UNIVERSITY
OF OSLO

Master's thesis

Explainable Artificial Intelligence for Out-of-Distribution Detection

Using irregularities in machine learning explanations to detect when a model is faced with unusual data

Jonatan Hoffmann Hanssen

Robotics and Intelligent Systems
60 ECTS study points

Department of Informatics
Faculty of Mathematics and Natural Sciences

Spring 2025

Jonatan Hoffmann Hanssen

Explainable Artificial Intelligence for Out-of-Distribution Detection

Using irregularities in machine learning explanations to detect when a model is faced with unusual data

Supervisors:
Hugo Lewi Hammer
Kyrre Harald Glette

Abstract

As Artificial Intelligence (AI) becomes a larger and larger part of society, the need for robust and understandable models becomes paramount. When neural networks are used in high-impact settings such as cancer detection or autonomous driving, we must not only require that they make predictions with high accuracy, but also that they are aware of their shortcomings, and alert us when faced with unusual data. The need for models which "know what they do not know" leads to the field of Out-of-Distribution (OOD) detection, which attempts to detect when models are exposed to data points that are far outside of their training data and thus unlikely to be classified correctly. Effective OOD detection makes AI models safer and more robust, and can enable them to be used for tasks where the consequence of failure is severe. OOD detection is a young and developing field, and there are, to date, no methods which achieve superior performance on all benchmarks. Thus, there is a need for novel methods which push the field forward.

Explainable Artificial Intelligence (XAI) is another field which is concerned with making AI models more trustworthy and robust. Like OOD detection, this field has seen increased interest as AI methods are used for a wider array of tasks than previously. XAI methods attempt to understand how AI models come to a particular decision, and use gradient information, counterfactuals and model internals to create explanations of predictions which can be inspected by humans. These methods, although they are intended primarily for human inspection, may also capture intricacies of AI models which could be used to detect OOD data points. To date, only one work has attempted to combine XAI and OOD detection in this way, with very poor results. This thesis further explores the possibility of using XAI explanations for OOD detection.

In this thesis, three XAI OOD detection frameworks have been developed, and methods under these three frameworks have been rigorously tested on all four standard OOD detection benchmarks. The results are very promising, and show that XAI methods can indeed be used for OOD detection, contrary to previous research. XAI OOD detection methods can compete with baseline OOD detection methods, and even surpass them in many cases. In addition, on three out of four benchmarks, a method developed under one of the frameworks created in this thesis achieves results which are quite close to State-of-the-Art (SoTA) OOD detection methods.

Contents

1 Introduction . . . 1
1.1 Motivation . . . 1
1.2 Problem Statement . . . 2
1.3 Scope . . . 3
1.4 Research Methods . . . 3
1.5 Ethical Considerations . . . 3
1.6 Main Contributions . . . 4
1.7 Thesis Outline . . . 5
2 Background . . . 7
2.1 Machine Learning . . . 7
2.1.1 Supervised Learning . . . 7
2.1.2 Unsupervised Learning . . . 8
2.1.3 Reinforcement Learning . . . 8
2.2 Neural Networks . . . 8
2.2.1 Feed Forward Neural Networks . . . 8
2.2.2 Convolutional Neural Networks . . . 9
2.3 Model evaluation . . . 11
2.3.1 Accuracy . . . 11
2.3.2 Metrics utilizing the binary classification confusion matrix . . . 12
2.3.3 Threshold Independent Metrics . . . 13
2.4 Explainable Artificial Intelligence . . . 14
2.4.1 The motivation for Explainable Artificial Intelligence. . . . 14
2.4.2 Taxonomy of Explainable Artificial Intelligence. . . . 15
2.4.3 XAI methods adapted to images: Saliency maps and segmentation . . . 16
2.4.4 Specific methods . . . 18
2.5 Out-of-Distribution Detection . . . 22
2.5.1 Motivation for Out-of-Distribution Detection . . . 22
2.5.2 Semantic versus covariate shift . . . 23
2.5.3 Benchmarking . . . 25
2.5.4 Methods . . . 25
2.5.5 Specific methods . . . 27
2.6 Related work . . . 31
2.7 Summary. . . . 32
3 Methodology . . . 33
3.1 Proposed XAI frameworks for OOD detection . . . 33
3.1.1 Stand-alone saliency framework: Saliency Aggregation . . . 33
3.1.2 Saliency integrated into existing OOD detection algorithms . . . 41

