JOHANNES KEPLER
UNIVERSITY LINZ

Submitted by
Viktor Maximilian Loreth
k12006268

Submitted at
Institute of Computational Perception

Supervisor
Katharina Hoedt, PHD

October 5, 2023

EVALUATION OF IMAGE RECOGNITION NEURAL NETWORK INTERPRETATION METHODS: AN IN-DEPTH LOOK

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 5, 2023

JOHANNES KEPLER
UNIVERSITY LINZ

Abstract

This paper focuses on the accuracy of interpretability methods for machine learning models. The main research problem is the lack of ground truth for evaluation methods in interpretability methods. While inherent interpretative models exist, black-box networks perform better and have developed rapidly. Existing interpretability methods for neural networks such as ROAR, KAR, BAM and Real-Time Image Saliency for Black Box Classifiers provide a numerical evaluation method, but they still suffer from ambiguity. This paper summarizes the different evaluation methods and compares them. Additionally, the ROAR method has been reconstructed using two saliency methods.

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 . . . 7

3 Interpretability of Supervised Machine Learning Algorithms **9**
3.1 Global Model-Agnostic Methods . . . 10
3.2 Local Model-Agnostic Methods . . . 11
3.3 Neural Network specific Interpretability Methods . . . 11
3.3.1 Feature Visualization and Network Dissection . . . 12
3.4 Attribution Maps . . . 13
3.4.1 Vanilla Gradient & Deconv-Net . . . 14
3.4.2 Grad-CAM, Guided Backprop and Integrated Gradient . . . 15
3.4.3 Ensembling Methods: Smooth Gradient, Smooth Grad² and VarGrad . . . 16
3.4.4 Excitation Backprop [50] . . . 17
3.4.5 Advantages and Disadvantages of Saliency Maps [29] . . . 18

4 Evaluation of post-hoc Interpretability Methods **19**
4.1 Evaluation Metrics . . . 19
4.2 Completeness and Soundness . . . 20
4.3 Perturbation based Methods . . . 21
4.4 A Benchmark for Interpretability Methods in Deep Neural Networks . . . 21
4.5 Benchmarking Attribution Methods (BAM) [46] . . . 22
4.6 Sanity Checks for Saliency Maps [2] . . . 24
4.7 Validating the Research . . . 25

5 Research Findings: Which evaluation method should be used? **26**

6 Reconstructing ROAR **27**
6.1 Scientific Motivation and Goal . . . 27
6.1.1 Project Setup MNIST . . . 27
6.1.2 Project Setup Food-101 . . . 29
6.1.3 Theoretical Implications . . . 30
6.1.4 Interpretation of the Results . . . 31

Appendix **36**
1 MNIST-Computation 36
2 Food101-Computation 36

List of Figures

2.1 Decision Tree Example 7

3.1 Interpretability categorization 9
3.2 Network Dissection: Feature Visualization 12
3.3 Activation Maximization 13
3.4 Attribution Map [1] 14
3.5 CNN classifiers top-down attention map 17
3.6 Identifying task-relevant neurons in the network. 18

4.1 Cascading Randomization on Image Net. 24

6.1 Integrated Gradients and Random Baseline Comparison 28
6.2 Accuracy: Random Baseline vs Integrated Gradient 29
6.3 Comparison of the accuracy behaviour of the original model vs the modified model. 30

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 [49]. 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 [29].

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)[21] is shown.

1.1 Structure of the Thesis

1. In chapter 2 "Machine Learning and their Interpretability", an overview of contemporary machine learning algorithms, 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" we delve 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 3.4 "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 "Research Findings", 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 data set [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. Following key objectives have been identified: [30] [36] [29] [18] [26] [14] [9] [13]

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 [23].

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 [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 [18].

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 [30] [5] [6].

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, [29]. 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 data sets are separated as much as possible. Because there may be classification errors and noise in the data set, 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 (see Figure 1).

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 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.

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

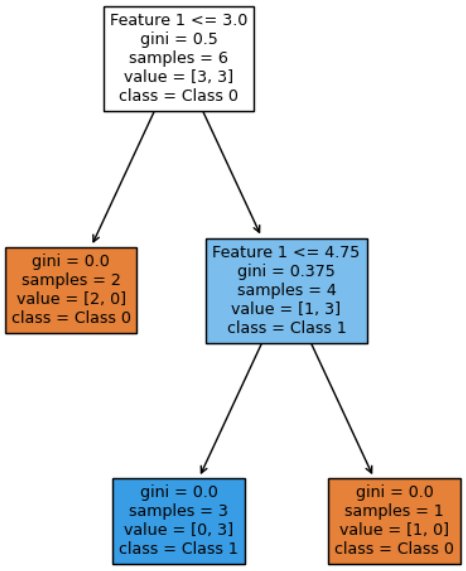

Figure 2.1: Decision Tree Example (created by Viktor Loreth)

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 [27] or partial dependence plots [17] can be employed to make them more understandable.
Gradient boosting like XGBoost [10], LightGBM [22] 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 strong predictive power makes them a common choice for classification tasks. But as they grow in complexity, the traditional approach of comprehending through weight examination becomes impossible. With the rise of CNN in 2012 [24] Neural Networks have become state-of-the-art for image prediction.

