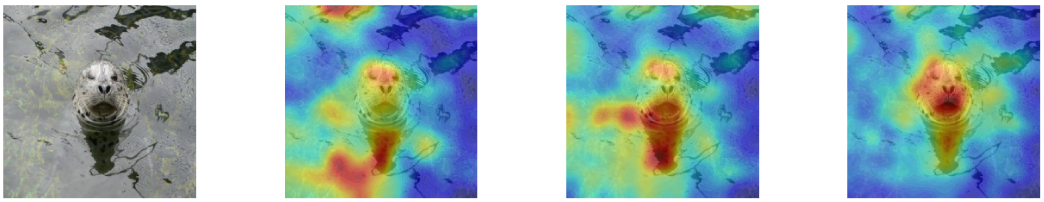
5.1.1 LOTS Target Selection

At the outset, we needed to determine whether to employ the zero target vector, the class target vector or the mean target vector as F_t (see section 4.2). For each image in our dataset subset, we employed either the corresponding class feature or the mean feature as the target for convergence. Hence, each image with the class source feature F_s^c was associated with a class-specific target F_t^c. It is worth noting that the LOTS with the zero target remained the same for every image. The LOTS target experiment was done on the ResNet-50 architecture.

Table 5.1 demonstrates that the zero F_t yields better results compared to the other two targets across all metrics. However, the differences between the zero F_t and mean F_t are relatively small in four out of five metrics. Significantly, the Coherency metric demonstrates that LOTS with a zero F_t produces more robust activation maps, indicating that the addition of the black pixels has minimal disruption. It is noteworthy that the zero F_t is particularly good on the Insertion and Deletion metrics. This could be attributed to the specific area that LOTS is targeting with the zero F_t. By using the zero F_t as the convergence goal for LOTS, as described in section 4.2, we can visualize all features of an image that the model identifies. To achieve high performance on the Insertion and Deletion metric, it is crucial to visualize all the features that contribute to a model's shift in confidence. This is where LOTS (with the zero target F_t) seems to have its strengths.

Furthermore, the variation among the three targets F_t can be attributed to the adversarial nature of LOTS. When specifying a class target F_t in LOTS, the visualization process includes all image features that do not resemble the target F_t. Therefore, the resulting activation map places less emphasis on class-discriminative pixels and more emphasis on pixels that only partially resemble the target F_t. This distinction is evident in Figure 5.1, where subfigure 5.1(b) primarily focuses on the shadow and water surrounding the sea lion, with minimal emphasis on the sea lion's head. Subfigure 5.1(c) removes less significant features related to the sea lion class, such as the water, while subfigure 5.1(d) primarily highlights the sea lion itself.

Furthermore, the zero F_t eliminates the need for an additional computational process that involves analyzing all features across the training images. By using the zero F_t, this additional step is avoided, simplifying the overall computational complexity and streamlining the visualization process. Based on the aforementioned advantages and superior performance, we have decided to



(a) Original image (b) LOTS with class target F_t (c) LOTS with mean target F_t (d) LOTS with zero target F_t

Figure 5.1: LOTS TARGET COMPARISON. *Subfigure 5.1(b), 5.1(c) and 5.1(d) were generated using the same parameters specified in the introduction of this section.*

utilize the zero F_t as the target for LOTS in the subsequent experiments.

5.1.2 LOTS Visualization Evaluation

In order to assess our proposed LOTS visualization extension, we conducted a comparative analysis with other CAM-based methods across four distinct backbones. The evaluation was based on the performance of the techniques across five metrics. We conducted experiments using four network architectures: AlexNet (Krizhevsky et al., 2012), ResNet-50 (He et al., 2016), DenseNet-121 (Huang et al., 2017) and ConvNext Tiny (Liu et al., 2022). The justification for choosing these architectures was based on their proven success in various computer vision tasks, such as image classification, object detection, and semantic segmentation. Their performance has been benchmarked on large-scale datasets such as ImageNet. Therefore, their widespread use provides a useful reference point for evaluating our proposed approach. Additionally, their distinct underlying network architectures also played a role in the selection process.

Next, We selected the CAM methods to be compared against the LOTS visualization. Part of the criteria for selecting CAM methods was based on their ease of implementation, considering the vast number of available CAM-based algorithms to choose from. Including both older and newer CAM methods in our evaluation, allowed us to compare the performance of newer methods against established methods. This is significant because the expectation is that newer methods should perform better, given that they are typically improvements on the initial CAM methods. Table 5.2 is arranged in order of the methods' age, with the older methods at the top and the newer ones at the bottom of the first column. In the case of all CAM-based methods, it was necessary to select a class-specific target. There were two options available: choosing the true class label as the target, or selecting the highest class probability identified by the network. We decided to adopt the second approach, as our objective was to visualize the highest features rather than the true class-specific features. This choice allowed for a meaningful comparison of the results with LOTS using the zero F_t. Furthermore, for each visualization method, we selected the last convolutional layer to generate our activation maps. To further assess the performance of the proposed methods, we included a Random-CAM method as a baseline to compare against. This allowed us to evaluate if the methods are achieving results beyond what could be expected by chance.

Table 5.2 showcases the quantitative performance of LOTS in object detection using the described dataset subset, in comparison to several chosen CAM-based visualization techniques. Attachment A.2 contains example visualizations that demonstrate the metrics being utilized.

When examining the oldest model, AlexNet, it is observed that LOTS surpasses the other

	AlexNet				
Method	Avg Drop ↓	Avg Increase ↑	Coherency ↑	Insertion ↑	Deletion ↓
Random-CAM	49.11	18.50	55.13	28.72	25.6
Grad-CAM	**18.31**	**41.80**	75.01	39.17	16.73
Grad-CAM++	19.41	39.20	84.92	37.98	16.95
SmoothGrad-CAM++	20.93	35.80	92.06	37.03	17.74
Layer-CAM	20.26	37.60	91.96	39.14	16.21
HiRes-CAM	20.03	38.90	80.98	**41.01**	**15.81**
LOTS	28.96	25.60	**92.47**	34.85	19.39
	ResNet-50				
Method	Avg Drop ↓	Avg Increase ↑	Coherency ↑	Insertion ↑	Deletion ↓
Random-CAM	38.07	25.80	60.45	29.43	26.99
Grad-CAM	15.72	47.40	94.22	34.67	**18.58**
Grad-CAM++	19.17	44.20	93.03	34.29	18.99
SmoothGrad-CAM++	15.66	45.20	92.32	33.22	19.94
Layer-CAM	15.35	47.20	**95.24**	34.09	18.89
HiRes-CAM	14.04	**48.60**	95.19	**34.92**	19.22
LOTS	**11.59**	42.90	90.01	32.18	18.96
	DenseNet-121				
Method	Avg Drop ↓	Avg Increase ↑	Coherency ↑	Insertion ↑	Deletion ↓
Random-CAM	42.55	15.60	56.39	49.77	45.78
Grad-CAM	8.59	43.90	97.90	63.88	**31.07**
Grad-CAM++	9.38	43.60	97.54	63.46	31.55
SmoothGrad-CAM++	9.16	40.90	**98.02**	62.21	32.64
Layer-CAM	8.77	42.70	98.62	63.12	31.53
HiRes-CAM	**8.23**	**45.00**	97.91	**64.48**	31.26
LOTS	14.31	31.20	88.48	59.22	32.07
	ConvNext Tiny				
Method	Avg Drop ↓	Avg Increase ↑	Coherency ↑	Insertion ↑	Deletion ↓
Random-CAM	63.48	4.00	59.40	32.91	30.96
Grad-CAM	39.07	14.80	92.45	37.86	22.47
Grad-CAM++	39.07	12.70	92.40	37.49	22.89
SmoothGrad-CAM++	42.77	12.20	88.05	35.86	24.72
Layer-CAM	**38.11**	12.80	93.07	38.11	**22.26**
HiRes-CAM	40.33	**15.20**	**93.25**	**39.54**	23.55
LOTS	38.99	6.70	52.03	33.15	25.89

Table 5.2: EVALUATION OF DIFFERENT CAM-BASED APPROACHES ALONGSIDE LOTS. *The evaluation was done on 4 different backbones including a Random-CAM for each network architecture. An upward-pointing arrow indicates that the metric should be maximized, while a downward-pointing arrow suggests the metric should be minimized. All the results presented in the table are expressed in percentage.*

methods in terms of Coherency, indicating a stronger performance. However, in all other metrics, LOTS demonstrates relatively poorer results compared to the other methods. The relatively lower performance of LOTS on metrics such as Average Drop, Average Increase, Insertion, and Deletion can be attributed to its approach of visualizing all features in the image, without considering a specific class, using the zero target F_t. In contrast, CAM-based methods focus solely on predefined classes for visualization and highlight features accordingly. When calculating the explanation map based on the generated activation map, this distinction is significant. CAM-based methods retain class-specific features in an image, while LOTS retains all identified features, even across classes. This fundamental difference in approach may explain the contrasting results between LOTS and the CAM-based methods on these metrics.