3.2 Relation to existing methods . . . 44
3.3 Benchmarks . . . 46
3.3.1 CIFAR10/CIFAR100 . . . 46
3.3.2 ImageNet200/ImageNet1K . . . 49
3.3.3 Overview of testing environment . . . 50
3.4 Networks . . . 51
3.5 XAI Saliency Methods . . . 52
3.5.1 LIME . . . 52
3.5.2 Occlusion . . . 53
3.5.3 GradCAM . . . 53
3.5.4 Guided Backpropagation . . . 53
3.5.5 Integrated Gradients . . . 53
3.6 Evaluation . . . 53
3.6.1 Metrics . . . 53
3.6.2 Statistical Analysis of Results . . . 55
3.7 Implementation . . . 56
3.7.1 Basic hardware and software . . . 56
3.7.2 Method Evaluation: OpenOOD . . . 56
3.7.3 Implementation of Saliency Methods . . . 58
3.8 Summary . . . 59
4 Experiments and Results . . . 61
4.1 Data Analysis of Saliency Maps . . . 61
4.1.1 ImageNet200 . . . 61
4.1.2 CIFAR10 . . . 72
4.1.3 Overall results on both validation benchmarks . . . 82
4.2 Evaluation of XAI OOD detectors . . . 83
4.2.1 Results for Saliency Aggregation . . . 84
4.2.2 Results for Saliency Aggregation plus Logit . . . 92
4.2.3 Results for SaliencyVIM . . . 101
4.3 Summary . . . 109
5 Discussion . . . 111
5.1 Answering the problem statement . . . 111
5.2 Limitations of the Results . . . 111
5.3 Deeper analysis of the results . . . 111
5.3.1 Analysing the reasons for the effectiveness of XAI based OOD detection . . . 112
5.3.2 Analysing the difference in performance between magnitude and dispersion aggregation . . . 112
5.3.3 Analysing the poor performance of occlusion saliency aggregation . . . 114
6 Conclusion . . . 117
6.1 Thesis summary . . . 117
6.2 Main contributions . . . 118
6.3 Future work . . . 118

List of Acronyms

AI Artificial Intelligence

DL Deep Learning

ACM Association for Computing Machinery

ML Machine Learning

CNN Convolutional Neural Network

FFNN Feed Forward Neural Network

ReLU Rectified Linear Unit

SoTA State-of-the-Art

AUPR Area Under Precision Recall Curve

AUROC Area Under Receiver Operating Characteristic

ROC Receiver Operating Characteristic

FPR False Positive Rate

TPR True Positive Rate

FPR95 False Positive Rate at 95% Recall

XAI Explainable Artificial Intelligence

GBP Guided Backpropagation

LRP Layer Relevance Propagation

CAM Class Activation Mapping

GradCAM Gradient Class Activation Mapping

GAP Global Average Pooling

LIME Local Interpretable Model-Agnostic Explanations

SLIC Simple Linear Iterative Clustering

ID In-Distribution

OOD Out-of-Distribution

MSP Maximum Softmax Probability

MLS Maximum Logit Score

MSP Maximum Softmax Probability

VIM Virtual Logit Matching

PCA Principal Component Analysis

RMD Relative Mean Absolute Difference

QCD Quartile Coefficient of Determination

CV Coefficient of Variation

List of Figures

2.1 Feed Forward Neural Network 9
2.2 Convolution example 10
2.3 CNN example 11
2.4 Binary classification confusion matrix 12
2.5 AUROC example figure. 13
2.6 Saliency Example 17
2.7 Segmentation comparison 18
2.8 Figure taken from [28], showing the steps required to create a Class Activation Map 20
2.9 CNN example 20
2.10 Mean Saliency visual explanation 24
2.11 Hypothetical ID/OOD distributions for an OOD detection metric. 27
2.12 Diagram showing the findings of [43] visually. By sampling null space noise from the network, we can create an image (the image to the right) which is completely distorted, but which is indistinguishable for the network and given the exact same prediction as if the noise was not present (the image to the left). 29

3.1 Mean Saliency visual explanation 35
3.2 Mean Saliency visual explanation 37
3.3 Spread of Saliency visual explanation 38
3.4 CIFAR10 dataset example images 48
3.5 ImageNet200 dataset example images 50
3.6 Density plots of MLS on CIFAR10 for all datasets individually and after combining Near- and Far-OOD 55