In a neural network, an neuron calculates a weighted sum of its inputs and applies an activation function to yield an output. This output then serves as the input for the subsequent layer. While various neuron types exist, including convolutional and recurrent neurons, the foundational neurons can be found in nearly all neural network architectures.

$x_j = f(\sum_{i=1}^{n} w_{ji} * x_i)$

- x_i input of a layer
- w_{ji} weights of the layer
- f activation function (e.g. sigmoid or ReLu)
- x_j output and furthermore the input of the next layer

Modern neural networks, such as ResNet50 [20] 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

After going through several supervised machine learning algorithms, an overview of interpretability methods is given. The goal of this section is to give the reader knowledge about potential interpretability methods which can be used before going to the next section, which evaluates the different interpretability methods and discusses the state of interpretability.

In this section, we present a framework [3] for classifying supervised machine learning methods and interpretability methods. Additionally, we introduce several commonly used interpretability methods and evaluate their suitability for image classification and neural networks. After introducing universal and specific techniques that are independent of the model, the text explains neural network interpretability methods and attribution maps, which represent the existing state of interpretability methods.

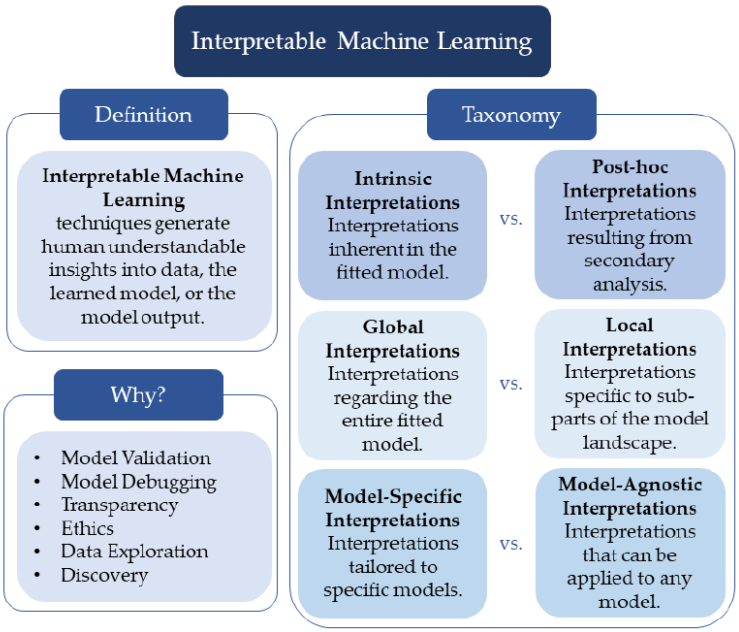

Figure 3.1: Interpretability categorization [3]

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 interpreta-

tions involve analyzing the model's behaviour after its creation.
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.

In this chapter, evaluation methods are described to analyze the behaviour of neural networks. To limit the scope of this thesis, only the application on neural networks will be discussed.

3.1 Global Model-Agnostic Methods

Global model-agnostic methods describe expected outcomes based on the distribution of the data. They can show a correlation between singular or multiple features and an outcome.

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 [17].

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 offers 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 commonly used in Neural Networks. In Chapter 3.3.1 a method to disassemble networks is presented.

5. **Permutation Feature Importance:** Permutation Feature Importance is regularly used in visual machine learning tasks. In section 4.3 an example of a perturbation method is shown.

6. **Prototype and Criticism:** Prototype and Criticism can be used as adversarial attacks in Neural Networks. [45] Evaluation methods use the underlying idea of prototype and criticism to evaluate saliency maps.

3.2 Local Model-Agnostic Methods

When it comes to explaining individual predictions, Local model-agnostic methods are used. A sub-part of the model can be explained.

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 numeric, text and 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. [16] 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. (Rashomon effect) The use of counterfactual explanations in image recognition as a standalone method is not common.
5. **SHAP (SHapley Additive exPlanations):** SHAP values [27] 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 prediction, while negative values indicate features that decrease it. It is used commonly in neural networks.

3.3 Neural Network specific Interpretability Methods

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 (CNN) are utilized. Given the immense amount of mathematical operations 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 evaluation 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.

3.3.1 Feature Visualization and Network Dissection

Modern Neural Networks like ResNet50 or Bard consist of several million layers.[32] Network dissection attempts to overcome this challenge by breaking down separate layers and connecting them with ideas.

The higher-level features in these networks relate to clear concepts, shown in Figure 3.2. As the features (image-pixels) pass through 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 changes image-based data into predictions.

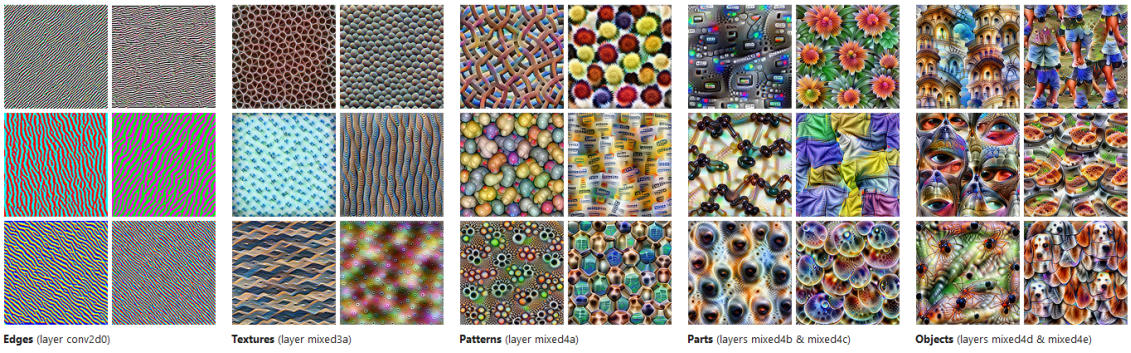

Figure 3.2: Network Dissection: Feature Visualization [32]

The image 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 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.

