JOHANNES KEPLER
UNIVERSITY LINZ

Submitted by
Viktor Maximilian Loreth
k12006268

Submitted at
Institute of Computational Perception

Supervisor
Katharina Hoedt, PHD

October 20, 2023

EVALUATION OF INTERPRETABILITY METHODS IN IMAGE RECOGNITION

Bachelor Thesis

to obtain the academic degree of

Bachelor of Science

in the Bachelor's Program

Artificial Intelligence

JOHANNES KEPLER UNIVERSITY LINZ
Altenbergerstraße 69
4040 Linz, Österreich
www.jku.at
DVR 0093696

JOHANNES KEPLER
UNIVERSITY LINZ

Sworn Declaration

I hereby declare under oath that the submitted Bachelor Thesis has been written solely by me without any third-party assistance, information other than provided sources or aids have not been used and those used have been fully documented. Sources for literal, paraphrased and cited quotes have been accurately credited.

The submitted document here present is identical to the electronically submitted text document.

Linz, October 20, 2023

JOHANNES KEPLER
UNIVERSITY LINZ

Abstract

This thesis focuses on the evaluation of interpretability methods. The primary challenge of this research area is the absence of a benchmark for evaluating interpretability methods. Although intrinsic interpretative models exist, the rise of black-box networks and their development present substantial challenges to interpretability. Despite various interpretability methods existing, the issue of evaluation remains unresolved. Existing evaluation methods, including "RemOve And Retrain" (ROAR), "Keep And Retrain" (KAR), "Benchmarking Attribution Methods" (BAM) and "Real-Time Image Saliency for Black Box Classifiers," offer numerical evaluations. Nevertheless, they remain ambiguous. In this thesis, we provide an overview of the current state of interpretability methods and their evaluation. Furthermore, we look at reconstructing the ROAR method using two attribution approaches.

JOHANNES KEPLER
UNIVERSITY LINZ

Contents

1 Introduction **1**
1.1 Structure of the Thesis . . . 2

2 Machine Learning and its Interpretability **3**
2.1 Supervised Machine Learning . . . 4
2.1.1 Linear Models . . . 4
2.1.2 Distance-based Methods . . . 5
2.1.3 Support Vector Machines . . . 6
2.1.4 Decision Trees . . . 6
2.1.5 Neural Networks . . . 8

3 Interpretability of Supervised Machine Learning Algorithms in Image Recognition **10**
3.1 Model-Agnostic Methods and their applicability to Neural Networks . . . 12
3.1.1 Global Model-Agnostic Methods . . . 12
3.1.2 Local Model-Agnostic Methods . . . 13
3.2 Model-Specific methods for Neural Networks . . . 13
3.2.1 Feature Visualization . . . 14
3.2.2 Network Dissection . . . 16
3.3 Attribution Maps . . . 16
3.3.1 Visualising Image Classification Models and & Deconv-Net . . . 17
3.3.2 Gradient-weighted Class Activation Map (Grad-CAM) . . . 17
3.3.3 Guided Grad-CAM . . . 17
3.3.4 Integrated Gradient . . . 18
3.3.5 Ensemble Methods: Smooth Gradient, Smooth Grad² and VarGrad . . . 18
3.3.6 Summarizing Attribution Maps . . . 19

4 Evaluation of post-hoc Interpretability Methods **20**
4.1 Completeness and Soundness . . . 21
4.2 Evaluating the Visualization of What a Deep Neural Network has learned . . . 21
4.3 A Benchmark for Interpretability Methods in Deep Neural Networks . . . 22
4.4 Benchmarking Attribution Methods (BAM) . . . 22
4.5 Sanity Checks for Saliency Maps . . . 23

5 Concluding Thoughts on Attribution and Evaluation Methods **25**

6 Reconstructing ROAR **26**
6.1 Scientific Motivation and Goal . . . 26
6.2 Project Setup MNIST . . . 26
6.3 Project Setup Food-101 . . . 28
6.4 Summary and Interpretation . . . 30

Appendix **35**
1 MNIST-Computation 35
2 Food101-Computation 35

List of Figures

2.1 Decision Tree Example 7
2.2 Neural Network 8

3.1 Interpretability categorization 11
3.2 Network Dissection: Feature Visualization 15
3.3 Activation Maximization 15
3.4 Attribution Map 16

4.1 Cascading Randomization on Image Net 24

6.1 Integrated Gradients and Random Baseline Comparison 27
6.2 Accuracy: Random Baseline vs Integrated Gradient 28
6.3 Comparison of the accuracy behaviour of the original model vs the modified model. 29

1 Introduction

Artificial Intelligence (AI) has undergone rapid development in the last few years. In today's modern era of mobile phones and computers, algorithms are used on a daily basis to have quick access to information and improve the efficiency of daily life.

While various algorithms (e.g.: Decision Trees, Linear Regression, Support Vector Machines, etc.), which are comprehensible by design, have been developed, the spotlight has turned to Deep Neural Networks (DNNs). This shift is attributed to the increase in computational power and the exponential increase in accessible data. Despite their remarkable accuracy, Deep Neural Networks remain opaque black boxes, which we struggle to understand [38]. Nevertheless, the immense improvement in performance and their ability to handle massive datasets have led to widespread adoption in contemporary devices [48]. It is predicted, that algorithms based on Neural Networks will be becoming increasingly popular in the next years.

However, one of the primary difficulties with Neural Networks is the lack of reliable interpretability techniques. Understanding of the models is needed for various reasons: Regulatory laws require understandability of the data. The deficiency of interpretability in machine learning models leads to a presence of biases, such as gender discrimination and racial disparities. In the absence of a thorough understanding of a model's working, it becomes exceedingly difficult to confirm the functionality. This leads to users not trusting and avoiding machine learning systems. Beyond those user-centric advantages, it also supports the model development process and allows experts to extract valuable insights [38].

Numerous interpretability methods have been developed, yet a universally reliable method remains missing. Particularly in the domain of image analysis, encompassing critical applications like automated driving and facial recognition, no solution is present. The decision-making rationale of neural networks remains unclear, attributed to factors like background elements, peripheral objects or lighting conditions. Efforts to address this issue have given rise to several gradient methods, aiming to assign significance values to pixels and represent their importance on neural network decisions [38].

Another alternative option to mitigate the black-box nature of algorithms involves employing model-agnostic methods. These methods offer a computational linkage between features and labels, irrespective of which model is used. Although highly effective for smaller datasets, they begin to

struggle as the data size and complexity increase. Because of this, they do not offer a reliable method to quickly make Neural Networks interpretable [28].

In light of these prevalent problems, the object of this thesis is to recapitulate interpretability methods for neural networks in computer vision. Emphasis is placed on the evaluation of post-hoc interpretability techniques, forecasting potential future developments and focusing on the strengths and weaknesses of distinct techniques. Concluding the theoretical segment, a practical demonstration showcasing the application of RemOve And Retrain (ROAR)[20] is shown.

1.1 Structure of the Thesis

1. In the first chapter 2 "Machine Learning and their Interpretability", an overview of contemporary machine learning algorithms is made, categorizing them into two main groups: algorithms with inherent interpretability and those without. The goal is to make clear how supervised methods can be applied to image recognition tasks.
2. In chapter 3 "Interpretability of Supervised Machine Learning Algorithms in Image Recognition" we dig into interpretability, emphasizing global and local model-agnostic techniques. These methods offer insights into overall model behaviour, regardless of algorithm specifics.
3. In subsequent sections 3.3 "Neural Network specific Interpretability Methods", model-specific post-hoc methods for Neural Networks are introduced. Feature visualization and gradient-based methods are explained.
4. In chapter 4 "Evaluation of post-hoc Interpretability methods", we focus on evaluating post-hoc interpretability methods. Various approaches to assess the effectiveness and dependability of these methods in offering meaningful insights into intricate models are introduced and discussed. Additionally, the advantages and disadvantages of these approaches are carefully examined to provide a comprehensive understanding of their applicability.
5. In chapter 5 "Which evaluation method and attribution method to use?", a conclusion is presented and an advice on which evaluation method to use is given.
6. In the last chapter 6 "Reconstructing ROAR", the practical application of the ROAR methodology using the Food-101 dataset [8] and MNIST dataset [12] is presented to exemplify the discussed concepts. This real-world instance illustrates the current state of art of evaluation techniques in image recognition.

2 Machine Learning and its Interpretability

In the rapidly evolving landscape of machine learning, interpretability has emerged as an important concept. Before going in-depth into various algorithms and methods, the fundamental question of: "What is interpretable Machine Learning (IML) and why do we need it?" is answered. A broad definition of IML given by [3]: "Interpretable machine learning is the use of machine learning techniques to generate human-understandable insights into data, the learned model, or the model output." This definition underscores the important role of understanding complex models and allowing humans to understand the computation.
Interpretability is important for various reasons, ranging from model validation and debugging to fostering validation and trust. Allen[3] summarizes a number of objectives for interpretable machine learning:

Model Validation: Interpretable models are essential for validating (by a human) whether a learned model behaves as expected and consistently aligns with prior expectations and knowledge about the system.