However, upon analyzing the ResNet-50 network architecture, the previously mentioned argument regarding the differences in Average Drop, Average Increase, Insertion, and Deletion metrics between LOTS and the CAM-based methods, as discussed for AlexNet, does not appear to hold. Surprisingly, LOTS showcases superior performance on the Average Drop metric, suggesting that its activation map effectively eliminates disruptive pixels that hinder the model from predicting the highest class probability, surpassing the performance of other CAM-based methods. However, it is worth noting that LOTS still exhibits the lowest average increase in confidence among all the methods. This implies that while the drop in confidence was generally smaller with LOTS, the other visualization techniques resulted in more instances where the model became more confident based on the explanation map. To verify the reliability of the results, we conducted two separate evaluations of LOTS on the ResNet-50 architecture. Although we could have referred to the results in Table 5.1 for the row corresponding to the zero target F_t, we took this opportunity to assess the stability of the LOTS visualization by ensuring consistent output across the two runs, which indeed proved to be the case.

Moving on to DenseNet-121, the performance of LOTS appears to be comparable to its performance on ResNet-50. However, it seems that the selected metrics may not fully capture the true strengths of LOTS with the zero target F_t, which lies in its ability to capture all features contributing to a classification. The metrics consistently penalize LOTS for detecting features that are not specific to a particular class.

A similar observation can be made for the newest architecture, ConvNext Tiny. In this case, LOTS exhibits instability in generating the same activation map when using the explanation map as input. This is peculiar considering that LOTS consistently produced similar activation maps on the other network architectures. In the case of the ConvNext Tiny architecture, it appears that LOTS was influenced by the presence of newly introduced masked pixels, resulting in a change in the highest class probability. This distraction caused LOTS to generate different activation maps when using the explanation map as input.

Overall, it is important to note that Grad-CAM and HiRes-CAM consistently outperformed the other methods. This is noteworthy as Grad-CAM is the oldest method among those presented, serving as inspiration for the other methods. On the other hand, HiRes-CAM belongs to the newer generation of visualization methods. Despite displaying some indications of good performance for LOTS, further evaluation from a qualitative standpoint is necessary (see section 5.2). Furthermore, comparing the visualization capabilities of LOTS with CAM-based methods is challenging. CAM-based methods do not utilize the concept of using a zero target F_t, meaning that these methods always require selecting a specific class for visualization. This necessitates prior knowledge of what should be visualized in the image. In contrast, LOTS encompasses all the features that contribute to a model's prediction, regardless of class. However, LOTS does offer the option to incorporate class-specific targets F_t (see class F_t in Table 5.1), introducing difficulties visualized in Figure 5.1. To provide an answer to RQ2, the chosen metrics provide insights into the performance of the methods, but evaluating a method like LOTS with the zero target F_t, which does not solely focus on one class feature, presents challenges. There is a need for further

Method \ Model	AlexNet	ResNet-50	DenseNet-121	ConvNext Tiny
Random-CAM	15.42	35.51	68.18	36.57
Grad-CAM	11.28	23.69	54.80	22.08
Grad-CAM++	17.18	33.04	76.43	43.33
SmoothGrad-CAM++	608.73	765.38	1699.67	251.02
Layer-CAM	**10.93**	**21.56**	**48.99**	**20.14**
HiRes-CAM	19.82	35.51	78.86	56.56
LOTS	25488.27	40515.11	63133.59	39143.17

Table 5.3: VISUALIZATION METHODS PERFORMANCE. *The results presented are expressed in milliseconds. Each specific number represents the average duration required by a single method to generate an activation map for an individual image within the dataset subset of 1000 images.*

research and development of suitable metrics that can effectively evaluate visualization methods across different features, without being specific to class features. This would facilitate a more comprehensive and accurate assessment of methods like LOTS.

While not placing excessive emphasis on performance analysis, we included Table 5.3 to provide an overview of the average time taken by each method for a single method call. Although performance may be of greater significance when applying these methods to large-scale datasets, it is of lesser importance to our specific research questions. Nevertheless, we included these performance metrics for the sake of completeness.

Table 5.3 indicates that Layer-CAM exhibits the highest speed among the methods, whereas LOTS demonstrates the slowest performance. Nonetheless, it is worth noting that optimizing LOTS parameters such as the number of iterations $iter$, threshold τ, or step width α can enhance its speed. All methods exhibit stable performance across different architectures, confirming the expected trend of longer execution times for larger network architectures.

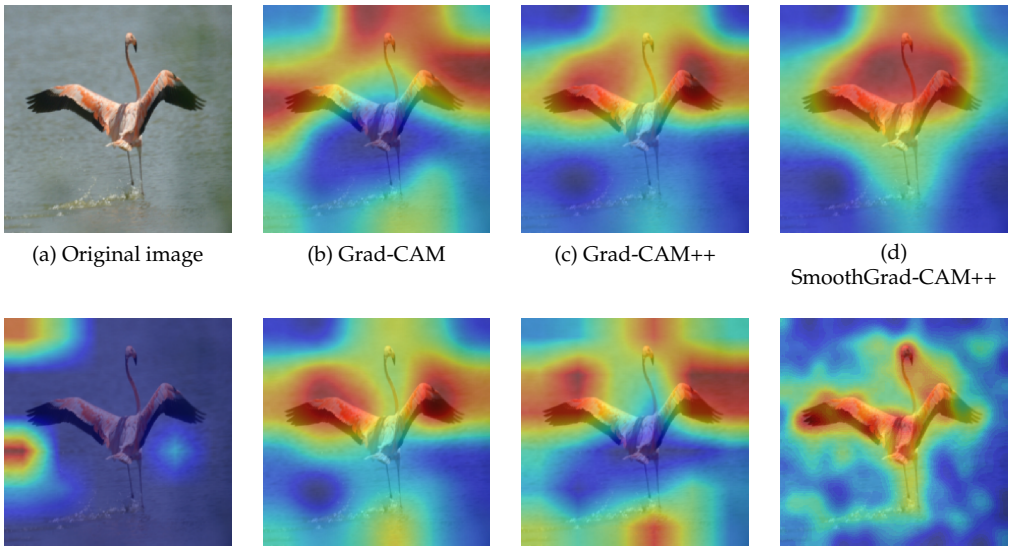
5.2 Qualitative Analysis

In this thesis, we have included a dedicated section on qualitative evaluation of visualization methods to complement the quantitative analysis. While quantitative metrics provide objective measures of performance, they may not capture the complete picture when it comes to evaluating visualization techniques. By incorporating a qualitative evaluation, we gain deeper insights into the strengths and limitations of our LOTS visualization method. We aim to assess the visual quality of the LOTS visualization method in comparison to other CAM-based methods. This approach allows us to consider factors such as the level of detail captured by the activation maps, and the ability to highlight features in the underlying data. Furthermore, qualitative evaluation enables us to uncover potential discrepancies between quantitative metrics and subjective human perception. It provides an opportunity to identify cases where certain methods may outperform others based on the visual information conveyed, despite potentially lower quantitative scores.

This section aims to address research questions RQ1.1, RQ1.2, and RQ3.

5.2.1 LOTS Visualization Comparison

This subsection focuses on evaluating the qualitative performance of LOTS, specifically addressing research questions RQ1.1 and RQ1.2.



(a) Original image (b) Grad-CAM (c) Grad-CAM++ (d) SmoothGrad-CAM++

(e) Random-CAM (f) Layer-CAM (g) HiRes-CAM (h) LOTS

Figure 5.2: VISUAL EXAMPLE ON ALEXNET. *In this Figure, the activation heatmaps generated by five distinct CAM-based methods are superimposed on the original image. These heatmaps correspond to the target class Flamingo, which achieved the highest score of 95.81% on the AlexNet model. Subfigure 5.2(e) depicts the visualization generated through random guessing, exhibiting significant deviation from the actual Flamingo. On the other hand, LOTS showcases its ability to capture a larger portion of the Flamingo compared to the other methods (Subfigure 5.2(h)). Notably, no modifications were made to the predefined parameters during this analysis. Best viewed in color.*

In the subsequent two subsections, we utilized images from the subset of the ImageNet validation set. This subset was employed for the quantitative analysis, as outlined in section 5.1.

Visualizations with larger areas

In this section, we focus on research question RQ1.1, which explores the possibility of expanding the LOTS method to generate visualizations with larger areas, similar to CAM-based methods. To conduct a qualitative analysis, we selected one image from the validation subset for each network architecture and visualized the activation heatmap on the original image. We used the same network architectures as in section 5.1. For the LOTS method, we used the zero target F_t, while for the CAM-based methods, we used the highest prediction class score as target. To ensure comparability with CAM-based methods, we kept all parameters unchanged, as described in section 5.1.