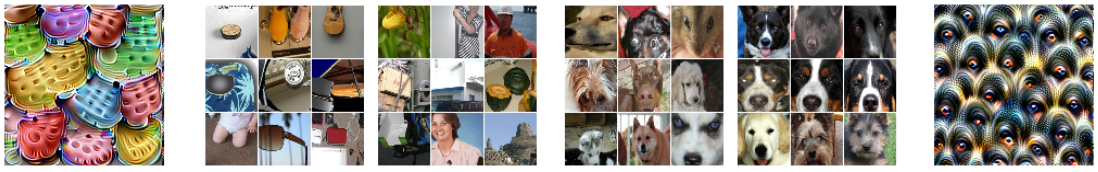

Figure 3.3: Activation Maximization [32]

Advantages of Feature Visualization:

1. **Initial Model Insights:** Feature visualization offers an initial view into a model's behaviour, improving the understanding of its inner layers.
2. **Enhanced Domain Understanding:** It has the potential to enrich domain understanding by aligning learned features with domain-specific knowledge. An example can be seen in the medical industry
3. **Debugging and Improvement:** Feature visualization assist in debugging and refining models, contributing to their overall performance enhancement.

Disadvantages of Feature Visualization:

1. **Unclear Decision-Making:** While activations are evident, understanding the meaning behind them and how they contribute to decision-making remains challenging.
2. **Subjective Interpretation:** The interpretation of visualized features can be subjective, potentially leading to differing conclusions among observers.
3. **Limited Applicability to Visual Data:** Feature visualization applicability is limited to visual data types.

3.4 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.

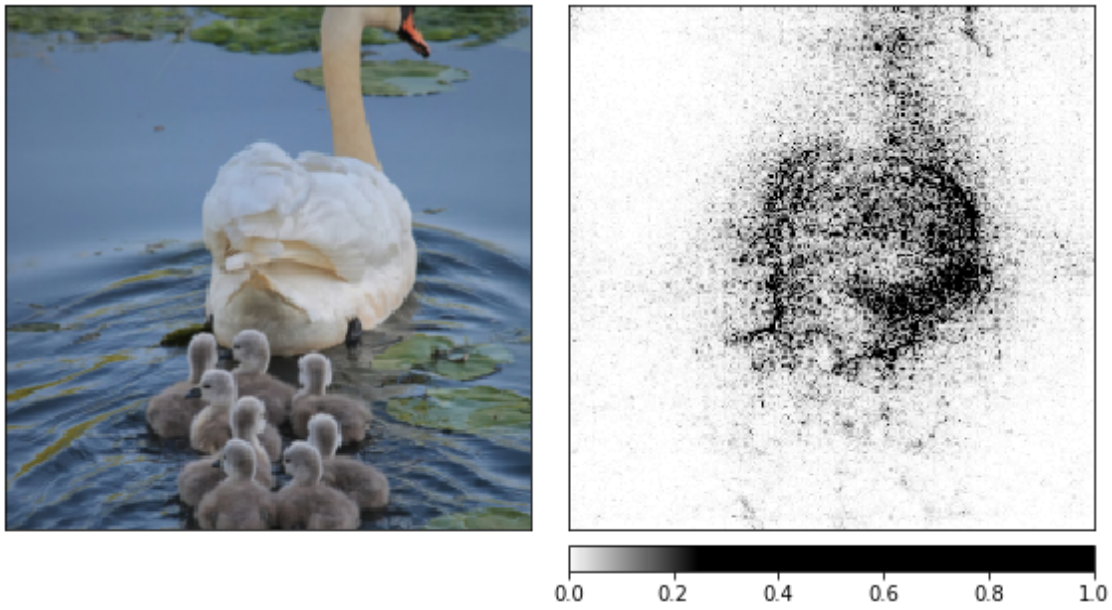

Figure 3.4: Attribution Map [1]

Attribution maps are commonly calculated using SHAP [27] or gradient methods. Attribution maps provide a direct and intuitive way to understand which parts of an input data are influencing a model's decision. The main difficulty is the generation of reliable attribution maps. In the following subsection, some commonly used methods are introduced.

3.4.1 Vanilla Gradient & Deconv-Net

Vanilla 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, Vanilla Gradient faces two challenges. First, when Rectified Linear Units (ReLU) are used, negative gradients can lead to information loss. Second, in pooling layers, gradients are absent, resulting in further information loss.
Deconv-Net [47] addresses these gradient problems. It starts with a conventional convolutional network that includes convolutional layers, fully connected layers, and activation functions. Each input image, denoted as x_i, maps to a probability layer, denoted as y_i.
A single Deconv-Net [48] is attached to each layer providing a path back to the image pixels. This process involves three key steps:
Unpooling: Pooling operations are non-invertible, but Deconv-Net records the original locations of the maxima using a set of switch variables. This information is then used to reconstruct the

activations.