Model Debugging: When unexpected behaviour occurs, finding the reasons for fault is impossible without understanding the system. Interpreting and understanding machine learning systems is critical for diagnosing, debugging and fixing systems [22].

Transparency, Accountability & Trust: IML transforms black-box machine learning systems into understandable systems. The utilization of high-stakes societal applications requires accountability and trust in machine learning systems [13, 37, 38, 44].

Ethics: Machine learning algorithms can be trained on biased data leading to unfair predictions that are discriminatory. To improve the fairness of machine learning algorithms intepretable methods need to be deployed [17].

Data Exploration and Discovery: Insights into major patterns, trends, groups, or artefacts of the data are achieved by applying human-interpretable techniques. These data exploration insights influence the data pre-processing and model decisions [5, 6, 29].

Before going into detail about the different methods of interpretability an overview of current supervised machine learning is given. Unsupervised machine learning is arguably intrinsically understandable, as the goal is to find a structure in the data. [3] With those examples, the necessity of interpretability for neural networks should be made evident.

2.1 Supervised Machine Learning

In supervised machine learning, there are a range of foundational classification methods. They are briefly introduced and analyzed for their interpretability. This section analyzes the base functionality of each algorithm. Furthermore, an analysis of the interpretability from a human perspective is made.
The interpretability of algorithms depends on two factors, [28]. When determining the transparency of an algorithm, we evaluate the human interpretability of how the model learns from the underlying structure of the data. Is it possible for a human to understand the implications of the mathematical operations? The second factor is the interpretability of the learned parameters. Understanding the factors of a linear regression model is easy. Grasping the millions of weights in a neural network is impossible.

In this section, we present a selection of frequently employed supervised machine learning techniques. As will become clear in the following subsections, interpretability was predominantly achieved through the methods' inherent interpretability.

2.1.1 Linear Models

When predicting outcomes, one of the simplest methods is to use a linear regression model. This model predicts by adding up n features (x_n) multiplied by an respective individual weight (α_n). The predictive output $\hat{y}_i$ is calculated:

$$\hat{y}_i = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + ... + \alpha_n x_n + \epsilon$$

The alphas α_i indicate the significance of each feature. The initial coefficient α_0 is known as the intercept, signifying the baseline. The noise ϵ describes the inevitable errors from inherent nonlinearity in real-world dynamics or measurement inaccuracies.
To train the model, the MSE-Loss (Mean Squared Error) or the ABS-Loss (Absolute Loss) is applied between the true label y_i and the predicted label $\hat{y}_i$. The goal is to minimize this loss function.

$$\text{MSE-Loss} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$\text{ABS-Loss} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

The interpretation of the model is simple. The factors are described through the coefficient matrix. Each feature is distinctive to the model and the weighting is visible in the factors α_i (assuming normalization).

$$\alpha = \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ ... \\ \alpha_i \end{bmatrix}$$

Although linear models possess comprehensibility, provide a straightforward method for prediction and are inherently understandable, their application is limited to linear relationships and small datasets. In the domain of image recognition, it is not applicable because the features are not linearly correlated.

2.1.2 Distance-based Methods

K-Nearest Neighbors serves as a classification method by considering the nearest neighbours.

Assume you have a dataset $D = (x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ where x_i represents the feature vectors, and y_i represents the corresponding class labels. Each x_i is a point in a multidimensional feature space, and y_i belongs to one of the classes (e.g., $y_i \in 0, 1$) for binary classification). Given a new data point x_{new} that you want to classify, K-NN works as follows:

1. Calculate the distance between x_{new} and all other data points in the training dataset.
2. Select the K data points (nearest neighbors) with the smallest distances to x_{new}.
3. Count the number of data points in each class among the selected KK neighbors.
4. Assign x_{new} to the class that has the majority of neighbors.

Although KNN's parameters are not interpretable by default, the underlying concept is straightforward: A sample has the same class as samples with similar features. This makes KNN's inherently interpretable and makes it a common choice when interpretability is needed. However, KNN's encounter difficulties when dealing with many features and larger datasets, therefore using it for image recognition is not recommended.

2.1.3 Support Vector Machines

Support Vector Machines (SVMs) [7] are used to find a hyperplane that best separates different classes y_i of data points. The primary objective is to maximise the distance between the hyperplane, defined by the weight vector $\vec{w}$, and the closest data points x_i from each class. This is done to ensure that the datasets are separated as much as possible. Because there may be classification errors and noise in the dataset, it is possible to set an initial regularization parameter C to handle outliers.

The optimization problem SVM in mathematical notion:

$$\text{Minimize } \frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$

$$\text{Subject to } y_i(\vec{w}\cdot x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \text{ for } i = 1, \ldots, n$$

$\sum_{i=1}^{n}\xi_i$ represents the sum of slack variables (ξ_i), which measure how much a data point violates the margin constraint. While maximising the margin, Support Vector Machines (SVM) seeks to reduce this sum, minimizing the violations.
b is the bias term, which shifts the hyperplane away from the origin.

SVMs can handle non-linear data through the kernel trick, in which a kernel is applied to the data, transforming it to a different space before computing the hyperplane. However, as the dimensionality increases, SVMs become non-interpretable, making it challenging to visualize and understand the weight matrix. Moreover, SVMs struggle with larger datasets and high-dimensional feature spaces, making them less effective for image classification.

2.1.4 Decision Trees

Decision trees are a commonly used machine learning algorithm that proves effective in both classification and regression tasks. The reason for their popularity can be attributed to their user-friendly structure, which has the capacity to handle both categorical and continuous data. A decision tree is made up of three main types of nodes: root, internal, and leaf nodes, which combine to form a representation often depicted as a tree. In figure 1, a decision tree is plotted, where the distribution is shown in the 'value' field, and the classification of the node/leaf is presented in the 'class' field.

The classes are separated in each root node by a decision criterion: F_i <= criterion. The Gini impurity is commonly used for calculating the splitting criterion, which aims to best separate the data in the leaf. New nodes and are defined until the dataset is perfectly separated. However, this

Dataset

Feature 1	Feature 2	Class
2.5	3.0	0
1.0	2.0	0
4.0	2.5	1
3.5	4.0	1
5.0	3.5	0
4.5	1.5	1

Decision Tree

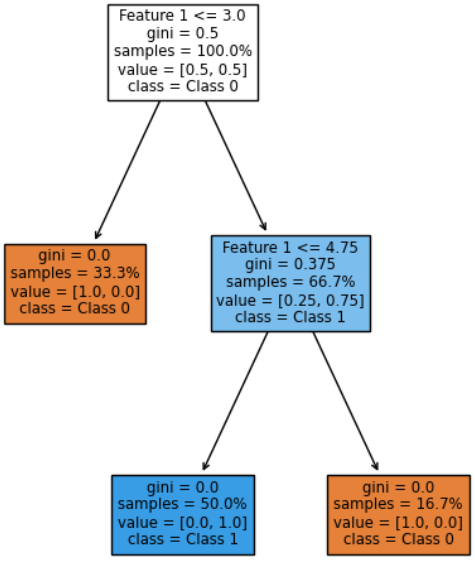

Figure 2.1: Decision Tree Example

makes decision trees prone to overfitting as the model becomes overly complex and captures noise or random fluctuations in the training data, rather than the underlying patterns or relationships. Consequently, the resulting tree fits the training data exceptionally, but does not generalize well to new or unseen data.

The depth of the tree can be restricted to avoid overfitting. Then, the majority is utilised to assign a label to the leaf. The depth of the tree can be restricted to avoid overfitting. Nevertheless, as the dimensionality grows, determining the depth becomes more and more challenging.

Random forests, comprise many decision trees and are known to prevent overfitting. However, they tend to be more challenging to interpret because of the use of multiple algorithms. Techniques like as SHAP values [26] or partial dependence plots [16] can be employed to make them more understandable.
Gradient boosting like XGBoost [10], LightGBM [21] and CatBoost [34] share similarities with random forests and decision trees, but they assign differential learned weights to each decision. They suffer from the same interpretability issues as random forests. While decision tree-based methods can be applied for image classification, their accuracy tends to be worse than in neural networks.

2.1.5 Neural Networks

The rise of neural networks and their powerful predictive capabilities has made them a popular choice for classification tasks. However, as these networks become increasingly complex, the traditional approach of understanding them through weight examination becomes challenging. With the emergence of Convolutional Neural Networks (CNN) in 2012 [23], neural networks have become the state-of-the-art for image prediction.

n a neural network, each neuron calculates a weighted sum of its inputs and applies an activation function to produce an output. This output serves as the input for the subsequent layer. While various neuron types exist, including convolutional and recurrent neurons, foundational neurons are present in nearly all neural network architectures. A visual representation of a neural network can be seen in Figure 2.2.