Figure 5.2 presents a comparison between our LOTS visualization method and the CAM-based methods on the AlexNet model. The attachments A.1 showcase the visualizations on the same example using the three other architectures. Our LOTS method captures more details of the flamingo and produces a more precise boundary around the flamingo object compared to the CAM-based methods. However, in this specific instance, LOTS exhibited a Drop in Confidence of 6.99%, while HiRes-CAM demonstrated a Drop in Confidence of 3.21% (see Subfigure 5.2(g)). This example highlights a case where the Average Drop metric fails to adequately describe the quality of a visualization. The reason behind the smaller drop in confidence for HiRes-CAM could be attributed to its focus on class-specific features, selectively retaining only those relevant

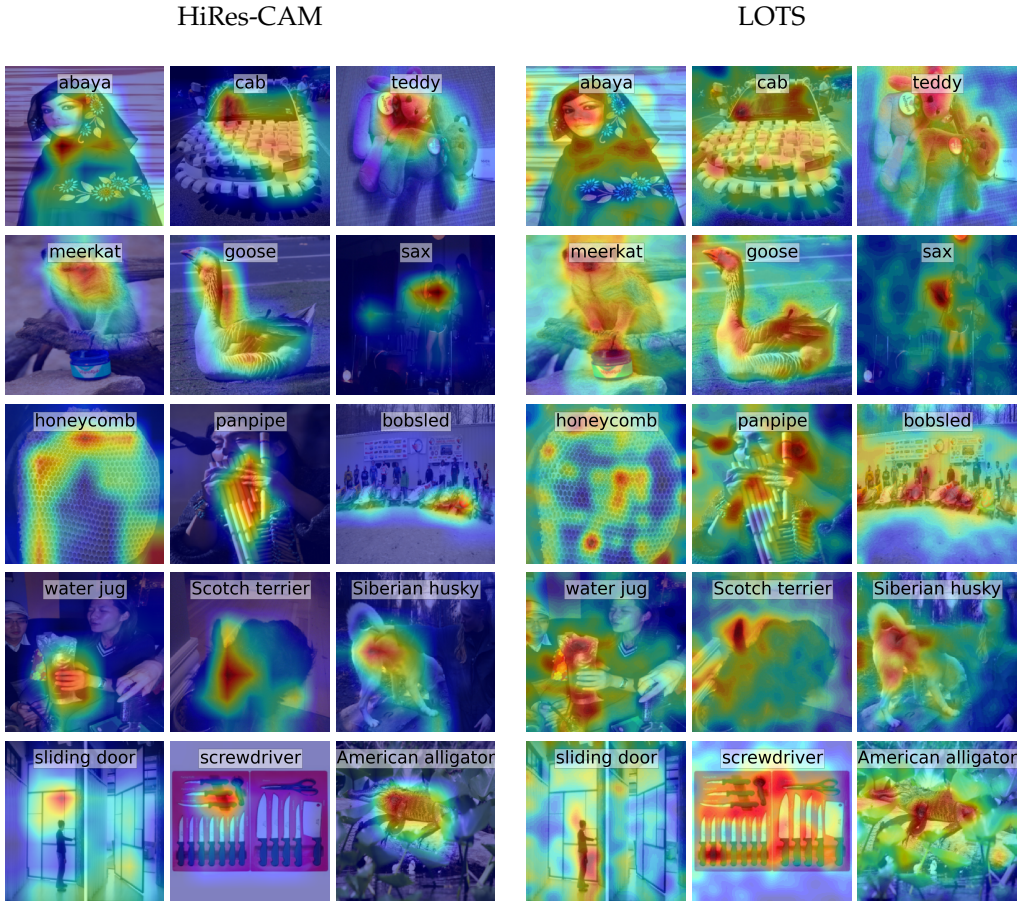


Figure 5.3: HiRes-CAM and LOTS activation heatmaps generated with ResNet-50. *The Figure displays 15 randomly-selected images from our ImageNet validation subset. It is divided into two halves, each containing the same images and classes. The classes represented are the highest predicted classes and may not necessarily correspond to the true labels of the images. The only difference between the two halves lies in the calculation of the activation maps. Best viewed in color.*

to the prediction, whereas LOTS includes all features contributing to any increase in prediction.

To perform a qualitative evaluation of ResNet-50, we focused exclusively on comparing HiRes-CAM with LOTS. HiRes-CAM demonstrated superior performance across all metrics discussed in section 5.1, particularly for the ResNet-50 and DenseNet-121 network architectures. Figure 5.3 displays 15 randomly selected examples from our ImageNet validation subset. The results indicate that LOTS is indeed visualizing features that go beyond a single class. Even among the small randomly selected examples, it is evident that there are numerous instances where multiple classes can be considered plausible, leading to different visualizations. For instance, an image is predicted as containing a cab with the ground truth being toilet paper (due to several rolls of toilet paper affixed to the car), or an image not only containing a panpipe but also other recognizable classes such as a microphone. Similarly, a water jug is wrapped in a plastic bag, which are both classes in the ImageNet dataset, adding further complexity to the visualization. Lastly, it's worth noting that one of the images belongs to the screwdriver class, but only contains knives. In this case, LOTS appropriately focuses on various instances, considering features beyond just the

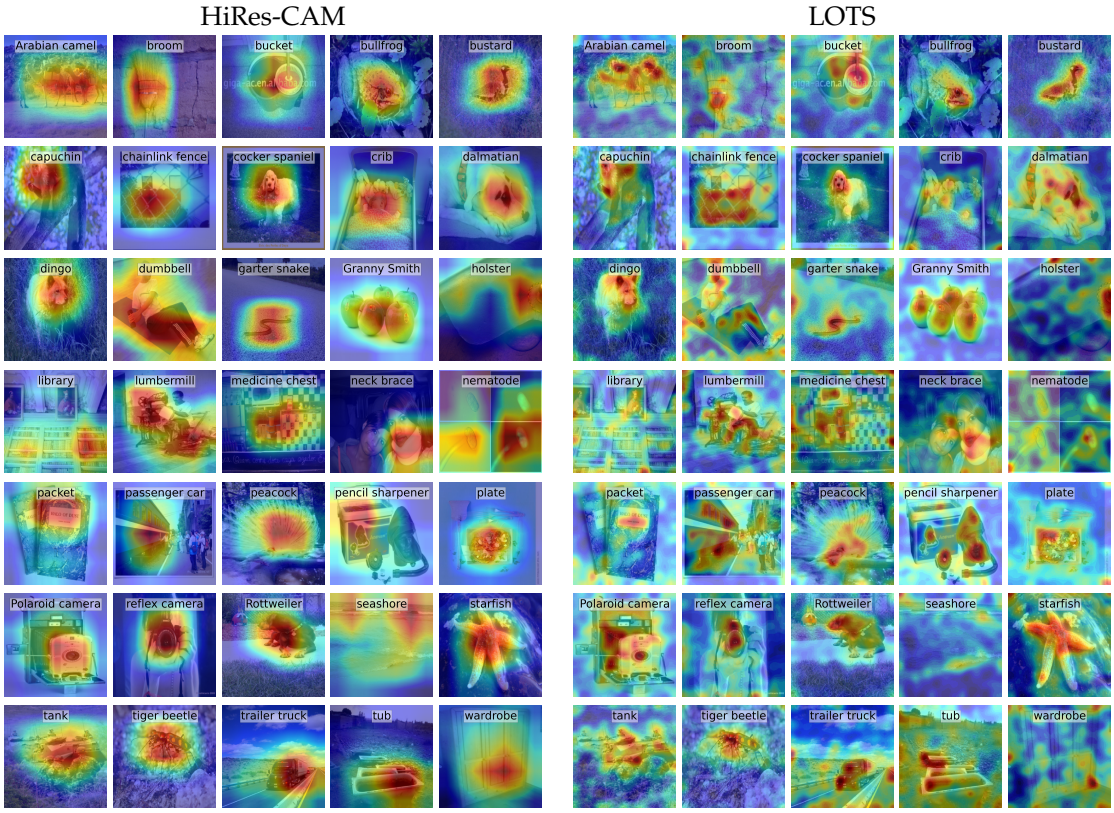


Figure 5.4: HiRes-CAM and LOTS activation heatmaps generated with DenseNet-121. *The figure displays 35 randomly-selected images from the ImageNet validationset subset. It is divided into two halves, each containing the same images and classes. The classes represented are the highest predicted classes and may not necessarily correspond to the true labels of the images. The only difference between the two halves lies in the calculation of the activation maps. Best viewed in color.*

highest predicted class. On the other hand, HiRes-CAM should not have highlighted anything, emphasizing the significance of being able to visualize the absence of relevant features when the predicted class is not present. The depicted Figure 5.3 further illustrates that the CAM-based method exhibits a more concentrated focus compared to LOTS. Specifically, the generated activation heatmap in the CAM-based method does not spread out extensively across the entire image. In contrast, LOTS exhibits a different behavior where, although the primary activation focus is visible, there are numerous smaller activations surrounding the identified object. While the focus of LOTS may not be crucial for human recognition, it can potentially impact the introduced metrics in section 5.1.