4.1 Density plot of the maximum softmax probability and maximum logit score on ImageNet200 62
4.2 Vector norm and RMD density plots for LIME on ImageNet200 63
4.3 Density plots of Norm and RMD for occlusion on ImageNet200. 65
4.4 Density plot of mean saliency for GBP on ImageNet200 68
4.5 Average AUROC scores for magnitude aggregations on ImageNet200 . . . 69
4.6 Highest AUROC score for each XAI saliency method on ImageNet200 . . . 71
4.7 Overall performance on ImageNet200 72
4.8 CIFAR10 maximum logit and softmax score distribution 73
4.9 CIFAR10 occlusion vector norm density plot 76
4.10 CIFAR10 mean and norm density plots for Integrated Gradients 78
4.11 Average AUROC scores for magnitude aggregations on CIFAR10. 80
4.12 Highest AUROC score for each XAI saliency method on CIFAR10 81
4.13 Overall performance on CIFAR10 82

4.14 Overall performance on ImageNet200 and CIFAR10 83
4.15 ImageNet200 Saliency Aggregation Bootstrap 85
4.16 CIFAR10 Saliency Aggregation Bootstrap 87
4.17 ImageNet1K Saliency Aggregation Bootstrap 89
4.18 CIFAR100 Saliency Aggregation Bootstrap 91
4.19 ImageNet200 Saliency Aggregation plus Logit Bootstrap 93
4.20 CIFAR10 Saliency Aggregation plus Logit Bootstrap 95
4.21 ImageNet200 Saliency Aggregation plus Logit Bootstrap 97
4.22 Average scores 97
4.23 CIFAR100 Saliency Aggregation plus Logit Bootstrap 99
4.24 ImageNet200 SaliencyVIM Bootstrap 102
4.25 CIFAR10 SaliencyVIM Bootstrap 104
4.26 ImageNet1K SaliencyVIM Bootstrap 106
4.27 CIFAR100 SaliencyVIM Bootstrap 108

5.1 Statistical dispersion example heatmaps 113
5.2 Saliency magnitude example heatmaps 114
5.3 LIME and Occlusion heatmap comparison 115

List of Tables

3.1 CIFAR benchmark datasets 47
3.2 ImageNet benchmark datasets 49

4.1 AUROC scores for LIME on ImageNet200. 64
4.2 AUROC scores for Occlusion on ImageNet200 66
4.3 AUROC scores for GradCAM on ImageNet200 66
4.4 AUROC scores for IntegratedGradients on ImageNet200 67
4.5 AUROC scores for GBP on ImageNet200 68
4.6 Average AUROC scores over all XAI saliency methods on ImageNet200 . . 70
4.7 AUROC scores for LIME on CIFAR10 74
4.8 AUROC scores for Occlusion on CIFAR10 75
4.9 AUROC scores for GradCAM on CIFAR10 77
4.10 AUROC scores for IntegratedGradients on CIFAR10. 77
4.11 AUROC scores for GBP on CIFAR10 79
4.12 Average AUROC scores over all XAI saliency methods on CIFAR10 80
4.13 Wilcoxon signed-rank test for salagg on ImageNet200 86
4.14 Wilcoxon signed-rank test for salagg on CIFAR10 88
4.15 Wilcoxon signed-rank test for salagg on Imagenet. 90
4.16 Wilcoxon signed-rank test for salagg on CIFAR100 92
4.17 Wilcoxon signed-rank test for salpluslogit on ImageNet200 94
4.18 T-test for Saliency Aggregation plus Logit on CIFAR10 96
4.19 Wilcoxon signed-rank test for salpluslogit on Imagenet 98
4.20 Wilcoxon signed-rank test for salpluslogit on CIFAR100 100
4.21 Wilcoxon signed-rank test for salvim on ImageNet200 103
4.22 Wilcoxon signed-rank test for salvim on CIFAR10 105
4.23 Wilcoxon signed-rank test for salvim on Imagenet. 107
4.24 Wilcoxon signed-rank test for salvim on CIFAR100 109

Preface

Generative AI has **not** been used to generate or enhance any written text contained in this thesis. However, generative AI has been used for some programmatic tasks. Services such as GPT UiO have been used for code debugging, for generating TikZ code used for creating diagrams, and for generating some boilerplate Python code. All data and personal information have been processed in accordance with the University of Oslo's regulations, and I, as the author of the document, take full responsibility for the validity of any generated code used as part of this work.