Rectification: Just as each layer in the forward pass passes through a ReLU non-linearity, the deconvnet applies a ReLU non-linearity in reverse to maintain positive signals during reconstruction.

Filtering: Deconv-Net employs inverted original filters. It uses transposed versions of these filters, essentially flipping each filter vertically and horizontally.

This results in the gradients methods [4] making it possible for visualizing the output activation of interest A_n^l with respect to x_i:

$$e = \frac{\delta A_n^l}{\delta x_i}$$

3.4.2 Grad-CAM, Guided Backprop and Integrated Gradient

Gradient-weighted Class Activation Map (Grad-CAM) provides visual explanations for Convolutional Neural Networks. The primary objective is to understand which regions in a convolutional layer are responsible for the classification.

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\delta y^c}{\delta A_{ij}^k}$$

c = class of interest
A^k = feature map
i,j = width and height dimension
Z = average term

In the context of Grad-CAM, uninteresting classes are set to zero. The method then backpropagates the gradients of the primary class of interest. These gradients are scaled within the [0,1] range and displayed. It is worth noting that Grad-CAM considers all convolutional feature maps except the last. The disadvantage of Grad-CAM is its relatively coarse accuracy.

Guided Backprop combines the backpropagation [42] with another method to have a better localization. It uses a modified backpropagation step that only focuses on positive gradients from the ReLu activation function. Guided Backprop works as a lens to focus on specific parts of the pixel-wise attribution map.

Integrated Gradients [43] aims to improve localization by combining traditional backpropagation with an additional technique. Its key objective is to assign importance to input features by dissecting the output activation A_n^l into contributions from individual input features. Integrated Gradients achieve this by interpolating a series of estimates for values between a non-informative reference point x^0 and the actual input x. This interpolation is done by summing values at small intervals between x^0 and x, represented as:

$$e = (x_i - x_i^0) \times \sum_{i=1}^{k} \frac{\delta f_w(x^0 + \frac{i}{k}(x - x^0))}{\delta x_i} \times \frac{1}{k}$$

e represents the final estimate of feature importance.
k is the number of intervals used for approximation.

The choice of k and the selection of the reference point x^0 significantly impact the final estimate e. As suggested by [43], it is common to use a black image as the reference point and set k to a value of 25.

3.4.3 Ensembling 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 technique used to enhance the performance of gradient-based methods. It introduces a smoothing approach that 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. These noisy estimates are then averaged to obtain a more reliable and accurate gradient estimate, as represented by the formula:

$$e = \sum_{i=1}^{J} (g_i(x + \theta, A_n^l))$$

SmoothGrad² [21] 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:

$$e = \sum_{i=1}^{J} (g_i(x + \theta, A_n^l)^2)$$

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:

$$e = Var(g_i(x + \theta, A_n^l)^2)$$

3.4.4 Excitation Backprop [50]

Excitation backpropagation focuses on a set of neurons within a neural network during the computational process. By doing so, it enables the combination of gradient information with the importance scores, leading to precise object localization within the network's feature maps.

The key feature of Excitation Backprop is the use of a probabilistic Winner-Takes-All (WTA) formulation. This formulation produces normalized attention maps, allowing for direct subtraction of these attention maps. This subtraction operation is crucial for highlighting the regions of interest within the neural network's activation, making it easier to understand which parts of the input data contribute most to a particular decision or classification.

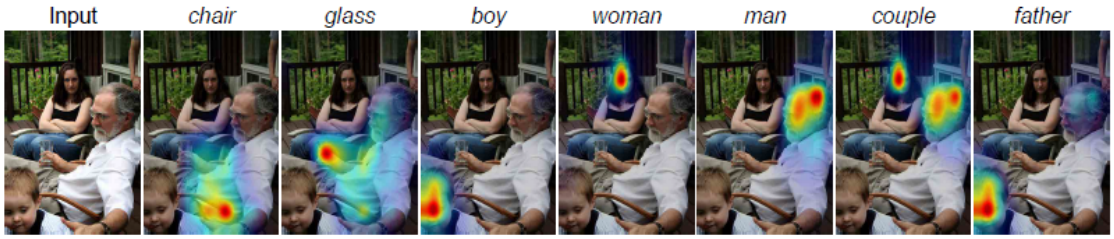

Figure 3.5: CNN classifiers top-down attention map [50]

The concept of identifying task-relevant neurons is central to Excitation Backprop. This involves evaluating the relative likelihood of a neuron "winning" against others within the same layer. Neurons with higher winning probabilities are deemed more relevant to the task at hand, and this information can be leveraged to understand the network's decision-making process and to localize objects or features within the input data.