Figure 5.4 aims to showcase the overall contrast between LOTS and the HiRes-CAM technique in generating activation maps. It is observed that LOTS exhibits a higher level of detail when examining the overall heatmaps. In contrast, HiRes-CAM captures a broader perspective but lacks suitability for finer regions. Figure 5.5 provides a clearer comparison between the LOTS method applied to DenseNet-121 and ConvNext Tiny. In cases where the predicted class did not change from DenseNet to ConvNext, the visualization appears more detailed for the corresponding examples. This observation emphasizes the significance of the underlying model in generating visualizations that minimize noisy pixel activations for LOTS. Furthermore, this example visu-

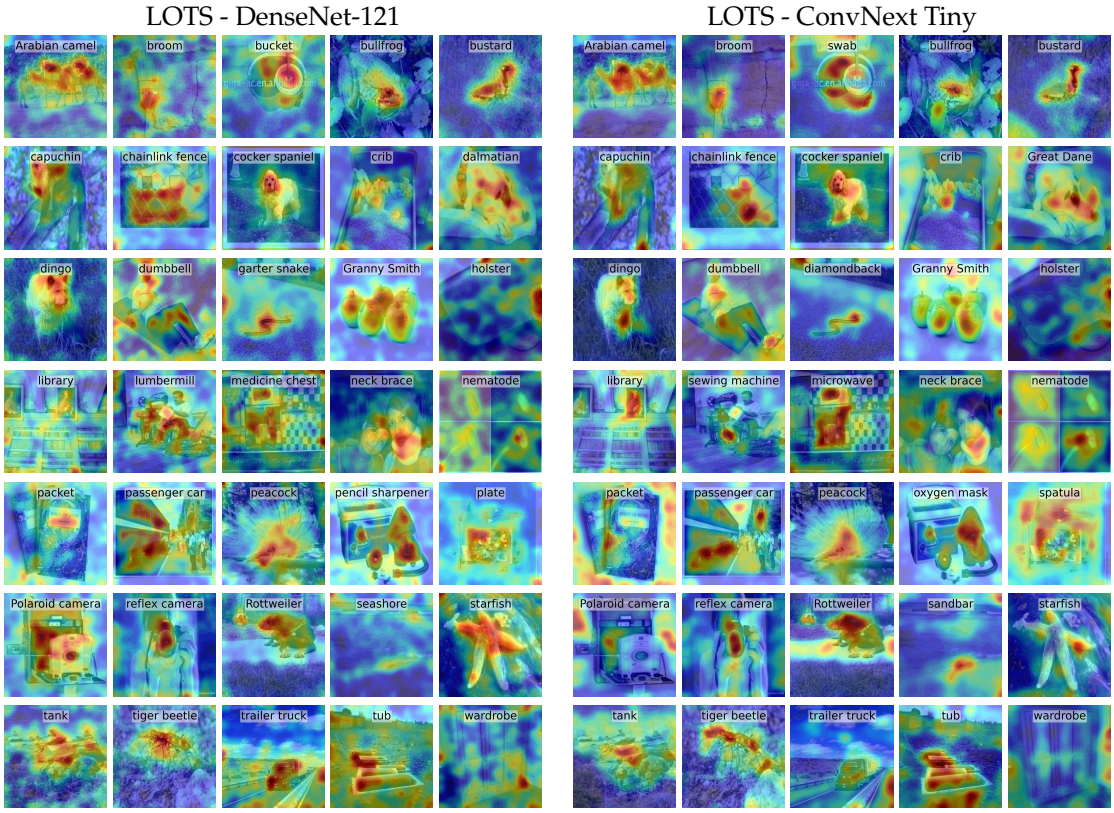


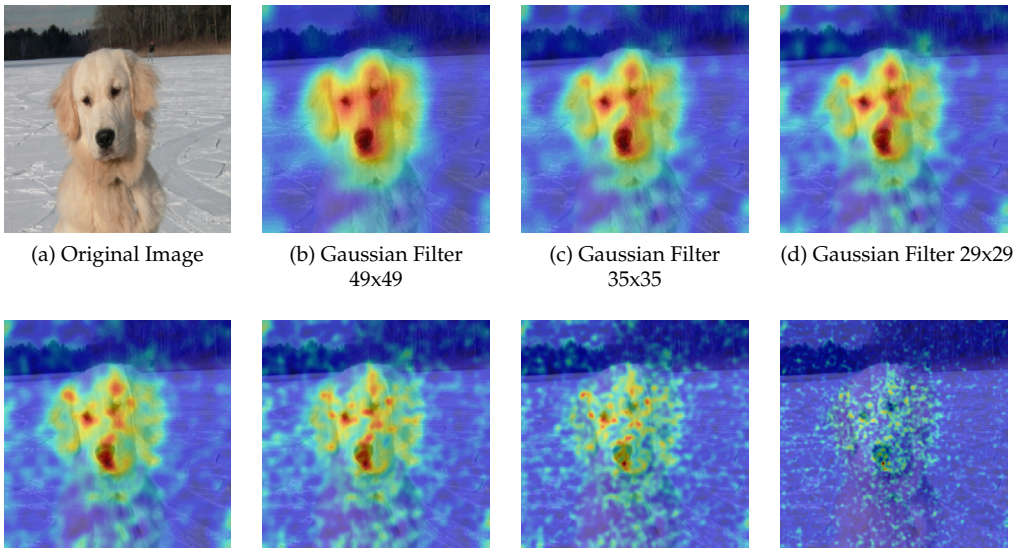
Figure 5.5: LOTS VISUALIZATIONS ON DENSENET-121 COMPARED TO CONVNEXT TINY. *The figure displays the same 35 randomly-selected images as in Figure 5.4. It is divided into two halves, containing the same images but not necessarily the same classes. The classes represented are the highest predicted classes. The only difference between the two halves lies in the underlying model architecture. Best viewed in color.*

alization can also serve as an explanation for why LOTS performed significantly poorer on the ConvNext architecture in terms of Coherency. When the generated activation map on the original image already captures a substantial amount of details, even a minor shift in activations on the explanation map leads to a considerably lower Coherency compared to an initial activation map that captures fewer details. Across all the generated activation maps, it is intriguing to observe that the LOTS method with a zero target F_t effectively visualizes features from various classes if there are multiple present (e.g., Sandbar vs. Seashore). Therefore, providing an answer to RQ1.1, LOTS not only provides clearer indications of where its focus lies, but also excels at identifying multiple instances of a single class within the same image.

Visualization for fine locations

Although the LOTS method already surpasses CAM-based methods in terms of visualizing more detailed features, there remains room for further improvement to enhance its focus specifically for RQ1.2. As mentioned in the section 4.3, our approach is to reduce the size of the Gaussian blur filter, resulting in activation maps that exhibit greater pixel-level focus.

Figure 5.6 illustrates the impact of using smaller Gaussian blur filter sizes compared to our



(a) Original Image (b) Gaussian Filter 49x49 (c) Gaussian Filter 35x35 (d) Gaussian Filter 29x29

(e) Gaussian Filter 23x23 (f) Gaussian Filter 17x17 (g) Gaussian Filter 11x11 (h) Gaussian Filter 5x5

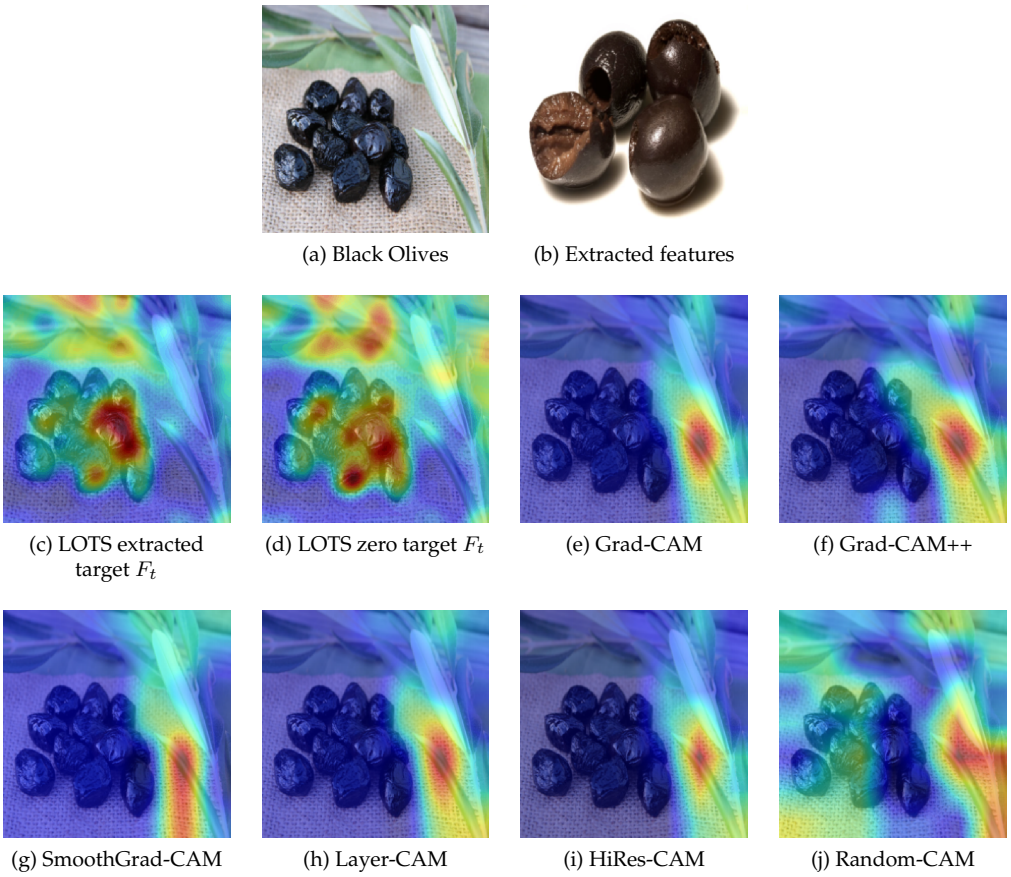
Figure 5.6: FINE GRAINED LOCALIZATION. *Within this Figure, we present several visualizations that demonstrate the impact of utilizing a smaller Gaussian blur filter size with our LOTS visualization. As the filter size decreases, the visualization becomes increasingly pixel-specific and detailed, highlighting finer aspects of the data. Example from DenseNet-121. Best Viewed in colour.*

standard (49x49) on the LOTS visualization. With decreasing filter size, the visualization becomes increasingly detailed at the pixel level. However, it is important to note that as the filter size decreases, noise is introduced into the visualization. This noise most probably originate from the model itself rather than being a direct result of LOTS. Since LOTS identifies pixels that contribute to the model's increase in confidence in any of the 1000 classes, it highlights the corresponding image regions. Hence, it appears to be a model-specific issue that is not evident when employing CAM-based visualization techniques. In this particular example, the region surrounding the dog's nose emerges as the most influential area in the model's prediction, as depicted in Subfigure 5.6(h). Therefore, in order to address RQ1.2 and highlight fine locations in an image, an option is to reduce the size of the Gaussian blur filter used in the LOTS method.