Chapter 1

Introduction

1.1 Motivation

Machine Learning (ML) generally, and Deep Learning (DL) specifically, have seen a tremendous increase in performance in recent years, performing comparable to humans in tasks such as image classification, speech and handwriting recognition, as well as many others [1]. Consequently, DL methods have been deployed in a multitude of fields and have become a part of our daily lives through their role in web search, text translation, computer vision, and in many other technologies which are taken for granted. In the medical field, deep learning has the potential to provide faster and more accurate detection of diseases by being trained on cases from thousands of previous patients [2]. Despite this, the adoption DL in high impact fields, such as medicine, has been slow, with [3] stating that: "surprisingly little in health care is driven by machine learning".

To explain this discrepancy, we should consider that, despite their impressive performance, the application of deep learning methods is not without drawbacks. Firstly, deep neural networks are inherently unexplainable due to the large number of parameters that any non-trivial network has. SoTA models will perform millions of operations to evaluate a single data point, and it is therefore impossible for humans to comprehend and explain the entire process which lead the model to make a particular decision. In medicine, this is a major limitation of deep learning methods, as both doctors and patients expect to be able to understand why a decision was made [4]. In other high-impact fields, such as autonomous driving, this lack of transparency also has serious practical and legal ramifications.

Secondly, although neural networks may attain high accuracy on test data and appear to have learned great insights about the tasks they are employed in, they often lack robustness and can suffer large drops in performance on data points which are slightly different from the training data. As [5] has shown, it is possible to create data points which are imperceptibly different from normal data points, yet still fool otherwise high performing models. More problematically, unlike humans, who recognize when they are faced with a novel situation where their expertise might be lacking, DL methods will predict equally confidently on data points which are far outside the data they have been trained on [4].

These two problems lead to the fields of XAI, and OOD detection. XAI attempts to explain the reasons why a model came to a decision, which helps to remedy the black-box nature of complicated DL models. In a healthcare setting, such explanations can be inspected by medical practitioners to confirm the diagnosis, and can be used to give patients information about why decisions regarding their health were made. In

autonomous driving or other automated high impact fields, XAI can be used to detect failure modes or to understand and improve trained models. OOD detection attempts to uncover when a data point is too different from the training data to be classified reliably. These methods could alert medical practitioners when such data points occur, thus avoiding potentially fatal misclassifications. In autonomous driving, the system could detect novel situations and cede control back to the user, avoiding accidents.

Both of these fields have seen increased interest in recent years, and are vital parts of any integration of DL in high impact settings. As two vibrant fields of study, there is great potential to combining insights from one field to improve performance in the other, an area which is underexplored. This thesis will focus on OOD detection, but will attempt to use XAI methods to improve detection performance. The overarching intuition is that by inspecting the explanation of a model on a specific data point, we may be able to uncover flaws or irregularities in the explanation which could help us determine whether the data point is OOD. This methodology is essentially non-existent in the literature: OpenOOD, the standard OOD detection benchmarking framework, which includes over 40 different methods, contains no methods which use XAI as part of their functioning.

1.2 Problem Statement

As explained in the previous segment, OOD detection is a developing field, which has become more important in recent years as machine learning is being used for higher impact tasks, such as disease detection, autonomous driving or infrastructure inspection. As reported by [6], there are to date no OOD detection methods which outperform all others on different benchmarks, which means that there is great potential for further research. Finding novel methods which improve a model's ability to detect when input is OOD is important to increase the robustness of machine learning models as they are used in real-world scenarios. The field of XAI is concerned with understanding the inner workings of a model, and could thus offer insights which can help us detect unusual behaviour in the model as a result of OOD data points. The problem statement is thus as follows:

To what degree can methods from the field of Explainable Artificial Intelligence be used to improve Out-of-Distribution Detection?

To answer this question, I introduce 3 objectives:

1. Develop one or more OOD detection frameworks which utilize XAI as part of their functionality, either by combining traditional OOD detection methods and XAI explanations, or by using XAI explanations on their own.

2. Analyse the behaviour of different XAI methods on ID and OOD data, to be able to develop effective OOD detectors under the benchmarks created as part of objective 1.