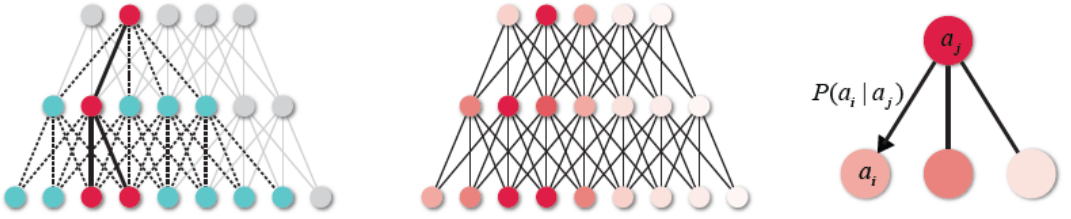

Figure 3.6: Identifying task-relevant neurons in the network. The red shading of a dot indicates its relative likelihood of winning against the other ones in the same layer. [50]

3.4.5 Advantages and Disadvantages of Saliency Maps [29]

The advantage of saliency maps is that they are fast to compute and easy to interpret. The explanations are visual and can immediately be recognized. Furthermore, there are many methods to choose from.
The disadvantages are unreliability, insensitivity to model and data and difficulty to know whether an explanation is correct. These topics are covered in the next chapter.

4 Evaluation of post-hoc Interpretability Methods

Gradient methods which generate saliency maps are hard to measure. While humans can evaluate the maps by giving a general statement, this can not be applied to thousands of images. Despite many significant recent contributions to saliency maps, the valuable effort of explaining machine learning models faces this methodological challenge: the difficulty of assessing the scope and quality of model explanations [2].

4.1 Evaluation Metrics

Various evaluation methods exist to assess the performance of saliency methods. They can be broadly categorized into two approaches: [19]

Extrinsic Evaluation: This approach involves human evaluations and comparisons against predefined ground truth explanations from experts.[50] The biggest disadvantage is that an expert needs to spend a lot of time evaluating the samples.
Intrinsic Evaluation: In contrast to extrinsic 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 the pre-trained trained model. (as in [11]) However, it is important to note that intrinsic evaluation violates a fundamental assumption in machine learning – that training and evaluation data should come from the same distribution. Without re-training it is not clear if the degrade in performance stems from the distribution shift or the removal of informative features. [21]
Additionally, Neely, 2021[31] states that comparing the rank correlation between attention weights and feature-additive methods is not appropriate. This means, using agreement as evaluation (using a method as a standard to compare to) should not be used. In the paper low empirical evidence was found, that models are similar. This means that just evaluating similar methods and taking the common results is not applicable. Instead, the use of rigorously defined metrics is recommended to ensure robust and meaningful evaluations of saliency methods.
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.

4.2 Completeness and Soundness

Gupta, 2022 [19] proposes Soundness and Completeness, two concepts which are required for evaluation metrics which involve using a composition of a heat map and the original input.
Soundness is needed: The masked input method proves that a certain part of the image "caused" the output of the network.

The method is α complete on f,x,a if:

$$g_{AUC}(x, a, m) >= \alpha f(x, a)$$

The method is β sound on f,x,a if:

$$g_{AUC}(x, a, m) <= \frac{1}{\beta}\alpha f(x, a)$$

$$\alpha(x, a) = min(\frac{max(g_{AUC}(x, a, n), \epsilon_1)}{f(x, a)}, 1)$$

$$\alpha(x, a) = min(\frac{max(f(x, a), \epsilon_2)}{g_{AUC}(x, a, n)}, 1)$$

Then $\alpha(m) = E_x[min_a(\alpha(x, a))]$
Then $\beta(m) = E_x[min_a(\beta(x, a))]$
If completeness α is close to 1:
The method still predicts the right label after applying the saliency map. This means the saliency map correctly identifies the pixels responsible for the classification.
If soundness β is close to 1:
The blocked input does not predict the right label with not enough information present. This means the method does not identify the class with high probability without enough information available.

x: input to the model
a: true label
f: model
m: saliency method.
ϵ = term if x is very small

Older evaluation methods only try to maximize the g_{AUC} curve [33]:
For s = [1, dim(x)] take the top s pixels as per saliency map m and plot the probability f(x, a) given by the model. The top s pixels of x are retained and the remaining pixels are assigned a default

value). Return the area under the curve.

This completely ignores if the method is overconfident and thus does not provide a meaningful evaluation.

Limitations:

There is an absence of human interpretability in this approach. Additionally, the masked images do not share the same distribution as the original trained maps, which could potentially lead to an undesired performance decrease. It is important to note that these masked images should not be seen as a substitute for other evaluation methods, as they cannot provide a comprehensive solution. Therefore, they should not be relied upon to validate a saliency method due to these limitations.

4.3 Perturbation based Methods

In the paper by Samek, 2017 [39] a method is described, where the most important pixels are systematically replaced with randomly sampled values. The objective is to measure the impact on the label value f(x). Depending on the method used, the method is more or less effective. If the edges are the most important in a saliency map, then deleting them results in a steep decrease. It is important to note that the method comes with significant computational overhead. It is also unclear which pixels to remove and how to adjust the values. Again, the distribution of the image also is altered, which leads to the vagueness of the implicit reason behind an accuracy drop.

4.4 A Benchmark for Interpretability Methods in Deep Neural Networks

In this paper [21] two methods to estimate the effectiveness of saliency maps are presented. "ReмOve And Retrain (ROAR)" and "Keep And Retrain (KAR)".

ROAR proposes a numerical solution to evaluate attribution maps. It is an algorithm which estimates the effectiveness of saliency maps. This is done, by removing supposedly informative features from the input and observing the reaction of the neural network. ROAR applies to any visual domain. In Section 6, an example for MNIST and Food-101 is done.

RoaR works as follows:

1. **Selection of the most important pixels from an attribution map**
   An attribution map provides a mapping to the most important pixels. Those pixels are ranked based on their importance. Depending on the algorithm, different values differ in importance. In the case of Integrated Gradient, the pixels with the highest absolute values are considered the most important.