5.2.2 LOTS on examples not present in training dataset

In order to address RQ3 regarding the applicability of LOTS to images not present in the training dataset of the model, we once again compared the zero target F_t with a class-specific target F_t derived from another image featuring the same object. Additionally, provided a comparison to existing CAM-based techniques.

Figure 5.7 presents the finding. Despite the ConvNext model assigning the highest confidence to the class "pot," which exists within the 1000 classes and bears some resemblance to the plant on the right side of the image, LOTS and CAM-based techniques yielded entirely distinct visualizations. Both LOTS targets emphasized the black olives and the upper portion of the plant (see Subfigures 5.7(c) and 5.7(d)), whereas all CAM methods focused on the lower section of the plant. Despite the distinct focus of LOTS and CAM techniques, which is attributed to LOTS not filtering



(a) Black Olives (b) Extracted features

(c) LOTS extracted target F_t (d) LOTS zero target F_t (e) Grad-CAM (f) Grad-CAM++

(g) SmoothGrad-CAM (h) Layer-CAM (i) HiRes-CAM (j) Random-CAM

Figure 5.7: LOTS COMPARISON FOR CLASS NOT PRESENT IN IMAGENET. *The visualizations presented in this Figure depict the results obtained from LOTS and CAM-based techniques, applied to an object that is not included in the ImageNet training dataset. The visualization 5.7(c) was generated using LOTS, utilizing deep features extracted from 5.7(b) as the target feature representation, denoted as* F_t.

for class-specific features, it is anticipated that the visualization of the plant would exhibit similar behavior in both methods. In this specific example, LOTS proved to be the only technique capable of capturing the presence of black olives. However, drawing conclusions from a single example, or even multiple examples, is challenging. To achieve meaningful insights, a large-scale analysis with defined metrics would be necessary. Conducting such an analysis poses difficulties as the model itself lacks knowledge of the correct answer, requiring manual annotation to identify the correct location within the image. Furthermore, the process of drawing bounding boxes presents its own challenges, as determining whether the entire object or only specific parts need to be included can be a complex task in itself, as described in section 3.3. While this example provides a promising indication of the possibility, it is crucial to emphasize that answering RQ3 requires extensive further research to provide a comprehensive and reliable response.

Chapter 6

Discussion

6.1 Experimental Shortcomings

While designing the experiments to evaluate LOTS, several decisions needed to be made. One crucial decision was determining the specific task to compare against, which was closely tied to the selection of the dataset. In our case, we opted to compare LOTS using the ImageNet dataset, commonly employed for object detection. However, we could have pursued a different direction, such as Pose Estimation or Image Generation, which would have necessitated selecting a distinct dataset for experimental purposes.

Furthermore, we made the decision to utilize a subset of 1000 images from the larger pool of 50,000 samples in the ImageNet validation set. This choice was primarily motivated by the desire to speed up calculations and enable a greater number of experiments to be conducted within a shorter timeframe. Nevertheless, employing a larger dataset for testing purposes would have yielded even more dependable results. During the course of our experiments, it became evident that the underlying dataset contained a considerable number of inadequate images. This issue has also been addressed and investigated by Kertész (2021). Their findings revealed various problems within the dataset, including incorrect labels, equivalent categories, images with minuscule objects, poor-quality photographs, samples associated with multiple correct categories, and images containing multiple objects. These issues with the dataset's quality and labeling pose challenges and considerations that need to be taken into account when interpreting the results and evaluating the performance of the models.

Due to practical constraints, we had to restrict ourselves to a limited number of models for testing purposes. However, to ensure the reliability and generalizability of our experiments, it would be crucial to evaluate the LOTS visualization method against a more diverse range of models. By testing LOTS across various models, we gain a better understanding of its performance across different architectures, dataset biases, and model complexities. This expanded evaluation would provide a more comprehensive and robust assessment of the LOTS visualization technique.

To evaluate the performance of the visualization methods, we attempted to incorporate automated metrics as a means of assessment. However, an alternative approach could have involved conducting human evaluations with a diverse set of participants. However, in such evaluations, a significant challenge arises in determining the criteria for successful object detection. For instance, in the case of detecting a dog in an image, should the algorithm be considered successful if it captures the entire dog, including its head, body, legs, and tail? Or would it suffice to only capture the pixels that contribute to distinguishing the dog class from other classes? The complexity of the task is further amplified when considering the numerous dog breeds, each requiring distinct visualizations to understand the specific characteristics leading to classification. In this scenario, visualizing the entire body of the dog may not be informative enough, as it would not reveal

the specific features responsible for the classification. Instead, a more specific and targeted visualization approach would be necessary. In contrast, if the dataset solely contains a single dog class, a broader representation of the object being highlighted may be sufficient for successful visualization, as there are no additional dog breeds to differentiate. The visualization approach should adapt accordingly based on the specific requirements of the task and the complexity of the dataset.

The process of selecting appropriate metrics poses a challenge in the field of XAI due to its relatively emerging stage, lacking widely accepted metrics within the community. Consequently, comparing different visualization methods becomes difficult. We observed instances where qualitative visualizations appeared significantly superior to what the quantitative metrics indicated. Hence, future evaluations should focus on taking significant strides towards unifying metrics to address this issue and foster better evaluation practices in XAI.

Another limitation that may have contributed to less conclusive results was related to the Average Drop/Increase metric. In this metric, we chose to represent non-activated pixels in the activation map as black in the explanation map. Additionally, we incorporated the Coherency metric to assess whether the visualization method solely focused on the object itself or shifted attention to the newly introduced black pixels. Except for ConvNext Tiny in the case of LOTS performance, it appeared that the introduction of black pixels did not generate new artifacts or distract the visualization method. We could have alternatively blurred the pixel values, as we did for the Insertion/Deletion metric. However, we intentionally opted against this approach to maintain consistency with the metric setup proposed by Chattopadhyay et al. (2018).

When conducting experiments for any deep learning task, it is crucial to determine the parameters to be used. In most cases, default parameter settings were adopted throughout our experiments. However, parameter optimization could lead to a different outcome of the experiments.

Furthermore, it is worth noting that the CAM-based methods exhibited slight variations in their visualizations depending on the specific implementation used. As a result, there were minor differences in the visual outputs. Additionally, it is important to consider the comparability between LOTS with the zero target F_t and CAM-based methods. LOTS highlights all features that influence the model's prediction, whereas CAM-based methods solely enable class-specific visualizations. This discrepancy raises questions about the direct comparability between the two approaches.

Regarding the target selection in LOTS, there exists a multitude of potential targets that could enhance the visualization outcomes. However, due to practical constraints, we had to make a decision on the most obvious choices. In future evaluations, an alternative approach to target selection could involve extracting deep features for all classes and for each image with a specific class, use another F_t corresponding to another class. This methodology could potentially yield stronger visualizations, particularly in regions where the class present in the image exhibits differences from the chosen target class. By exploring such target selection strategies, we can potentially uncover more nuanced and informative visual representations using the LOTS technique.

6.2 Revisit LOTS Visualization Algorithm

The algorithm described in Algorithm 2 utilizes the Gaussian blur filter as the primary technique to generate perturbations on the activation map. However, it is important to acknowledge that this approach often leads to the presence of noisy activations surrounding the main focus. This noise can be attributed to the models themselves, as they might be considering too many pixels simultaneously. To address this issue, there are a couple of potential solutions. To begin with, one way to enhance the models is by providing higher quality images that would enable the

model to focus on more specific pixel areas for its prediction. By doing so, LOTS would then generate perturbations that are more focused and concentrate on the key areas of interest. This improvement could result in cleaner and more accurate visualizations without excessive noise, which some examples already demonstrated (see section 5.2. Alternatively, a post-processing step could be implemented to remove activations below a certain threshold. By setting a threshold, activations deemed insignificant or noisy could be filtered out, leading to a cleaner and more refined visualization output. This thresholding technique can help reduce the impact of noise and improve the overall quality of the visualizations.

There are additional areas for improvement that relate to the selection of parameters in LOTS. The LOTS algorithm offers several parameters, such as the threshold τ, step width α, and number of iterations $iter$. Optimizing these parameters has the potential to enhance the efficiency of the visualization technique while maintaining or even improving its quality.

The limitation of our LOTS algorithm with zero target F_t arises in situations where we possess prior knowledge about the presence of a specific class and aim to visualize only the pixels that influenced the classification of that particular class. In such cases, CAM-based methods demonstrate their superiority. These methods excel in precisely highlighting the relevant pixels associated with a specific class, providing more targeted and class-specific visualizations compared to our algorithm. However, there is potential for optimizing LOTS with class targets F_t.