3. Perform comprehensive tests on the developed OOD detectors on SoTA OOD detection benchmarks, to be able to accurately assess the performance of XAI based OOD detection in comparison with existing OOD detection methods.

1.3 Scope

As we will see in chapter 2, both the fields of XAI and OOD detection are very large, which makes it impossible to explore all the possible ways one might combine XAI and OOD detection. Thus, it is necessary to restrict the scope of both the XAI and OOD detection methods used. In this section, I will describe the choices I've made and give a short explanation. In later chapters, the choices will be justified more thoroughly.

The field of OOD detection is primarily concerned with image classification tasks. Thus, my project will also deal exclusively with image classification datasets. Given this type of data, the choice of XAI algorithms naturally gravitates towards post-hoc, saliency based methods. The choice of OOD detection methods is not significantly restricted by the choice to deal exclusively with image data, but I will exclude methods which use outlier exposure to improve performance, or which require retraining of the model. These choices are informed by [7], which have found that outlier data is not necessary to achieve SoTA performance, and that "post-hoc methods [...] are generally no worse than methods that require training". Excluding methods which require training drastically decreases the time required for development and testing of new methods, and is thus suitable for an exploratory thesis such as this one. For image classifiers, I limit myself to Convolutional Neural Network (CNN) based models, as they are compatible with a far larger amount of XAI methods than other computer vision models, such as vision transformers.

1.4 Research Methods

Association for Computing Machinery (ACM) [8] defines three paradigms for conducting research in the field of computing: *theory*, *abstraction* and *design*. The research methods in this thesis most closely align with the abstraction paradigm. This paradigm has the following four stages:

1. Form a hypothesis
2. Construct a model and make a prediction
3. Design an experiment and collect data
4. Analyse the results

In this thesis, I hypothesize that XAI methods can enhance OOD detection algorithms. Informed by this hypothesis, I construct three XAI based OOD detection frameworks. I perform several experiments across four OOD detection benchmarks, and finally analyze the results of these experiments.

1.5 Ethical Considerations

When developing machine learning models, it is paramount to always consider the ethical implications of the work that is conducted. ML algorithms are prone to bias, have potentially large impacts on the environment when trained, and may be opaque and difficult to comprehend, weakening the rights of those that are subjected to AI decision-making. Both the field of OOD detection and XAI are attempts to remedy some of these issues, and this work could therefore have positive societal impacts. OOD detection is

concerned with alerting users when an AI model is faced with unexpected data, and may thus mitigate problems associated with biased datasets and improve the safety of AI models integrated in high-impact decision-making.

In this work, only publicly available datasets like ImageNet, CIFAR and Places365 have been utilized, minimizing any potential impact of using sensitive or private data. Furthermore, the methods developed in this thesis do not require training, reducing the environmental impact. However, extensive testing has been done on multiple datasets, over many hours of compute time, which has incurred a moderate amount of energy usage. In addition, OOD detection methods make AI models more robust in real-world use cases, which can lead to job displacement, creating detrimental societal effects that must be dealt with.

1.6 Main Contributions

In this thesis, the goal has been to investigate whether XAI methods can be used for OOD detection, and to introduce proof-of-concept OOD detection frameworks which use XAI explanations as part of their functioning. As part of this work, I have developed three frameworks for XAI based OOD detection methods, which use the saliency maps (heatmaps) generated by XAI methods applied to images: *Saliency Aggregation*, *Saliency Aggregation plus Logit* and *SaliencyVIM*. These three frameworks have been thoroughly tested on all four OOD detection benchmarks included in OpenOOD [6, 7], the de-facto standard OOD detection benchmarking tool.

Through the testing of these three methods, I show that XAI methods can indeed be used for OOD detection, contrary to the results found by previous research [9]. The key takeaways of this thesis are as follows:

- XAI saliency mapping methods generate values which can be used to predict OOD samples comparable to baseline methods. However, most XAI methods output normalized saliencies, which removes valuable information necessary to perform OOD detection. By removing this normalization and using raw saliency values, I show substantial improvements over previous methods [9].

- A simple aggregation of raw saliency values performs slightly below the baseline methods of Maximum Logit Score (MLS) and Maximum Softmax Probability (MSP) on Near-OOD datasets, and sometimes outperforms the baselines on Far-OOD datasets. This shows that XAI saliency maps, on their own, capture enough information about a network's response to a data sample to effectively discriminate between ID and OOD data points.