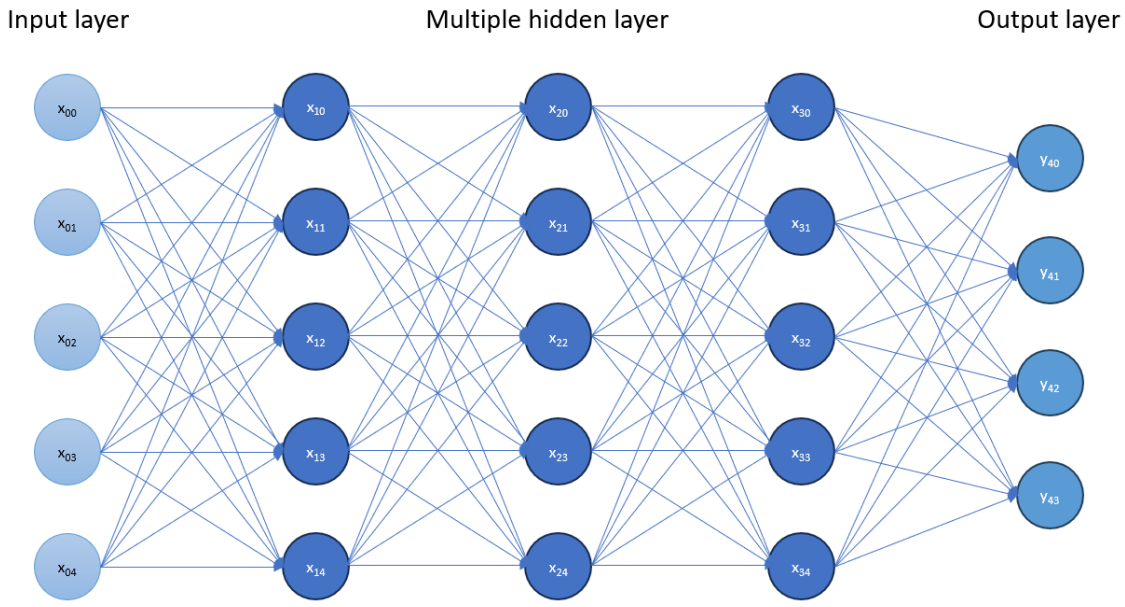

Figure 2.2: Neural Network

In this neural network, each circle, except those in the input layer, represents a weight matrix denoted as $w_{j,i}$. Here, 'j' signifies the hidden layer number, and 'i' represents the weight number within that layer. The output of a neuron is determined by summing the products of the inputs x_{ji}, their corresponding weights, and a bias term b_{ji}, expressed as $y_{(j+1)i} = \sum_{i=1}^{n} w_{ji} * x_{ji} + b_{ji}$. If the neuron is not in the output layer, an activation function, such as Sigmoid or ReLU, is applied to y_{ji}, transforming it into the new input for the next layer $x_{ji} = f(y_{ji})$.

Labeled data is used to train a neural network. The prediction error is minimised by calculating the prediction error and the gradients. This is typically done by backpropagation over several epochs until the network can accurately predict new data.

Modern neural networks, such as ResNet50 [19] comprise over 50 layers and utilize more than thousand kernels. Although they produce the greatest results, they lack an intuitive explanation of the learned parameters, unlike other types of supervised machine learning algorithms. Nonetheless, interpreting them to some degree is a necessity. Therefore, methods were developed which attempt to make any supervised machine learning algorithm interpretable.

3 Interpretability of Supervised Machine Learning Algorithms in Image Recognition

Before going into detail of the chapter, the primary challenge of image recognition is explained: Consider an image with dimensions of 3x224x224. The key objective in achieving model interpretability is to discern which elements of the image are responsible for influencing the model's output. The task is to highlight or establishing a clear link between specific regions within the image and the output. The process of creating a correlation between image features and classification becomes exceedingly complex and non-linear for neural networks. This complexity explains why many interpretability methods often prove inadequate for image recognition. For instance, when an object within the image transitions from one quadrant to another, different areas of the image become responsible for the classification. Furthermore, challenges such as image blurriness, mirroring or the presence of multiple potential classifications make interpretability difficult. To address these issues, special model specific interpretability methods have been developed.

Before going into details of interpretability methods for this problem, a framework 3.1 is presented [3] for classifying supervised machine learning methods and interpretability methods.

Intrinsic vs Post-hoc: Interpretability methods can be broadly classified into intrinsic and post-hoc techniques. Intrinsic methods are inherently understandable by design, while post-hoc interpretations involve analyzing the model's behaviour after its creation. Typical intrinsic understandable methods are Decision Trees 2.1.4 and KNNs 2.1.2. In this chapter only post-hoc methods are described.
Global Interpretations vs Local Interpretations: Global Interpretations encompass the entirety of a fitted model. In contrast, local interpretations zoom in on specific portions of the model landscape, such as class boundaries.
Model-Specific Interpretations vs Model-Agnostic Interpretations: Model-specific interpretations are tailored to a particular class of algorithms, like gradient methods for neural networks. In contrast, model-agnostic interpretations, such as LIME or SHAP, can be applied across any classification algorithm.

The goal of this section is to present several commonly used interpretability methods and evaluate their suitability for image classifications. As Neural Networks are the best performing methods

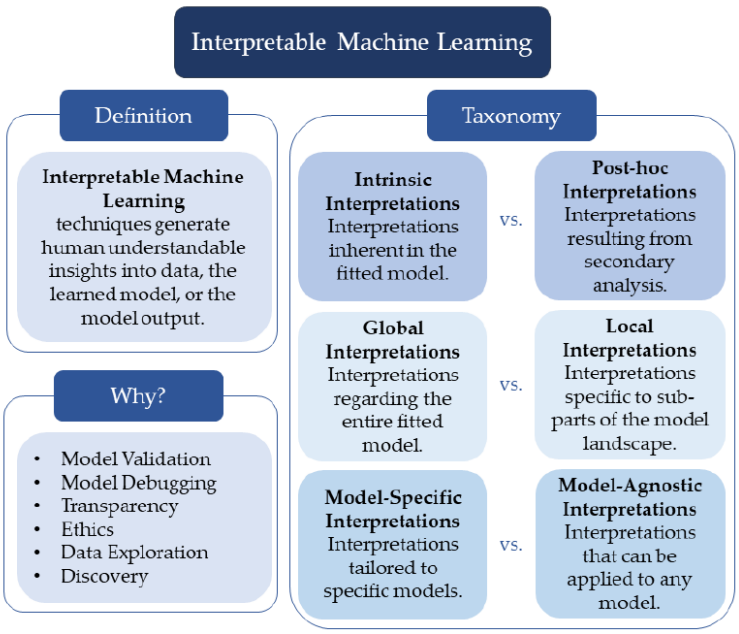

Figure 3.1: Interpretability categorization (Figure from [3])

for image recognition, the focus lies on the suitability of the interpretability methods for Neural networks. After going over model-agnostic interpretability methods in the next sections, we go in-depth about model-specific interpretations for neural networks in 3.2.

3.1 Model-Agnostic Methods and their applicability to Neural Networks

Model-agnostic methods can be applied to any method. However, it does not make sense to use the same method for every algorithm. In this section, global and local agnostic methods are summarised and their applicability to neural networks in image classification is discussed.

3.1.1 Global Model-Agnostic Methods

The aim of global methods is to provide a complete overview of the functioning of a model. However, it is very difficult to represent the non-linearity in image classification for multiple classes. While some methods work reasonably well for the computation required, there is no global agnostic method that fully covers the need for interpretability.

1. **Partial Dependency Plots:** Partial Dependency Plots (PDPs) are a straightforward method to display the relationship between individual or multiple features and an outcome. However, when dealing with a bigger feature size, there are too many plots to make sense out of the data. Therefore they are not feasible for Neural Networks and image recognition tasks [16].
2. **Accumulated Local Effects:** Accumulated Local Effects (ALE) are an advancement of PDP. While it overcomes certain difficulties from PDPs, they struggle with the same problem when it comes to a bigger feature size.
3. **Feature Interaction:** Feature Interaction analyzes the interaction between features inside the model. The underlying Friedman's H-statistic is a method to evaluate the correlation and variance. Because of the complexity of image tasks and the high computational costs, this method is not applicable to Neural Networks in a meaningful matter.
4. **Functional Decomposition:** Functional Decomposition is used in Neural Networks. In Chapter 3.2.1 a method to disassemble networks is presented.
5. **Permutation Feature Importance:** Permutation Feature Importance is regularly used in visual machine learning tasks. In section 4.2 an example of a perturbation method is shown.
6. **Prototype and Criticism:** The creation of clusters using the prototype and criticism method involves categorising well presented data instances as 'prototypes' and sparsely presented ones as 'criticism'. This approach aims to improves the interpretability of these clusters by verifying their classification accuracy. However, a significant challenge with this method arises when applying it to images, as it can be difficult to generate reliable clusters that effectively group the data.

3.1.2 Local Model-Agnostic Methods

The aim of local methods is to reveal detailed knowledge of how a model behaves, often tailored to individual cases. In the field of image classification, where non-linearity and multiple classes contribute added complexity, achieving accurate interpretability for a single class is already challenging. Although some techniques effectively manage the trade-off between computational efficiency and interpretability in certain scenarios, there is currently no universally accepted model-agnostic method for achieving this.