2. **Replacement of x% of the pixels with the mean value**
   In the Roar algorithm, 0.1, 0.3, 0.5, 0.7 or 0.9 % of the most important pixels are identified and replaced by the mean value. Both the training and the testing data undergo this process.

3. **Retraining the model using the modified dataset**
   To ensure the consistency of the model, retraining is necessary. Afterwards, the model is evaluated using several seed runs on the test set. The same split should be used for all runs. This is crucial because the training data and the test data must be drawn from the same distribution. Without retraining the model this property is ignored.

4. **Comparison of the attribution map with a random baseline**
   For each training setup, respectively with 0.1, 0.3, 0.5, 0.7 or 0.9 % pixels replaced, a random baseline is considered. The random baseline also replaces the same percentage of pixels. The random baseline is expected to perform worse than a sophisticated method.

5. **Evaluation**
   An attribution method is deemed effective if it consistently outperforms the random baseline across various setups.

KAR operates similarly to ROAR, but instead of removing the most important pixels, the least important pixels are removed. Because KAR performs worse than ROAR, it is not described further. The conclusion of [21] is that some commonly used attribution estimators like Gradients, Integrated Gradient and Guided BackProp 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 data set used. While it does provide a numerical evaluation, it is sensitive to the model and data set used and therefore cannot independently and universally evaluate the effectiveness of an algorithm.

4.5 Benchmarking Attribution Methods (BAM) [46]

In practice, it is often challenging to determine the absolute importance of a feature. However, it is possible to calculate the relative importance of one feature concerning one model compared to another. Given the relative feature importance, the metrics compare attributions between pairs of models and pairs of input.

The core concept behind BAM is straightforward: By introducing a grey square into every training image across various classes, it is expected that this square would be considered less important than the original image region it covers. Similarly, if an object (not present in the dataset) is added to every image, it should also be assigned lower importance than the original pixels. Any explanations that assign higher importance to the inserted object over the original pixels are considered false positives.

To address the challenge of distribution shift, BAM carefully selects objects to insert that do not significantly alter the overall image distribution. For instance, objects with means similar to the dataset's mean are chosen.

The BAM algorithm works as follows:

1. **BAM dataset construction**
   The BAM dataset is constructed by pasting object pixels from MSCOCO [25] into scene images from MiniPlaces [51]. An object is re-scaled to between 1/3 to 1/2 of a scene image at a randomly chosen location. The resulting images have an object label and a scene label. Either can be used to train a classifier. Every object class appears in every scene class and vice versa. Scenes which contain original BAM objects are not used.

2. **Common features and commonality**
   Common features are defined as a set of pixels with semantic meaning (e.g. looks like a dog) which commonly appear in all examples of one or more classes. For example, a dog which appears in all bamboo forests is less common than a dog which appears in all images of bamboo forests, bedrooms and corn fields.

3. **2 Classifiers are trained and attribution maps are created**
   An object detection classifier and a scene detection classifier are trained. Objects should be significantly more important than the scene to the object detector than to the scene detector. To verify this intuition, the objects are removed. With this knowledge, attribution maps can be tested by checking if the pixels are assigned noticeably higher attributions.

4. **Relative importance is calculated.**
   With this knowledge, the object pixels should be more important than any other pixels for the detection classifier. Additionally, for the scene attribution, the attribution maps should be higher if the object pixels are replaced with the original pixels.

The advantage of BAM over ROAR [21] 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 data set and model and not offering a universal truth.

4.6 Sanity Checks for Saliency Maps [2]

This paper proposes an actionable methodology, for assessing the limitations of explanation methods. 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:

1. **Model Parameter randomization test**
   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.

2. **Data randomization test**
   The data randomization test randomly shuffles the labels of the data. Afterwards, the model is trained on the altered data set. 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.

   If either of these two hypotheses fails during testing, the method under evaluation can be rejected, indicating that it fails a critical sanity check. The paper conducts extensive experiments across various datasets and model architectures. On the tested methods, Gradients & GradCam pass the Sanity checks, while Guided BackProp & Guided GradCAM fail.