6.3 Other Use Case

Through our experiments, we have demonstrated that LOTS, in general, places greater emphasis on specific pixels rather than generating a circular activation map around the pixels of interest. This characteristic could be particularly valuable in other real-life situations where a more precise visualization is necessary, such as the example of an x-ray image containing a tumor.

While the initial motivation behind our work was to develop a method capable of highlighting more detailed parts of an image, we have successfully demonstrated through LOTS that this objective can be qualitatively achieved. However, there is another valuable application for this visualization method, which is its potential use as a image quality assessing tool. By employing LOTS with the zero target F_t, we can highlight all the pixels that contribute to any model prediction. In contrast to CAM-based methods, which require prior knowledge of the class present in an image to emphasize its features, this approach is distinct. To evaluate the image quality, one approach is to examine the visualization produced by the LOTS method and observe the pixel areas it highlights. This analysis helps determine whether the model accurately predicts class-specific pixels or becomes distracted by pixels outside the intended class. However, conducting such an assessment requires human judgment and expertise. The primary difference to CAM-based models is that we no longer require prior knowledge about the presence of a particular class in the image. Instead, we compare the image F_s to the zero target F_t. Therefore, LOTS serves as a valuable tool for diagnosing and improving deep learning models by providing insights into the features utilized by the model for classification.

LOTS has also the potential to serve as a tool for determining whether an image is appropriate for enhancing model confidence or if it introduces noise. This can be accomplished by assessing whether an image exclusively presents features from a single class, or if it might contain features from other recognized classes. This, once again, underscores the need for human judgment in the process.

Chapter 7

Conclusion

In this thesis, we introduced the LOTS algorithm and provided details about the network architecture and dataset employed in our thesis. Our first major contribution was the extension of the LOTS algorithm to enable visualizations. Specifically, we demonstrated how the LOTS visualization technique can be applied to broader regions, resembling the approach used in CAM-based methods. Furthermore, we showcased its capability to generate pixel-specific visualizations, allowing for fine-grained analysis of the model's decision-making process. We arrived at the conclusion that utilizing the zero target F_t is the most suitable approach for our LOTS visualization technique.

As our second contribution, we proposed a method to evaluate visualization techniques by employing a set of metrics and comparing their performance across various model architectures. To facilitate this evaluation, we implemented five distinct metrics and calculated their values for all the methods considered, using a subset of the ImageNet validation set. Our findings also emphasized the necessity of establishing a unified framework for evaluating visualization methods to enable meaningful comparisons between them. We observed variations in the results across different models, indicating the importance of standardizing the evaluation process to ensure fair and accurate comparisons between visualization techniques. Additionally, we highlighted the need to establish a clear objective in determining what constitutes a "good" visualization. It is crucial to define whether a "good" visualization should focus on capturing the entire object or highlighting important features within the image.

Third, we investigated whether the LOTS visualization technique could be extended to classes that are not included in the training set of a given dataset. To provide a definitive answer to this question, further qualitative experiments would be necessary. Alternatively, incorporating a reliable metric capable of assessing whether the visualization accurately identified the new object or misclassified a known object would be valuable in determining the performance of extending LOTS to unseen classes.

Finally, it is worth emphasizing that the proposed LOTS visualization method does not require any prior knowledge of the underlying classes. Unlike CAM-based methods, it can classify and visualize what the model perceives without the need to focus on a predefined class. This characteristic proves to be highly advantageous when aiming to enhance the quality and interpretability of deep learning models.

We see several ways in which this work could be extended in the future. On the one hand, more work could be invested in finding better hyperparameters. In addition, more experiments with different targets F_t for LOTS should be done, and a unified metrics framework should be proposed to be able to compare visualization methods. Furthermore, there is a potential need for improving model performance to enhance the quality of visualizations. When considering the application of such methods in critical domains like medicine, where lives are at stake, it becomes crucial for doctors to have complete confidence in the outcomes provided by these visualizations.

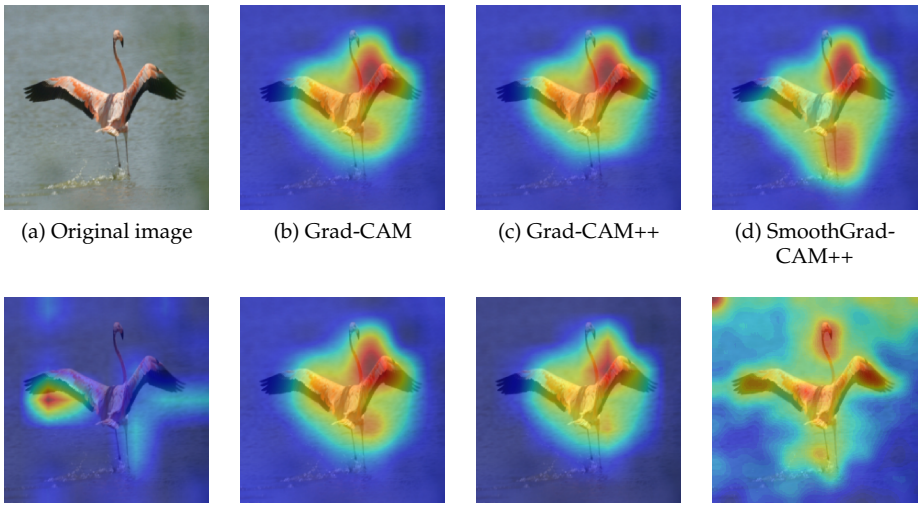
Ensuring their trustworthiness and reliability becomes crucial in leveraging these techniques for life-saving purposes.

Appendix A

Attachments

A.1 CAM-based Methods Compared to LOTS

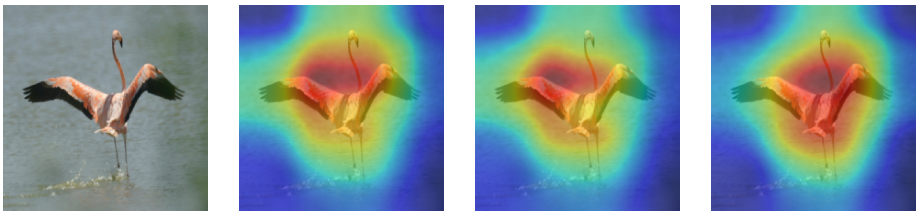
The following visualizations serve as additional examples on how the visualizations performed for ResNet-50, DenseNet-121 and ConvNext Tiny (see Section 5.2).



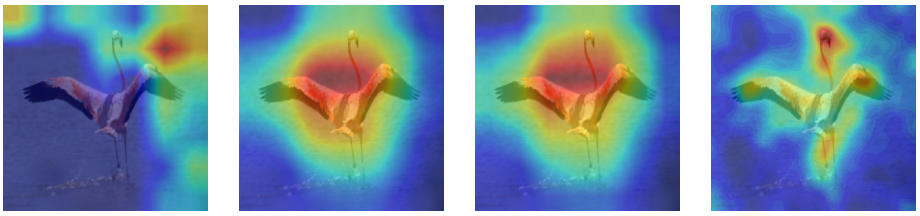
(a) Original image (b) Grad-CAM (c) Grad-CAM++ (d) SmoothGrad-CAM++

(e) Random-CAM (f) Layer-CAM (g) HiRes-CAM (h) LOTS

Figure A.1: VISUALIZATION EXAMPLE ON RESNET-50. *In this Figure, the activation heatmaps generated by five distinct CAM-based methods are superimposed on the original image. These heatmaps correspond to the target class Flamingo. Subfigure A.1(e) depicts the visualization generated through random guessing, exhibiting significant deviation from the actual Flamingo. LOTS is depicted in Subfigure A.1(h). Notably, no modifications were made to the predefined parameters during this analysis. Best viewed in color.*

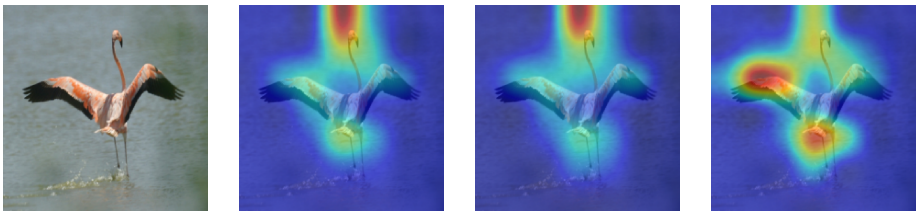


(a) Original image (b) Grad-CAM (c) Grad-CAM++ (d) SmoothGrad-CAM++

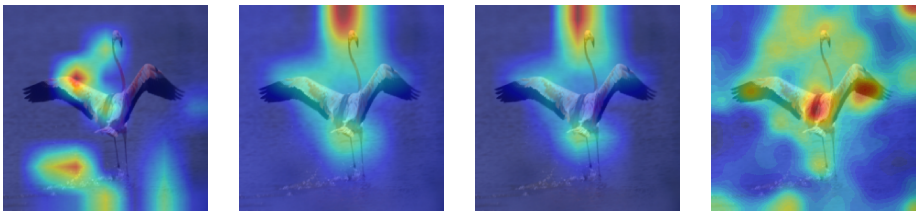


(e) Random-CAM (f) Layer-CAM (g) HiRes-CAM (h) LOTS

Figure A.2: VISUALIZATION EXAMPLE ON DENSENET-121. *In this Figure, the activation heatmaps generated by five distinct CAM-based methods are superimposed on the original image. These heatmaps correspond to the target class Flamingo. LOTS is depicted in Subfigure A.2(h). Notably, no modifications were made to the predefined parameters during this analysis. Best viewed in color.*