1. **LIME: Local Interpretable Model-agnostic Explanations:** Lime generates explanations in a local scope by training interpretable models on the predictions of a model. LIME only covers a single local context of the model. LIME can be applied to various types of data, including image data.
2. **Scoped Rules (Anchors):** Anchors are distinctive patterns or conditions which guarantee a prediction. Finding anchors becomes increasingly expensive with more features present and is not normally used in neural networks. [35]
3. **Individual conditional expectation curves:** ICE's display one line per data sample and display how the sample changes when a feature changes. It is an individual version of PDP. [15] It is not used in image classification or to evaluate neural networks.
4. **Counterfactual explanations:** Counterfactual explanations of a prediction explain the smallest change of feature values which is necessary to change the prediction of an output. The problem in using this method is that for each instance multiple counterfactual explanations exist. The use of counterfactual explanations in image recognition as a standalone method is not common.
5. **SHAP (SHapley Additive exPlanations):** SHAP values [26] calculate how much each feature contributes to the difference between a models prediction for a specific instance and the average prediction across all instances. Positive SHAP values indicate features that increase the predicted likelihood for a particular class, while negative values indicate features that decrease it. It is used commonly in neural networks.

3.2 Model-Specific methods for Neural Networks

In the domain of Natural Language Processing (NLP) and Computer Vision, Deep Learning has proven very successful. By passing the features through a sequence of layers, characterized by matrix multiplications with kernel weights and nonlinear transformation functions, a prediction

is computed. Depending on the specific task, additional elements like Long Short-Time Memory(LSTM) layers and Convolutional layers are utilized. Given the immense amount of mathematical operations and the non-linearity underlying a single prediction, humans are not fit to apprehend the mapping. To interpret predictions, we would have to decipher the intricate learned knowledge of numerous different kernels and weights. Recognizing that humans cannot grasp millions of weights, the demand for interpretability methods is high. To assess the behaviour and predictions of Deep Neural networks, specific interpretability methods were developed. These methods calculate the likelihood of a feature being responsible for the result.

While model-agnostic methods offer an approach to understanding Neural Networks, the sheer size of the data used to train and test Neural Networks makes this task extremely hard. For instance, in an image with the dimensions of 3x224x224, as commonly encountered in Food-101, the data features exceed 150.000. In NLP tasks, where vocabularies often encompass around 20.000 words, the computational complexity renders most model-agnostic techniques as too expensive.
In the pursuit of comprehending the complexity of Deep Neural Networks, it makes sense to utilize the weights in the model. The information saved in the hidden layers as learned weights can be used to evaluate the network. Moreover, the gradients can be taken into consideration as well. In the following subsections, several concepts for understanding Deep Neural Networks are introduced.

3.2.1 Feature Visualization

Feature Visualization tries to make parameters in singular layers in neural networks understandable. The goal is to give understanding of a deep neural network by dissecting each layer and understanding the underlying parameters through visualization.

The higher-level features in these networks relate to clear concepts, shown in Figure 3.2. As the features (image-pixels) pass through the layers, the feature changes at each layer. In each convolutional layer, the network gains new and more complex features. The smooth joining of fully connected layers then converts image-based data into predictions.

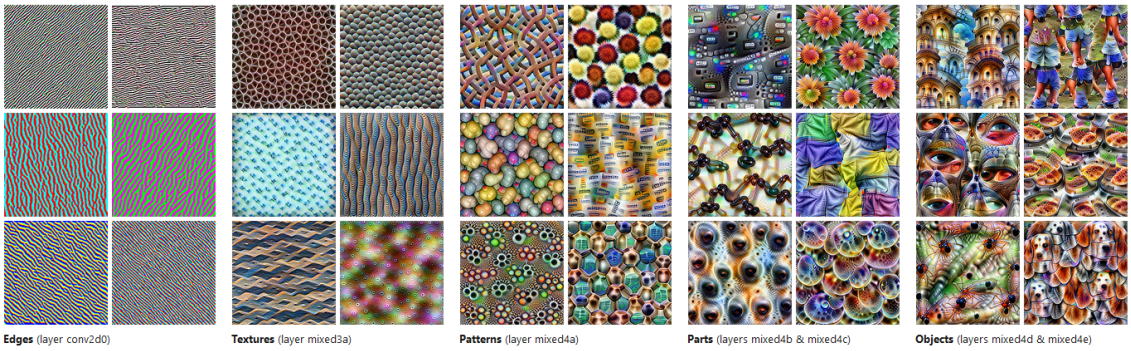

Figure 3.2: Network Dissection: Feature Visualization (Figure from [31])

The figure explains this process. The first convolutional layers find simple features like edges and basic textures. Later, they recognize more detailed patterns. The deepest layers learn about parts and objects. This object information passes to the other hidden layers, which then finally make a prediction.

Feature visualization is based on activating one kernel in the network. This involves maximizing the activation of a specific neuron (Visible in figure 3.3). There are two methods for achieving this. First, we can make use of the training image that triggers the highest activation. Yet, this approach faces a significant problem. When an image contains multiple objects, it is hard to pinpoint which object causes the activation. Because of this, an alternative route is adopted: generating new images from random noise. This is accomplished through methods like Generative Adversarial Networks (GANs) or other diffusion-based techniques.

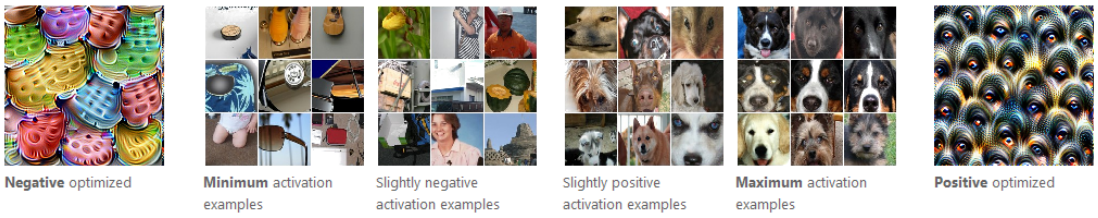

Figure 3.3: Activation Maximization (Figure from [31])

Feature visualisation provides a first insight into the behaviour of a model, improving the understanding of its inner layers. It also has the potential to enrich domain understanding by aligning learned features with domain-specific knowledge. Furthermore, it can assist in debugging and refining models, contributing to their overall performance improvement. However, interpretation of the visualised features is difficult and decision making is challenging to understand.

3.2.2 Network Dissection

Network dissection, a technique introduced by Olah in 2018[32], builds on the principles of feature visualization. This method establishes a link between individual kernels and the prediction of specific features by exploiting the concept of visualizing the importance of channels and kernels, as introduced in the section before.

Although this method is effective in comprehending low-level features, comprehending high-level features still remains a challenge. Furthermore, analyzing a single sample is very time consuming. However, this interpretability technique holds significant potential for enhancing the transparency of neural networks.

3.3 Attribution Maps

Attribution maps are visualizations that highlight the regions of an input image that have the most significant impact on a models output. By revealing the areas that strongly influence a prediction, saliency maps bridge the gap between the models "black-box" nature and human understanding. An example attribution map is visible in figure 3.4.

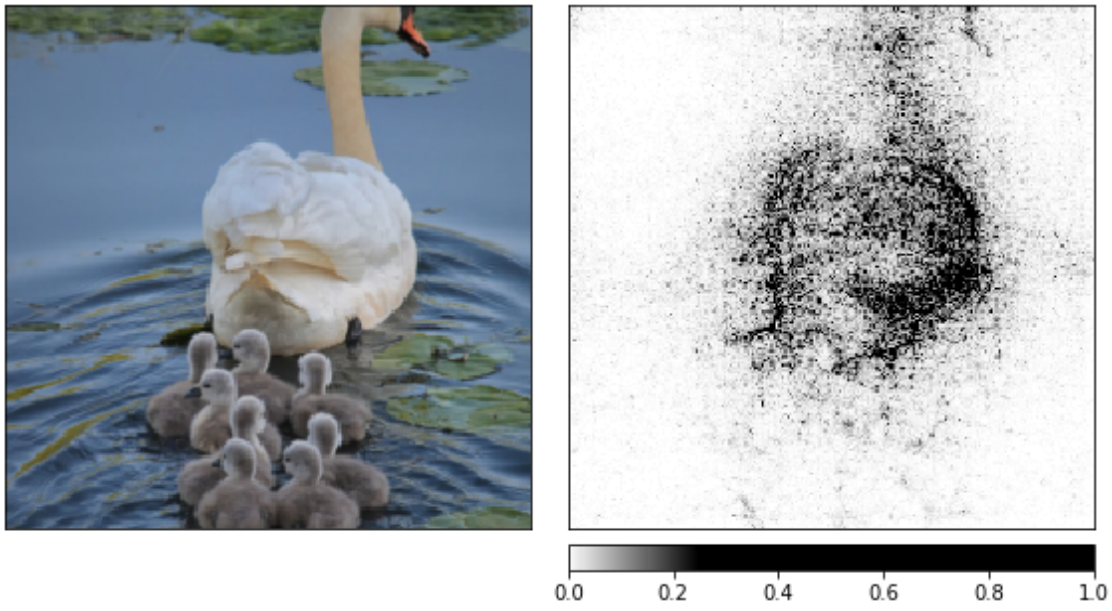

Figure 3.4: Attribution Map (Figure from [1])

Attribution maps are typically calculated using SHAP[26] or gradient methods. Attribution maps provide a direct and intuitive way to understand which parts of input data influence a model's decision. The main difficulty is in generating reliable attribution maps. The remaining subsections

introduce some commonly used methods. The key objective for all following methods is the same: Generating a reliable and easily understood attribution map.