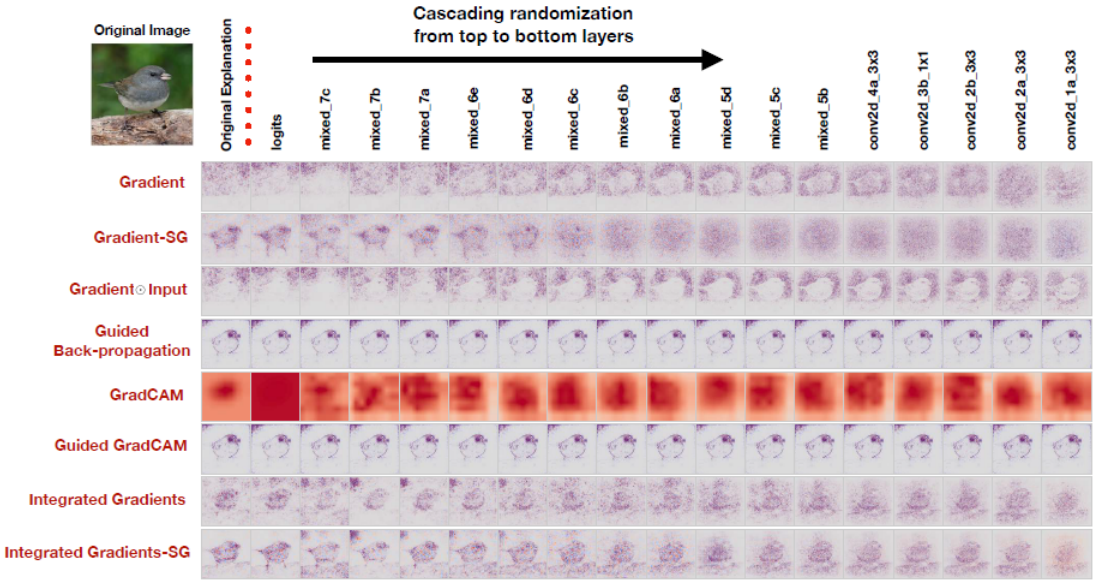

Figure 4.1: Cascading Randomization on Image Net. The figure shows the original explanations. Progression from left to right indicates complete randomization of network weights up to that block inclusive. The last block corresponds to a network with completely reinitialized weights. [2]

4.7 Validating the Research

After understanding the several options for evaluation one common problem is identified: validating the research [3]. How can a method be accepted or rejected?

1. Replicability, Reliability and Trust. Evaluation methods need to be repeatable to some degree. Predictions and findings should be robust to sensitivity tests like minimal changes in the data or the model, and out-of-sample prediction tests. They also should align consistently with domain knowledge. [28]

2. Why do machine learning interpretations fail?

a. The machine learning model could be a poor fit for the model and any resulting interpretability approach would poorly reflect the signal in the data.

b. The interpretability approach could be a poor fit for the model. Especially when fitting a second model to generate the interpretation, like LIME.

c. There can be a mismatch between the employed interpretability technique and the desired discovery task. For example, attempting to understand why a model makes incorrect predictions may be impossible if the saliency map generated is of poor quality.

For prediction tasks established techniques for validation were developed: Using a split for train, test and validation data. Using this the predictive model should generalize well to new, unseen data. one may consider utilizing a training and test set for validation. [3] This prospect is way more complicated for interpretability. There exists no fair metric that determines which interpretation is the best. Using human evaluation or domain experts could also be an option, but this also faces the challenge of missing a robust evaluation method.

Validating discoveries from interpretable machine learning is still a challenge. It underscores the need for further research to promote trustworthy and reliable data science.

5 Research Findings: Which evaluation method should be used?

Numerous saliency maps and possible evaluation methods have been introduced, leaving data scientists with one question: Which saliency map and evaluation method should I choose?
The unsatisfactory answer is: There is no definitive right and wrong answer. Depending on the model and data set used, different evaluation methods give varying outcomes. In Sanity checks for Saliency Methods [2] and ROAR[21] some methods were seen as unsuitable for the task considered.

Opting for a SG - GRAD or a VAR-GRAD approach results in favorable results for ROAR, and SG - Grad also performs well in Sanity Check for saliency maps. Excitation Backprop has not been analyzed yet with this method, but could also prove trustworthy.
However, it is important to state there is not enough reliable research to state one method is superior to others. 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

In the ROAR benchmark study [21] the paper omits the standard deviation of the trained nets. A validation of the results is expected by achieving similar results.

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. Additionally, an evaluation using the MNIST dataset is added.

6.1.1 Project Setup MNIST

The MNIST dataset [12] was chosen as a 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 are tested, as is it only an example.

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 data set consists of 10 classes and 70.000 different images. It was chosen, because it is easy to evaluate the effectiveness of integrated gradients and ROAR:
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, usable data set. The images are 28x28 pixels big and one pixel has a value ranging from [0,1] in gray-scale. The data set 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). The training data was shuffled randomly using the inbuilt functions. The Cross entropy loss was selected for training runs. The same data set split was used for all training runs to ensure consistency in the results.

C. Example of the new 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).

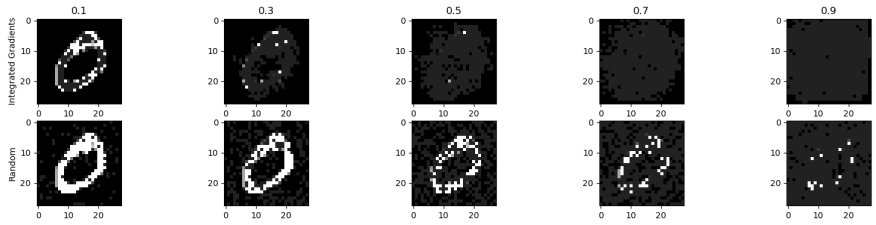

Figure 6.1: Integrated Gradients and Random Baseline Comparison

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 data set. 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 data set 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

Integrated Gradient is performing better than the random baseline.