(a) Original image (b) Grad-CAM (c) Grad-CAM++ (d) SmoothGrad-CAM++

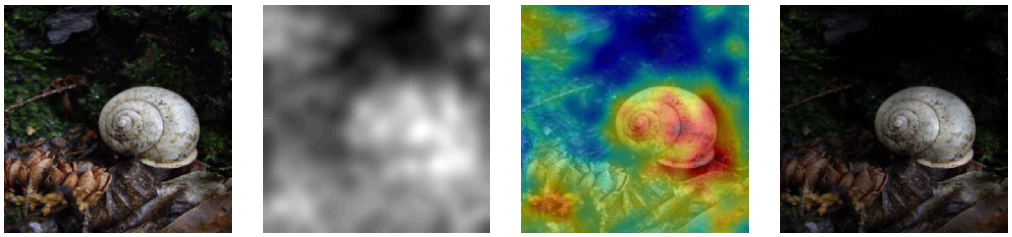


(e) Random-CAM (f) Layer-CAM (g) HiRes-CAM (h) LOTS

Figure A.3: VISUALIZATION EXAMPLE ON CONVNEXT TINY. *In this Figure, the activation heatmaps generated by five distinct CAM-based methods are superimposed on the original image. These heatmaps correspond to the target class Flamingo. LOTS is depicted in Subfigure A.3(h). Notably, no modifications were made to the predefined parameters during this analysis. Best viewed in color.*

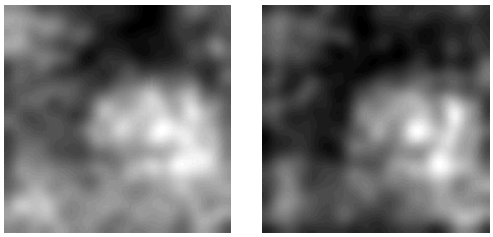
A.2 Metrics Example Visualization

This serves as a visualization example on the five metrics in section 3.3.



(a) Original Image (b) Activation Map (c) Activation Heatmap (d) Explanation Map

Figure A.4: DROP/INCREASE IN CONFIDENCE VISUALIZATION EXAMPLE. *The process depicted in the Figure illustrates the calculation of the initial prediction confidence with Subfigure A.4(a) (88.52% snail). Subsequently, an Activation Map is generated using LOTS (Subfigure A.4(b)), and an explanation map is constructed based on it (Subfigure A.4(d)). The prediction is then made again using the explanation map, resulting in a prediction confidence of 49.06% snail, hence a Drop in Confidence.*



(a) Activation Map (b) Activation Map on Explanation Map

Figure A.5: COHERENCY VISUALIZATION EXAMPLE. *The Figure demonstrates two scenarios. Subfigure A.5(a) displays the activation map generated on the original image using LOTS, while Subfigure A.5(b) depicts the activation map calculated with the explanation map as input. The resulting Coherency score is 94.59%.*

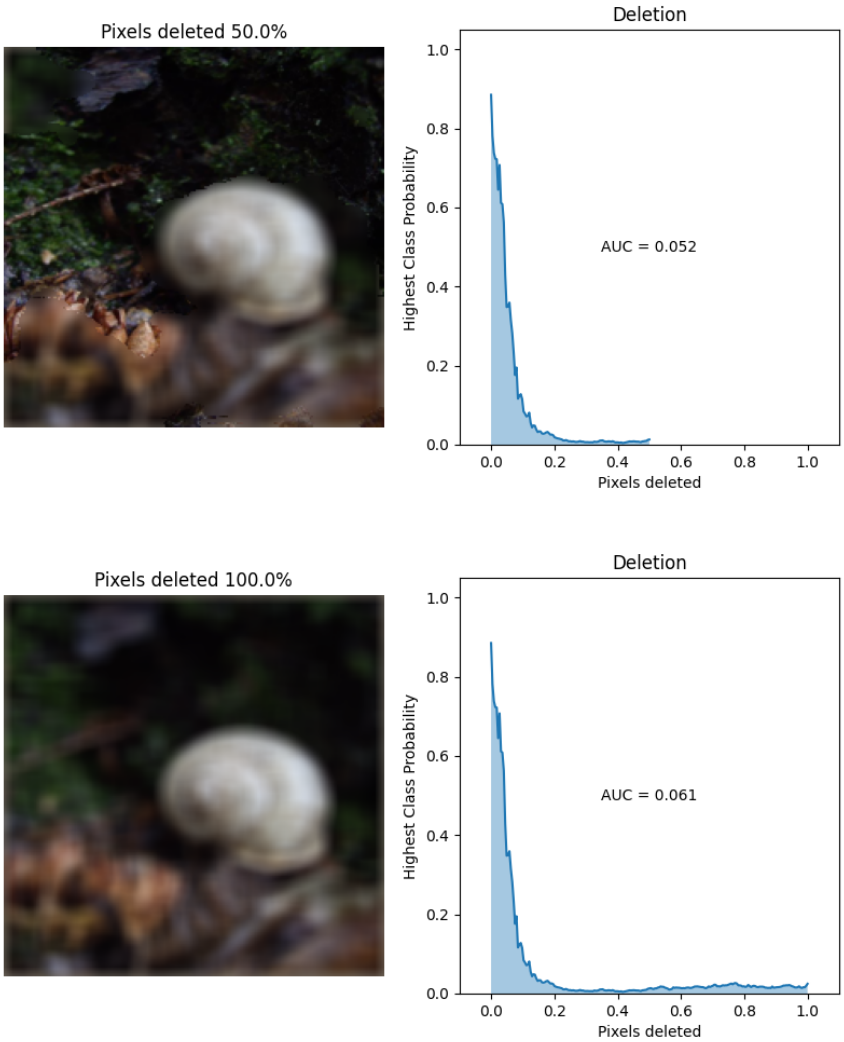


Figure A.6: DELETION VISUALIZATION EXAMPLE. *The image undergoes a gradual masking process, where pixels are systematically removed based on their decreasing importance. The resulting highest class probability predicted by the network is then plotted against the fraction of removed pixels. The objective is to achieve a lower AUC.*

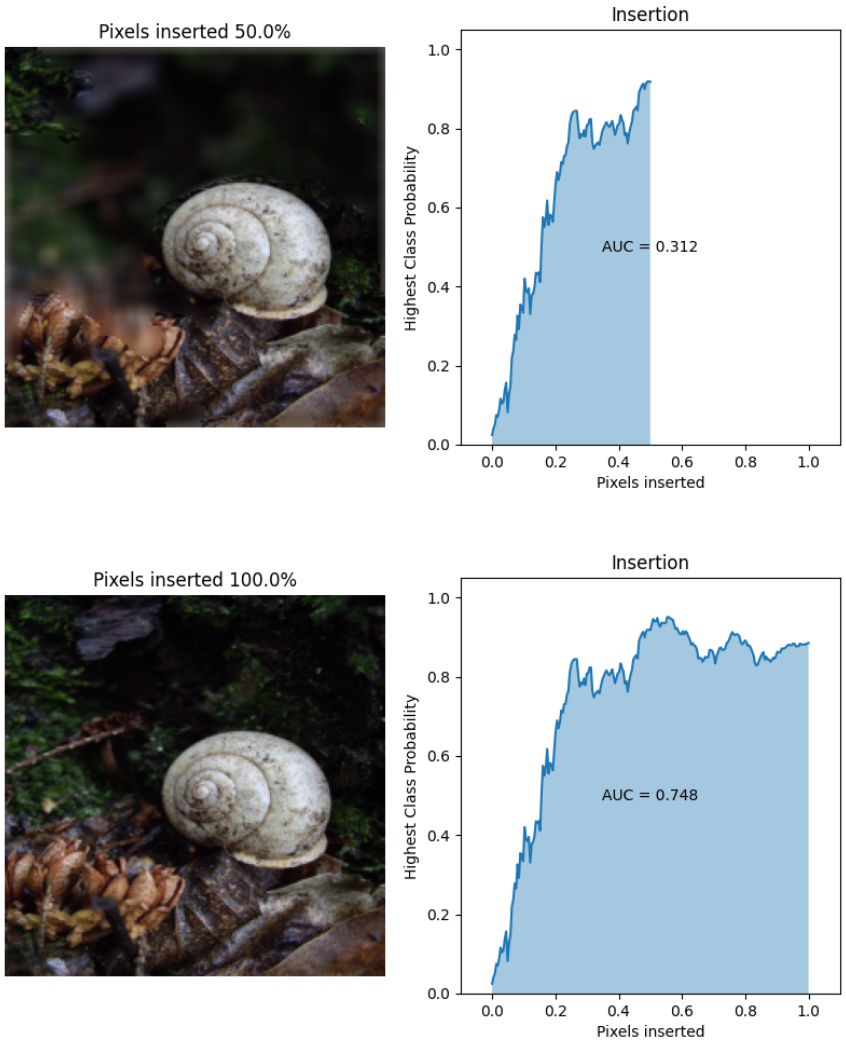


Figure A.7: INSERTION VISUALIZATION EXAMPLE. *The process involves gradually replacing blurred pixels with the original pixels, starting from the most important and progressing to the least important ones. The resulting highest class probability predicted by the network is plotted against the fraction of inserted pixels. The objective is to achieve a higher AUC.*

List of Figures

List of Tables

Bibliography

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A. Q., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., and Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *J. Big Data*, 8(1):53.