3.3.1 Visualising Image Classification Models and & Deconv-Net

Visualising image classification models (Gradient) [40] focuses on computing gradients within neural networks. It involves generating a forward pass of an image and then computing gradients for the class scores. These gradients are visualized, akin to the backpropagation process. However, this method faces two challenges. First, when Rectified Linear Units (ReLU) are used, negative gradients are seemed as unimportant and are set to zero. This leads to an information loss in the gradients. Second, in pooling layers, gradients are absent, resulting in further information loss.
The Devonc-Net from Zeiler [46] addresses these problems: A Deconv-Net-layer [47] is attached to each convolutional layer providing a path back to the image pixels. The Devonc-Net practically learns weights to reverse the process from input to the last convolutional layer. Unpooling in Deconv-Net addresses the non-invertibility of pooling operations by preserving the original maxima locations through switch variables, which are subsequently employed to reconstruct the activations. To maintain positive signals during reconstruction, a reverse ReLU non-linearity is applied in each layer, similar to the forward pass. Additionally, Deconv-Net utilizes transposed filters, effectively mirroring each filter both vertically and horizontally to achieve filtering in the reverse direction.

This results in making it possible for visualizing the responsible input pixels for a classification output.

3.3.2 Gradient-weighted Class Activation Map (Grad-CAM)

Grad-CAM [42] ignores low-relevancy classes, which implies that percentages for predictions below a certain threshold are not taken into account. The method then backpropagates gradients specifically for the primary class of interest. These gradients are rescaled to fall within the [0,1] range and visualized. Grad-CAM takes into account all layers except for the convolutional layers. Evaluating the importance of each kernel before the convolutional layers offer insights into the significance of each input feature. This also explains the coarse accuracy, as pooling layers and convolutional layers are not handled and the resolution is up-scaled to the required format.

3.3.3 Guided Grad-CAM

Guided Grad-CAM combines the backpropagation [42] with another method to have a better localization. The up-sampled attribution map from Grad-Cam is multiplied with another attribution

method. This acts as a lenses for other attribution methods to focus on specific parts of the attribution map.

3.3.4 Integrated Gradient

Integrated Gradient [43] computes the integral of the gradients of the model's prediction with respect to its input variables. By integrating over this path, it quantifies how each feature contributes to the change in the model's prediction. A reference variable, such as a baseline is used to measure how each feature influences changes in each prediction. The method is commonly used, as it considers the entire input space and is applicable to any model.

3.3.5 Ensemble Methods: Smooth Gradient, Smooth Grad² and VarGrad

All the following methods can be applied to gradient-based techniques to adjust their behaviour and enhance their robustness in various applications.

SmoothGrad [41] is a smoothing approach to mitigates noise in gradient calculations. To achieve this, it generates a set of J noisy estimates by independently adding Gaussian noise θ to the input x. These noisy estimates are then averaged to obtain a more reliable and accurate gradient estimate, as represented by the formula. g is describing the calculation of an attribution map, A, is the attribution map:

$$A = \frac{1}{J}\sum_{i=1}^{J}(g(x+\theta))$$

SmoothGrad² [20] builds upon the concept of SmoothGrad by squaring all estimates before averaging them. This modification is aimed at emphasizing the contributions of gradients while further reducing the impact of noise. The formula is as follows:

$$A = \frac{1}{J}\sum_{i=1}^{J}(g(x+\theta)^2)$$

VarGrad [2] offers an alternative approach to aggregating gradient estimates. Instead of summing them up or averaging them, VarGrad focuses on estimating the variance of these estimates. This aggregation provides insights into the variability of gradient information across different perturbations, offering a unique perspective on the model's behaviour. The formula for VarGrad is expressed as:

$$A = Var(g(x+\theta))$$

3.3.6 Summarizing Attribution Maps

There exist many more attribution maps but there exists one major problem: There is no consensus which method should be used. While attribution maps have advantages [28], such that they are fast to compute and easy to interpret, they are also unreliable and can be insensitive to model and data. Furthermore, it is difficult to know whether an explanation is correct. In the next chapter, the evaluation of interpretability methods and metrics are discussed.

4 Evaluation of post-hoc Interpretability Methods

In this chapter the evaluation of attribution maps and the underlying gradient methods is discussed. Finding suitable evaluation methods for attribution maps is difficult. Still, various evaluation methods have been developed which try to assess the performance of saliency methods. While they do not offer an objective truth if one saliency methods outperforms another one across all tests, they can offer some guidance one the effectiveness of saliency methods. Gupta[18] proposes that those evaluation methods can be broadly categorized into two types:

Extrinsic Evaluation: The evaluation is compared against predefined ground truths explanations, established by human annotator or by another network, as demonstrated in Zhang[49]. However, the sheer amount of images makes the evaluation challenging. Furthermore, the quality of an attribution map is hard to judge. When is an evaluation correct? Also, Adebayo [2] underscores the issue of confirmation bias, where an explanation may be considered correct, because it highlights the same regions as humans might choose, but not ensuring that the model employs the same learned concepts. Furthermore, in extrinsic evaluation a significant challenge is the lack of ground truths in attribution maps. Research still faces the challenge that the development of such a model is still missing.

Intrinsic Evaluation: Intrinsic methods rely on computational analyses within the neural network itself, without requiring human judgments. These methods are based on creating a new composite input using the heat map and the original input. Then they are evaluated using either the same model[11] or new model is trained and evaluated[20]. While both these methods generate meaningful results, they both do not offer an objective evaluation method. Without re-training it is not clear if the degrade in performance stems from the distribution shift or the removal of informative features [20]. By re-training, a different model is evaluated.

Many interpretable AI-methods exist and there is a question: Can I compare the effectiviness between them? Can I combine the results and the one which seemingly has the most correlation is good? Neely, [30] has tested several explainable AI methods in the domain of NLP and can only find little correlation. Additionally, the correlation is model and task dependent, which furthermore weakens this assumption. It is unclear however, if this result is the same in the task of image classification.

After considering all these facts, it is not possible to meaningfully evaluate saliency methods by comparison or without violating fundamental assumptions. In the upcoming sections, potential evaluation metrics and methodologies are introduced to provide a more meaningful assessment. These metrics and methods do not fundamentally solve the problems, but they offer some guidance for evaluation.

4.1 Completeness and Soundness

In evaluation the composite input of the image $x \circ A$ describes an image with x% (depending on the type of evaluation) of the pixels replaced by a grey value or the mean value. The pixels are chosen according to the highest or lowest values in the attribution map. Those inputs are then measured using intrinsic methods. The hypothesis is: If the not removed pixels are not important and the original net outputs the same labels, then the removed pixels are not important. This hypothesis is wrong, because the position of gray pixels can also be informative. Gupta, 2022 [18] proposes Soundness and Completeness, which formalizes this problem and creates a numerical framework to evaluate the effectiveness of evaluating composition methods.

Soundness means: A certain portion of the image caused the net's output. x% pixels are responsible for the classification of the label.
Completeness means: No matter which mask is chosen, as long as the x% most important pixels are visible, the classification is correct.

This two metrics are useful to evaluate the correctness of masking. However, it is only a metric for other evaluation frameworks and is dependent on model and dataset. It is important to note however, because this metric was created in 2022 and is not included in any of the older papers. This information should give some background for the correctness of other older methods.

4.2 Evaluating the Visualization of What a Deep Neural Network has learned

In the paper by Samek[39] a method is described, where the most important pixels detected by an attribution method are systematically replaced with randomly sampled values using a greedy algorithm. The objective is to measure the impact on the classification value f(x). The method can be applied with only removing the important pixels or by defining a local neighborhood around the pixel which is replaced as well. This neighborhood is defined, because from the context of the nearby pixels the same information can be deduced.

While the method produces results which confirm that removing important pixels leads to a performance drop, the models are not retrained. This leads to an evaluation of a different data distribution and therefore the results are not objective.

4.3 A Benchmark for Interpretability Methods in Deep Neural Networks

This paper [20] presents two methods for numerically evaluating attribution maps. "RemOve And Retrain (ROAR)" and "Keep And Retrain (KAR)". This is done by removing supposedly informative features from the input and observing the response of the neural network. The section 6 gives an example for MNIST and Food-101.

ROAR can be applied to any method that produces attribution maps. It identifies 10, 30, 50, 70 or 90 % of the most important pixels in an attribution map and ranks them according to their numerical importance. In order to have a baseline against which to compare, a random baseline is generated which randomly identifies the x% most important pixels.
After the most important pixels have been identified, new datasets are generated for each threshold. In this step, the x% most important pixels are replaced by the mean value of the data set.
Once the new data sets have been generated, new models with the same architecture and learning parameters are trained on this modified data set. The same train/test split should be used as for the original model. This is done for several runs to ensure consistency of results. After retraining for each set of removed pixels and attribution maps, the results are compared. A random baseline should have a slower loss of accuracy than an attribution map. According to Hooker [20], a method is worse than a random baseline if this is the case.