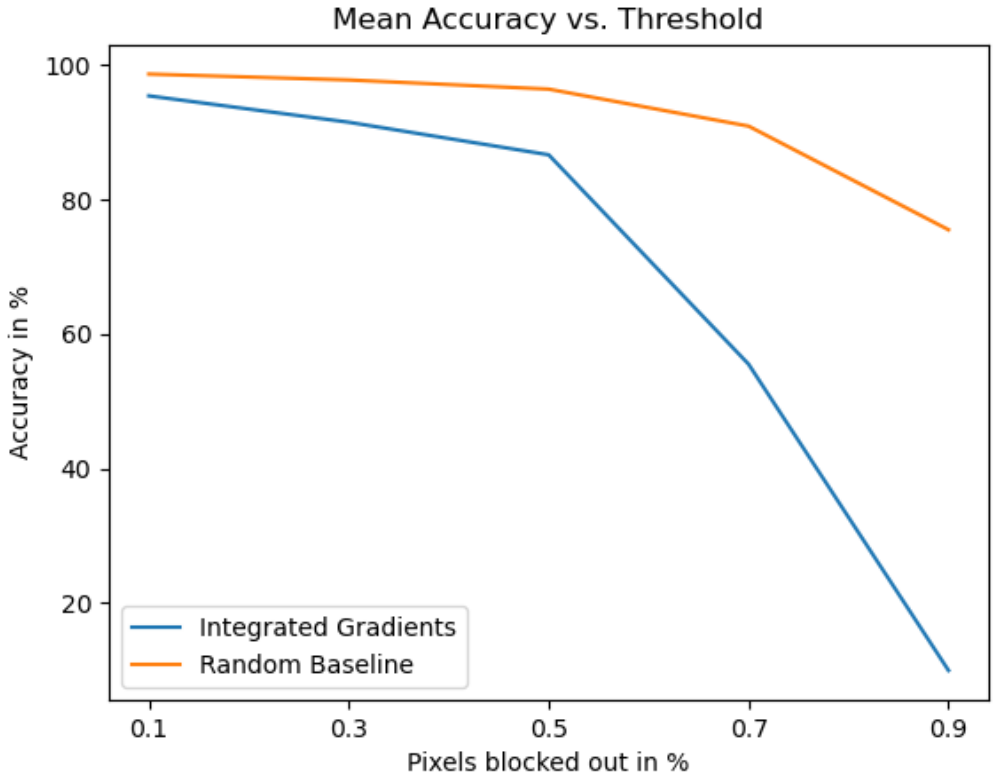

Figure 6.2: Accuracy: Random Baseline vs Integrated Gradient

6.1.2 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 [20]. 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 =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. 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 the training was noticeably shorter because only an NVIDIA GTX 1080 was available.

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

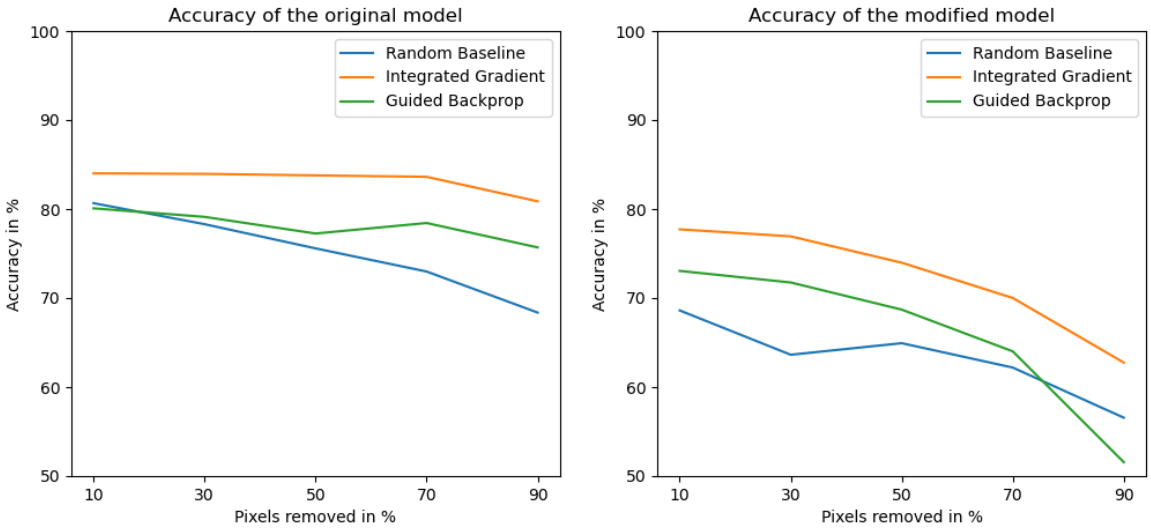

Figure 6.3: Comparison of the accuracy behaviour of the original model vs the modified model.

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.

6.1.3 Theoretical Implications

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.

6.1.4 Interpretation of the Results

The main idea of ROAR was confirmed using the MNIST example. For simple data sets 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 data set? 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 data sets 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 data set.

- 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] Jerome H. Friedman, Trevor Hastie, and Rob Tibshirani. "Regularization Paths for Generalized Linear Models via Coordinate Descent". In: *Journal of Statistical Software* 33.1 (2010), pp. 1–22. DOI: 10.18637/jss.v033.i01. URL: https://www.jstatsoft.org/index.php/jss/article/view/v033i01.

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

[17] 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].

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

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

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

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

[22] 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.

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

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

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

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

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

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

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

[30] 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.

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

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

[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] Han Xu et al. *Adversarial Attacks and Defenses in Images, Graphs and Text: A Review*. 2019. arXiv: 1909.08072 [cs.LG].

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

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

[48] 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.

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

[50] 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.

[51] 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