Ancona, M., Ceolini, E., Öztireli, C., and Gross, M. H. (2019). Gradient-based attribution methods. In Samek, W., Montavon, G., Vedaldi, A., Hansen, L. K., and Müller, K., editors, *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, volume 11700 of *Lecture Notes in Computer Science*, pages 169–191. Springer.

Chattopadhyay, A., Sarkar, A., Howlader, P., and Balasubramanian, V. N. (2018). Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018, Lake Tahoe, NV, USA, March 12-15, 2018*, pages 839–847. IEEE Computer Society.

Chavannes, N. (2022). Multi-target adversarial attacks with lots. Master's thesis, University of Zurich.

Dabkowski, P. and Gal, Y. (2017). Real time image saliency for black box classifiers. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6967–6976.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255.

Desai, S. and Ramaswamy, H. G. (2020). Ablation-CAM: Visual explanations for deep convolutional network via gradient-free localization. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 972–980.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Draelos, R. L. and Carin, L. (2020). Use HiResCAM instead of Grad-CAM for faithful explanations of convolutional neural networks. *arXiv e-prints*, page arXiv:2011.08891.

Fong, R. C. and Vedaldi, A. (2017). Interpretable explanations of black boxes by meaningful perturbation. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 3449–3457. IEEE Computer Society.

Fu, R., Hu, Q., Dong, X., Guo, Y., Gao, Y., and Li, B. (2020). Axiom-based Grad-CAM: Towards Accurate Visualization and Explanation of CNNs. In *31st British Machine Vision Conference 2020, BMVC 2020, Virtual Event, UK, September 7-10, 2020*. BMVA Press.

Fujiyoshi, H., Hirakawa, T., and Yamashita, T. (2019). Deep learning-based image recognition for autonomous driving. *IATSS Research*, 43(4):244–252.

Gildenblat, J. and contributors (2021). Pytorch library for CAM methods. https://github.com/jacobgil/pytorch-grad-cam.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015). Explaining and harnessing adversarial examples. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Gur, S., Ali, A., and Wolf, L. (2021). Visualization of supervised and self-supervised neural networks via attribution guided factorization. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 11545–11554. AAAI Press.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Hedström, A., Weber, L., Krakowczyk, D., Bareeva, D., Motzkus, F., Samek, W., Lapuschkin, S., and Höhne, M. M. M. (2023). Quantus: An Explainable AI Toolkit for Responsible Evaluation of Neural Network Explanations and Beyond. *Journal of Machine Learning Research*, 24(34):1–11.

Howard, A., Pang, R., Adam, H., Le, Q. V., Sandler, M., Chen, B., Wang, W., Chen, L., Tan, M., Chu, G., Vasudevan, V., and Zhu, Y. (2019). Searching for mobilenetv3. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 1314–1324. IEEE.

Hu, J., Shen, L., and Sun, G. (2018). Squeeze-and-excitation networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7132–7141.

Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2261–2269. IEEE Computer Society.

Ivanovs, M., Kadikis, R., and Ozols, K. (2021). Perturbation-based methods for explaining deep neural networks: A survey. *Pattern Recognit. Lett.*, 150:228–234.

Jiang, P., Zhang, C., Hou, Q., Cheng, M., and Wei, Y. (2021). LayerCAM: Exploring Hierarchical Class Activation Maps for Localization. *IEEE Trans. Image Process.*, 30:5875–5888.

Kertész, C. (2021). Automated cleanup of the imagenet dataset by model consensus, explainability and confident learning. *CoRR*, abs/2103.16324.

Klette, R. (2014). *Concise Computer Vision - An Introduction into Theory and Algorithms*. Springer.

Krishna, M., Neelima, M., Mane, H., and Matcha, V. (2018). Image classification using deep learning. *International Journal of Engineering & Technology*, 7:614.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C., Bottou, L., and Weinberger, K., editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 9992–10002. IEEE.

Liu, Z., Mao, H., Wu, C., Feichtenhofer, C., Darrell, T., and Xie, S. (2022). A convnet for the 2020s. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 11966–11976. IEEE.

Mahendran, A. and Vedaldi, A. (2015). Understanding deep image representations by inverting them. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 5188–5196. IEEE Computer Society.

Muhammad, M. B. and Yeasin, M. (2020). Eigen-CAM: Class activation map using principal components. In *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020*, pages 1–7. IEEE.

Olah, C., Mordvintsev, A., and Schubert, L. (2017). Feature visualization. *Distill*. https://distill.pub/2017/feature-visualization.

Palechor, A., Bhoumik, A., and Günther, M. (2023). Large-scale open-set classification protocols for imagenet. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2023, Waikoloa, HI, USA, January 2-7, 2023*, pages 42–51. IEEE.

Petsiuk, V., Das, A., and Saenko, K. (2018). RISE: randomized input sampling for explanation of black-box models. In *British Machine Vision Conference 2018, BMVC 2018, Newcastle, UK, September 3-6, 2018*, page 151. BMVA Press.

Poppi, S., Cornia, M., Baraldi, L., and Cucchiara, R. (2021). Revisiting the evaluation of class activation mapping for explainability: A novel metric and experimental analysis. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2021, virtual, June 19-25, 2021*, pages 2299–2304. Computer Vision Foundation / IEEE.

Radosavovic, I., Kosaraju, R. P., Girshick, R. B., He, K., and Dollár, P. (2020). Designing network design spaces. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10425–10433. Computer Vision Foundation / IEEE.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the Demonstrations Session, NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 97–101. The Association for Computational Linguistics.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408.

Rozsa, A., Günther, M., and Boult, T. E. (2017). LOTS about attacking deep features. In *2017 IEEE International Joint Conference on Biometrics, IJCB 2017, Denver, CO, USA, October 1-4, 2017*, pages 168–176. IEEE.

Rozsa, A., Rudd, E. M., and Boult, T. E. (2016). Adversarial diversity and hard positive generation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2016, Las Vegas, NV, USA, June 26 - July 1, 2016*, pages 410–417. IEEE Computer Society.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. S., Berg, A. C., and Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.*, 115(3):211–252.

Sandler, M., Howard, A. G., Zhu, M., Zhmoginov, A., and Chen, L. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4510–4520. Computer Vision Foundation / IEEE Computer Society.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. (2017). Grad-cam: Visual explanations from deep networks via gradient-based localization. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 618–626. IEEE Computer Society.

Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). Deep inside convolutional networks: Visualising image classification models and saliency maps. In Bengio, Y. and LeCun, Y., editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Workshop Track Proceedings*.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Smilkov, D., Thorat, N., Kim, B., Viégas, F. B., and Wattenberg, M. (2017). Smoothgrad: removing noise by adding noise. *ICML Workshop on Visualization for Deep Learning, 2017*.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. A. (2015). Striving for simplicity: The all convolutional net. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*.

Srinivas, S. and Fleuret, F. (2019). Full-gradient representation for neural network visualization. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 4126–4135.

Su, J., Vargas, D. V., and Sakurai, K. (2019). One pixel attack for fooling deep neural networks. *IEEE Trans. Evol. Comput.*, 23(5):828–841.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1–9. IEEE Computer Society.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., and Fergus, R. (2014). Intriguing properties of neural networks. In Bengio, Y. and LeCun, Y., editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Tan, M. and Le, Q. V. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR.

Tan, M. and Le, Q. V. (2021). Efficientnetv2: Smaller models and faster training. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 10096–10106. PMLR.

Tu, Z., Talebi, H., Zhang, H., Yang, F., Milanfar, P., Bovik, A. C., and Li, Y. (2022). Maxvit: Multi-axis vision transformer. In Avidan, S., Brostow, G. J., Cissé, M., Farinella, G. M., and Hassner, T., editors, *Computer Vision - ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXIV*, volume 13684 of *Lecture Notes in Computer Science*, pages 459–479. Springer.

Wang, H., Wang, Z., Du, M., Yang, F., Zhang, Z., Ding, S., Mardziel, P., and Hu, X. (2020). Score-CAM: Score-weighted visual explanations for convolutional neural networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR Workshops 2020, Seattle, WA, USA, June 14-19, 2020*, pages 111–119. Computer Vision Foundation / IEEE.

Yosinski, J., Clune, J., Nguyen, A. M., Fuchs, T. J., and Lipson, H. (2015). Understanding neural networks through deep visualization. *ICML Deep Learning Workshop, 2015*.

Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In Fleet, D. J., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*, volume 8689 of *Lecture Notes in Computer Science*, pages 818–833. Springer.

Zhang, J., Bargal, S. A., Lin, Z., Brandt, J., Shen, X., and Sclaroff, S. (2018). Top-down neural attention by excitation backprop. *Int. J. Comput. Vis.*, 126(10):1084–1102.

Zhou, B., Khosla, A., Lapedriza, À., Oliva, A., and Torralba, A. (2016). Learning deep features for discriminative localization. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2921–2929. IEEE Computer Society.

Zintgraf, L. M., Cohen, T. S., Adel, T., and Welling, M. (2017). Visualizing deep neural network decisions: Prediction difference analysis. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.