KAR works in a similar way to ROAR, but instead of removing the most important pixels, it removes the least important pixels. As KAR performs worse than ROAR, it is not described further. The conclusion of [20] is that some commonly used attribution estimators like Gradient[40] , Integrated Gradient[43] and Guided Backpropagation[42] are worse than random assignments. On the other hand, the effectiveness of Smooth-Grad-Squared and VarGrad was confirmed. However, ROAR is sensitive to the dataset used. While it does provide a numerical evaluation, it is sensitive to the model and dataset used and therefore cannot independently and universally evaluate an algorithm.

4.4 Benchmarking Attribution Methods (BAM)

As already mentioned before, determining the ground truth of an attribution map is impossible. However, it is possible to compute the relative importance of specific features when comparing it between two different models[45].

In BAM[45], this leads to essential assumption:

Visualize an image consisting of two distinct pixel sets:
Each image contains one object, represented by the p_o pixels, and the remaining pixels p_s make up the background scene. By training two distinct classifiers, one trained to detect objects and the other trained for scene identification, we expect that the relative feature importance for the p_o pixels should be higher for the object classifier than in the scene classifier. Conversely, the same holds true for the p_s pixels for the scene classifier.
In BAM, such image-scene pairs are created systematically. To address the distribution shift problem, objects with a mean similar to the mean of the dataset are chosen so that they do not significantly alter the image distribution. Attribution maps are calculated for each image and for both trained classifiers. Those attribution maps are then compared using different metrics. However, the results show different results for various metrics and no clear result can be given to any method.
The advantage of BAM over ROAR [20] and other methods is the lower computational cost. No retraining or perturbation is required. However, the method has the usual problems of evaluation algorithms: sensitivity to the dataset and model and not offering a universal truth.

4.5 Sanity Checks for Saliency Maps

Adebayo [2] proposes two "sanity" tests to evaluate whether attribution methods are meaningful. It reveals that visual inspection by humans does not determine if the explanation is sensitive to the underlying model and data. Two instances of the framework are tested:
The model **parameter randomization test** involves comparing the output of a saliency method on a trained model with the output of a randomly initialized untrained network of the same architecture. The output should differ, otherwise, the saliency map is considered ineffective. An example for this test is visible in figure 6.3. The **data randomization test** randomly shuffles the labels of the data. Afterwards, the model is trained on the altered dataset. If the saliency maps do not differ from a normally trained model, then the method does not depend on the relationship of the images and labels.

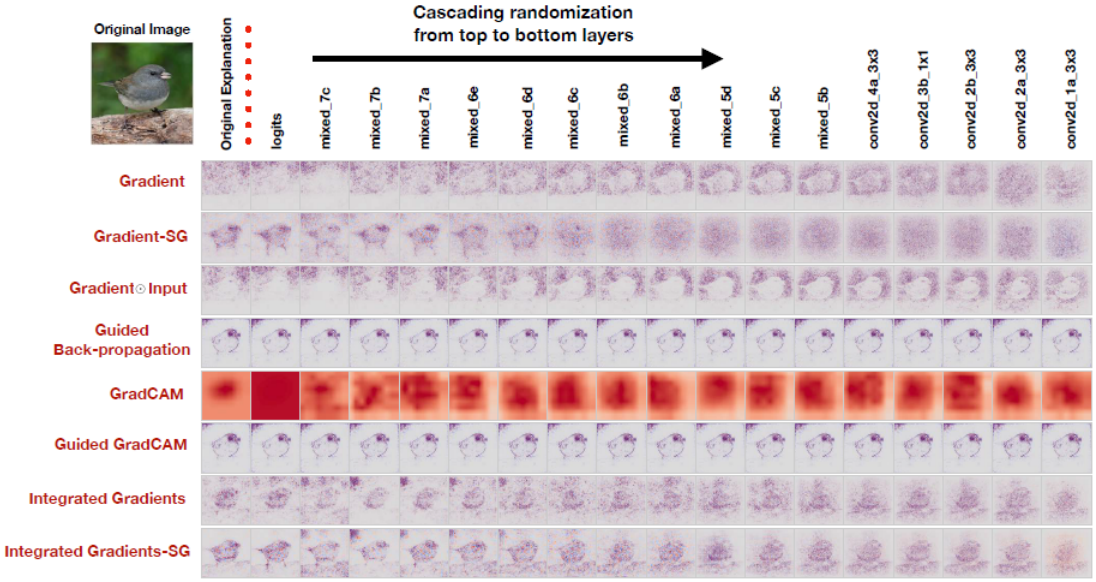

Figure 4.1: Cascading Randomization on Image Net. The figure shows the original explanations. Progression from left to right indicates complete randomization of network weights up to that block inclusive. The last block corresponds to a network with completely reinitialized weights. (Figure from [2])

If either of these two hypotheses fails during testing, the method under evaluation can be rejected, indicating that it fails a critical sanity check. On the tested methods, Guided BackProp & Guided GradCAM fail the sanity checks.

5 Concluding Thoughts on Attribution and Evaluation Methods

Numerous attribution maps and possible evaluation methods have been looked at and it leaves data scientists with one question: Which evaluation method and attribution method should I choose? The unsatisfactory answer is: We do not know the answer yet. Depending on the model and dataset used, different evaluation methods give varying outcomes. In Sanity check by Adebayo [2] and ROAR by Hooker[20] some methods were seen as unsuitable for the task considered. However, this was dependent on the model and the dataset. Considering there exists no ground truth for an evaluation method, there also is no ground truth for the attribution method. Opting for a SG - GRAD or a VAR-GRAD approach might result in favorable results for ROAR and Sanity checks. However, it remains difficult to evaluate visual explanations and there is no established ground truth to determine which evaluation method is the best. Consequently, further research is required to provide clearer guidance in this area.

6 Reconstructing ROAR

6.1 Scientific Motivation and Goal

After looking at various methods in theory, we look at an approach in detail. We decided to look into ROAR[20], because it a relatively recently published study. We try to reproduce some experiments. Not all are looked into, because of the computational complexity. Additionally, we also want to look into the fluctuations between the different runs.

Due to resource constraints, the research was limited to evaluating food-101 [8] using two interpretability methods along a baseline model. Instead of five control runs, only four were made. As an initial proof-of-concept experiment the evaluation was done on the MNIST dataset[12].

6.2 Project Setup MNIST

Before starting a big computation on the food-101 dataset, the method was tested on a smaller dataset (MNIST) and only with one interpretability method. Integrated Gradient was chosen. The advantage of this setup is that the information loss is predictable by humans. By looking at the images with gray pixels, the information loss can be seen by human inspection. This information loss is visible as completely blank images and images with only very few white pixels. Those few pixels are the same for every image and therefore humans can make the assumption, that there exists no difference between the classes.

A. Training the model

The model used in this report is a simple convolution neural network architecture consisting of two convolution layers, two pooling layers, and three linear layers. The model was trained for ten epochs with a learning rate of 1e-3 and an SGD optimizer with a momentum term of 0.9. The batch size was 64. Across five separate runs, the model achieved an average accuracy of 97.4% with a standard deviation of 0.4% on the test data.

B. Dataset preparation and Splitting

The MNIST dataset consists of 10 classes and 70.000 different images. It was chosen, because it is a small sized dataset and reconstructing the method is achieved easily. a) The importance of the

pixels is clear from a human perspective. b) The model required to achieve a solid performance is very modest and speeds up the retraining. c) The information loss in retraining is visible in the pictures.

Furthermore, it is a small, simple dataset. The images are 28x28 pixels big and one pixel has a value ranging from [0,1] in gray-scale. The dataset used in this report was normalized using the mean and standard deviation and was split using the PyTorch library's default splitting method (60.000 images for the training data, 10.000 images for the test data). This split has a fixed test and train set which include a homogeneous distribution of the classes. The training data was shuffled randomly using the inbuilt functions. The Cross entropy loss was selected for training runs. The same dataset split was used for all training runs to ensure consistency in the results.

C. Example of the modified dataset

The modified dataset is the result of combining the attribution map with the original image pixels $A \circ p$. In this operation the most x% most important pixels are replaced with the mean value of the dataset. In 6.1 the difference between Integrated Gradients and the Random Baseline is visible. In the first row, the random baseline is visible, in the second row the Integrated Gradient evaluation is shown. [10, 30, 50, 70, 90%] of the pixels are blacked out (left to right).

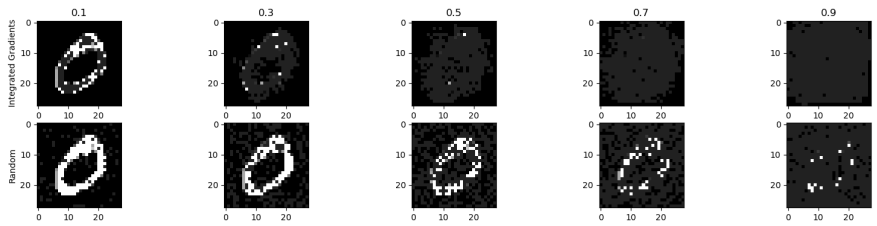

Figure 6.1: Integrated Gradients and Random Baseline Comparison

Judging by human intuition, Integrated Gradients appear to perform better overall thresholds. The model is expected to perform worse following retraining.

D. Retraining upon the new datasets

To ensure that the model learns consistently, the same split and learning parameters were utilized as in the standard case to retrain the model on the new dataset. An untrained network was created and then retrained for each training run. To validate the results and minimize the impact of random seed initialization, the training was performed with 5 different seeds and 5 models were created. In the Appendix, the exact results can be found.

E. Advantages and Disadvantages of the setup

Because of the small size of the dataset and the limited amount of classes, training a model is fast and effective. One disadvantage is the limited generalization of this setup for other datasets and

models. The majority of use cases primarily involve working with images in the RGB colour space, utilizing deep learning models that are often more complex, and dealing with a larger number of distinct classes. Furthermore, the pixels themselves are blurred in their importance. In RGB images 3 channels are combined, which makes attribution maps more difficult to compute and understand.

F. Results

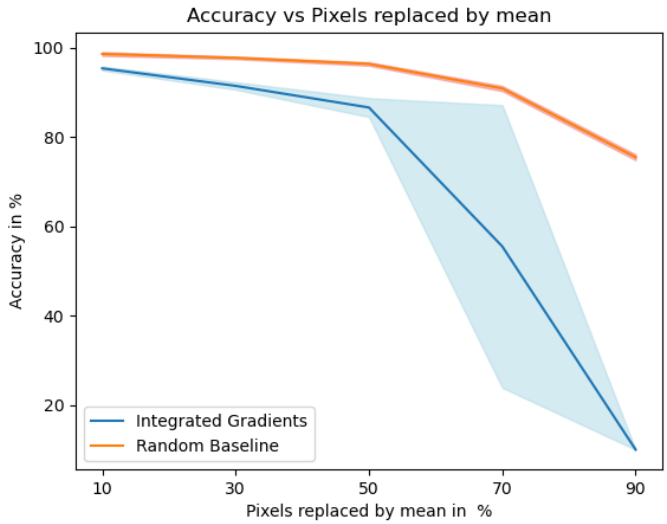

Figure 6.2: Accuracy: Random Baseline vs Integrated Gradient

Integrated Gradient is performing better than the random baseline.

6.3 Project Setup Food-101

The MNIST dataset [12] was chosen as proof-of-concept. Because calculating the gradients and judging if a mask is meaningful is easier here, it should give a reliable baseline for the results. Only integrated gradients is tested, as is it only an example.

A. Training the model

The model used in this report is ResNet50 [19]. The Food-101 [8] dataset was chosen to replicate. The learning rate was adjusted to 0.175 and the batch size was chosen as 64. Using an SGD optimizer with momentum of 0.9 and weight decay 0.0001 and a scheduler at epoch 30 with gamma=0.1, an accuracy of average 70.5% with a standard deviation of 0.01 was achieved over 5 separate training runs on the test score (No validation split was employed). In total 31 epochs were done. By using this particular training setup, at the last epoch, the model learns more about the specific data. In the last epoch, an accuracy of 95% on the training data was achieved.

In the original paper, an accuracy of 84.54% was achieved on the Food-101 dataset. A smaller accuracy was achieved, as different training parameters were used in our experiment. A smaller batch size of 64 instead of 256 and less epochs, 31 instead of 90 were used. More finetuning could have been done to achieve a better test score, but the results were deemed as good enough for our experiments.

The data was resized with center-crop to make all images 3x224x224. Afterwards, they were normalized. No data augmentation like flips and mirroring was applied. The standard split 75:25 was applied. The saliency maps to generate the datasets were computed using the same models.

B. Results

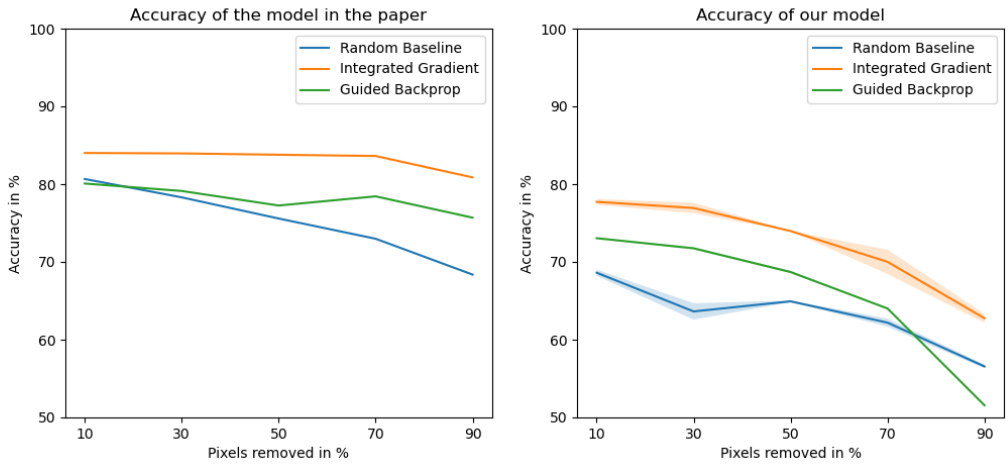

Figure 6.3: Comparison of the accuracy behaviour of the original model vs the modified model.

1. In the random baseline the results of the original paper are confirmed. A drop in performance can be explained by the loss of information.
2. In Integrated Gradient the results are also confirmed. But an unexpected behaviour occurs: The performance gets better by removing pixels.
3. The Guided Backprop result is also confirmed. But the behaviour is also unexpected. The performance is getting better by removing pixels.

Due to the fact, that the accuracy declines so slowly, it is unclear if the results given are really meaningful. Furthermore, there is no shared threshold when enough pixels are removed. To provide a theoretical explanation for the implications of removing input dimensions on the resulting accuracy, the following factors should be considered:

1. Input dimensions are removed and the accuracy drops:
   The removed input was informative for the model and their absence reduces the ability of the model to identify the right class.

2. We remove inputs and the accuracy does not drop.
   (a) It is possible that the removed input dimensions were not important for the model's decision-making process. The attribution map failed to identify important features, such as background pixels. (b) The input could be redundant and the information can be reconstructed using other available inputs.

3. We remove inputs and the accuracy increases:
   The network removed consistent information which is available in the training set. By removing this pattern, over-fitting is reduced and the generalization improves.

6.4 Summary and Interpretation

The main idea of ROAR was confirmed using the MNIST example. For simple datasets like MNIST, the concept can clearly be shown as meaningful.

The retraining performance of Food101 was partially confirmed, with a rise in accuracy by removing pixels contrary to the original runs.

The following questions are open:

- Is ROAR depending on the trained data and the dataset? Example: Using a dataset which does identify the species of animals. The face of the animal is clearly important. The eyes of the animals are also important. What is the relation of how important they are? How important is the zoom? Depending on the labels existing, the network learns different weights. In datasets like Birdsnap where the images are very common, only a small percentage of pixels are important and therefore blocking them out results in a lower accuracy.

- Does removing background pixels improve the accuracy because then the model focuses only on important patterns? Well, this depends on the dataset.

- By removing information which is important through the correct label, maybe a pattern of recognition emerges. All cakes have removed pixels at a position x. This could be the case for standardized pictures.

- Maybe removing the correct pixels to some degree makes it easier to recognize a pattern? For example, removing 50% of the face and 50% of the ears which are distinct to the species, more weight is given to the ears.

Knowing this -> Does ROAR still make sense?

While ROAR underlying logic makes sense, it is impossible to know whether the relation of a pixel makes sense and which patterns are learned. It does indeed offer an evaluation method, but the evaluation method in itself is not understandable. We do not know if the evaluation method does make sense, which simply moves the problem to a higher layer.

Bibliography

[1] Captum Library. URL: Source:%20https://captum.ai/tutorials/Resnet%5C_TorchVision%5C_Interpret.

[2] Julius Adebayo et al. *Sanity Checks for Saliency Maps*. 2020. arXiv: 1810.03292 [cs.CV].

[3] Genevera I. Allen, Luqin Gan, and Lili Zheng. *Interpretable Machine Learning for Discovery: Statistical Challenges & Opportunities*. 2023. arXiv: 2308.01475 [stat.ML].

[4] David Baehrens et al. "How to Explain Individual Classification Decisions". In: *Journal of Machine Learning Research* 11.61 (2010), pp. 1803–1831. URL: http://jmlr.org/papers/v11/baehrens10a.html.

[5] P. Berkhin. "A Survey of Clustering Data Mining Techniques". In: *Grouping Multidimensional Data* (2006), pp. 25–71. URL: http://dx.doi.org/10.1007/3-540-28349-8_2.

[6] H. Beyer. "Tukey, John W.: Exploratory Data Analysis. Addison-Wesley Publishing Company Reading, Mass. — Menlo Park, Cal., London, Amsterdam, Don Mills, Ontario, Sydney 1977, XVI, 688 S." In: *Biometrical Journal* 23.4 (1981), pp. 413–414. DOI: https://doi.org/10.1002/bimj.4710230408. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/bimj.4710230408. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/bimj.4710230408.

[7] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. "A training algorithm for optimal margin classifiers". In: *Proceedings of the fifth annual workshop on Computational learning theory*. 1992, pp. 144–152.

[8] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. "Food-101 – Mining Discriminative Components with Random Forests". In: *European Conference on Computer Vision*. 2014.

[9] Diogo V. Carvalho, Eduardo M. Pereira, and Jaime S. Cardoso. "Machine Learning Interpretability: A Survey on Methods and Metrics". In: *Electronics* 8.8 (2019). ISSN: 2079-9292. URL: https://www.mdpi.com/2079-9292/8/8/832.

[10] Tianqi Chen and Carlos Guestrin. "XGBoost". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, Aug. 2016. DOI: 10.1145/2939672.2939785. URL: https://doi.org/10.1145%2F2939672.2939785.

[11] Piotr Dabkowski and Yarin Gal. *Real Time Image Saliency for Black Box Classifiers*. 2017. arXiv: 1705.07857 [stat.ML].

[12] Li Deng. "The mnist database of handwritten digit images for machine learning research". In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 141–142.

[13] Finale Doshi-Velez and Been Kim. *Towards A Rigorous Science of Interpretable Machine Learning*. 2017. arXiv: 1702.08608 [stat.ML].

[14] Mengnan Du, Ninghao Liu, and Xia Hu. *Techniques for Interpretable Machine Learning*. 2019. arXiv: 1808.00033 [cs.LG].

[15] Alex Goldstein et al. *Peeking Inside the Black Box: Visualizing Statistical Learning with Plots of Individual Conditional Expectation*. 2014. arXiv: 1309.6392 [stat.AP].

[16] Brandon M. Greenwell, Bradley C. Boehmke, and Andrew J. McCarthy. *A Simple and Effective Model-Based Variable Importance Measure*. 2018. arXiv: 1805.04755 [stat.ML].

[17] Riccardo Guidotti et al. *A Survey Of Methods For Explaining Black Box Models*. 2018. arXiv: 1802.01933 [cs.CY].

[18] Arushi Gupta et al. *New Definitions and Evaluations for Saliency Methods: Staying Intrinsic, Complete and Sound*. 2022. arXiv: 2211.02912 [stat.ML].

[19] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[20] Sara Hooker et al. *A Benchmark for Interpretability Methods in Deep Neural Networks*. 2019. arXiv: 1806.10758 [cs.LG].

[21] Guolin Ke et al. "LightGBM: A Highly Efficient Gradient Boosting Decision Tree". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 3149–3157. ISBN: 9781510860964.

[22] Pang Wei Koh and Percy Liang. *Understanding Black-box Predictions via Influence Functions*. 2020. arXiv: 1703.04730 [stat.ML].

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Commun. ACM* 60.6 (2017), pp. 84–90.

[24] Tsung-Yi Lin et al. *Microsoft COCO: Common Objects in Context*. 2015. arXiv: 1405.0312 [cs.CV].

[25] Zachary C. Lipton. *The Mythos of Model Interpretability*. 2017. arXiv: 1606.03490 [cs.LG].

[26] Scott Lundberg and Su-In Lee. *A Unified Approach to Interpreting Model Predictions*. 2017. arXiv: 1705.07874 [cs.AI].

[27] Xiao-Li Meng. "Reproducibility, Replicability, and Reliability". In: *Harvard Data Science Review* 2.4 (2020). https://hdsr.mitpress.mit.edu/pub/hn51kn68.

[28] Christoph Molnar. *Interpable Machine Learning. A Guide for Making Black Box Models Explainable*. 2nd ed. 2022. URL: https://christophm.github.io/interpretable-ml-book.

[29] W. James Murdoch et al. "Definitions, methods, and applications in interpretable machine learning". In: *Proceedings of the National Academy of Sciences* 116.44 (2019), pp. 22071–22080. DOI: 10.1073/pnas.1900654116. eprint: https://www.pnas.org/doi/pdf/10.1073/pnas.1900654116. URL: https://www.pnas.org/doi/abs/10.1073/pnas.1900654116.

[30] Michael Neely et al. *Order in the Court: Explainable AI Methods Prone to Disagreement*. 2021. arXiv: 2105.03287 [cs.LG].

[31] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. "Feature Visualization". In: *Distill* (2017). https://distill.pub/2017/feature-visualization. DOI: 10.23915/distill.00007.

[32] Chris Olah et al. "The Building Blocks of Interpretability". In: *Distill* (2018). https://distill.pub/2018/
blocks. DOI: 10.23915/distill.00010.

[33] Vitali Petsiuk, Abir Das, and Kate Saenko. *RISE: Randomized Input Sampling for Explanation of Black-box Models*. 2018. arXiv: 1806.07421 [cs.CV].

[34] Liudmila Prokhorenkova et al. *CatBoost: unbiased boosting with categorical features*. 2019. arXiv: 1706.09516 [cs.LG].

[35] Marco Ribeiro, Sameer Singh, and Carlos Guestrin. "Anchors: High-Precision Model-Agnostic Explanations". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 32 (Apr. 2018). DOI: 10.1609/aaai.v32i1.11491.

[36] Ribana Roscher et al. "Explainable Machine Learning for Scientific Insights and Discoveries". In: *IEEE Access* 8 (2020), pp. 42200–42216. DOI: 10.1109/ACCESS.2020.2976199.

[37] Cynthia Rudin. *Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead*. 2019. arXiv: 1811.10154 [stat.ML].

[38] Wojciech Samek and Klaus-Robert Müller. "Towards Explainable Artificial Intelligence". In: *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Springer International Publishing, 2019, pp. 5–22. DOI: 10.1007/978-3-030-28954-6_1. URL: https://doi.org/10.1007%2F978-3-030-28954-6_1.

[39] Wojciech Samek et al. "Evaluating the Visualization of What a Deep Neural Network Has Learned". In: *IEEE Transactions on Neural Networks and Learning Systems* 28.11 (2017), pp. 2660–2673. DOI: 10.1109/TNNLS.2016.2599820.

[40] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. *Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps*. 2014. arXiv: 1312.6034 [cs.CV].

[41] Daniel Smilkov et al. *SmoothGrad: removing noise by adding noise*. 2017. arXiv: 1706.03825 [cs.LG].

[42] Jost Tobias Springenberg et al. *Striving for Simplicity: The All Convolutional Net*. 2015. arXiv: 1412.6806 [cs.LG].

[43] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. *Axiomatic Attribution for Deep Networks*. 2017. arXiv: 1703.01365 [cs.LG].

[44] Feiyu Xu et al. "Explainable AI: A Brief Survey on History, Research Areas, Approaches and Challenges". In: Sept. 2019, pp. 563–574. ISBN: 978-3-030-32235-9. DOI: 10.1007/978-3-030-32236-6_51.

[45] Mengjiao Yang and Been Kim. *Benchmarking Attribution Methods with Relative Feature Importance*. 2019. arXiv: 1907.09701 [cs.LG].

[46] Matthew D Zeiler and Rob Fergus. *Visualizing and Understanding Convolutional Networks*. 2013. arXiv: 1311.2901 [cs.CV].

[47] Matthew D. Zeiler, Graham W. Taylor, and Rob Fergus. "Adaptive deconvolutional networks for mid and high level feature learning". In: *2011 International Conference on Computer Vision* (2011), pp. 2018–2025. URL: https://api.semanticscholar.org/CorpusID:975170.

[48] Daniel Zhang et al. *The AI Index 2022 Annual Report*. 2022. arXiv: 2205.03468 [cs.AI].

[49] Jianming Zhang et al. "Top-Down Neural Attention by Excitation Backprop". In: *International Journal of Computer Vision* 126 (Oct. 2018). DOI: 10.1007/s11263-017-1059-x.

[50] Bolei Zhou et al. *Places: An Image Database for Deep Scene Understanding*. 2016. arXiv: 1610.02055 [cs.CV].

Appendix

1 MNIST-Computation

Precision and standard deviation of the retrained models per threshold.

Pixels blocked	10%	30%	50%	70%	90%
Random Baseline	98.64%	97.76%	96.4%	90.9%	75.5%
std	0.42%	0.28%	0.39%	0.58%	0.65%
Integrated Gradient	95.4%	91.48%	86.64%	55.52%	10%
std	0.44%	0.83%	2.1%	31.64%	0%

Table 1: Precision and standard deviation

2 Food101-Computation

Precision and standard deviation of the retrained models per threshold.

Pixels blocked	10%	30%	50%	70%	90%
Random Baseline	68.61%	63.63%	64.93%	62.19%	56.56%
std	0.40%	1.05%	0.10%	0.51%	0.25%
Integrated Gradient	77.72%	76.93%	73.97%	70.01%	62.75%
std	0.37%	0.67%	0.01%	1.56%	0.57%
Guided Backprop	73.05%	71.75%	68.70%	64.01%	51.56%
std	0.06%	0.15%	0.14%	0.01%	0.02%

Table 2: Precision and standard deviation