- Depending on the choice of XAI method, XAI saliency maps output values which are relatively uncorrelated with traditional baseline OOD methods. This means that their outputs can be combined with other OOD detection methods to increase OOD detection performance. Inspired by the work of [10], I combine XAI saliency aggregates with MLS by a simple addition of Z-scores. Under this framework, I find that the performance of XAI based OOD detection is increased by several percentage points over the baselines. In addition, on three out of four benchmarks, one method under this benchmark performs quite close to the SoTA amongst OOD detection models which do not retrain the underlying classifier.

- Saliency values can also be appended directly to the model logits in OOD detection methods such as Virtual Logit Matching (VIM) [11], leading to statistically

significant improvements over using the method without XAI saliency values across several benchmarks.

These three frameworks are highly general and enable further research into the integration of XAI and OOD detection. The performance attained methods under these frameworks show immense potential, and comes close SoTA methods in several instances.

In addition, I have had code merged into the codebase for OpenOOD, the definitive OOD detection framework and benchmarking tool. By improving the code used when benchmarking new OOD detection methods, I have contributed to the broader field of OOD detection.

1.7 Thesis Outline

Chapter 2 gives a short introduction to machine learning, followed by a deeper look at the fields of XAI and OOD detection. Chapter 3 introduces my methodology; the methods I have used to compare and contrast XAI explanations on ID and OOD data, as well as the three general OOD detection frameworks which integrate explanations into their functioning. Furthermore, I introduce the different benchmarks which have been used to test each method, the statistical methods used to ensure that the results are statistically significant and the software and hardware architecture used. Chapter 4 first goes into the results of the comparison between ID and OOD explanations, detailing the differences between different XAI methods. These investigations have been done on validation benchmarks which are not used during testing, to ensure unbiased results. Based on these results, I have chosen a selection of XAI OOD detection methods under the developed frameworks. These frameworks are then tested on bootstrapped testing benchmarks and compared against baseline methods using statistical analysis. After the experiments follow a discussion (chapter 5), where I discuss the results in relation to the problem statement, and conduct deeper analyses on the overall performance of the developed methods. Finally, the conclusion (chapter 6) gives a short conclusion of my findings, and envisions a way forward for future work.

Chapter 2

Background

In this chapter, I first give a short introduction to important concepts in the field of machine learning generally, followed by a more in-depth look at the fields of OOD detection and XAI. Finally, I give an overview of related works; papers which have attempted to use XAI for OOD detection.

2.1 Machine Learning

Machine Learning is the field of algorithms that are able to learn from data, as opposed to being explicitly programmed. Such algorithms use statistical methods to learn relationships in data, and use these relationships to generalize to unseen data. More formally, [12] gives the following definition of machine learning algorithms:

Definition. A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E.

Thus, machine learning is a different paradigm from traditional problem solving, where programs are made to solve problems by following explicit rules. For example, a traditional image classification system attempting to differentiate between malignant and benign tumors might use hand-crafted rules which consider the texture, color and size of a tumor, developed by medical professional with years of experience. As one might imagine, such rules will quickly become very complicated when we consider all the possible factors which might influence the appearance of a tumor. Using a machine learning approach, we would instead feed an algorithm with thousands of images of both benign and malignant tumors, and the rules could then be automatically updated until the algorithm predicted the correct category with a high enough accuracy.

Machine Learning is commonly divided into the three subcategories of supervised, unsupervised and reinforcement learning

2.1.1 Supervised Learning

Supervised Learning is a subcategory of machine learning where we have a dataset containing both inputs and desired outputs. In the example above, we could use supervised learning by creating a dataset of images of tumors (the input) and corresponding labels which indicate whether each tumor is malignant or benign (the desired output). The learning goal of the algorithm is then to associate images with the correct label. Because we know the correct answer, we are able to fine-tune the

algorithm automatically whenever it makes a mistake. However, supervised learning requires labeled data, which can be very costly, especially in the medical domain, where deciding whether a tumor is malignant or benign requires expert knowledge.

2.1.2 Unsupervised Learning

In unsupervised learning, we do not have any labels. In these cases, we might not know whether data points belong to different classes or not. Instead, we can use machine learning to uncover patterns in the data, for example by attempting to cluster the data into different groups and seeing if these groups are sufficiently separated. An example use case could be for fraud detection in a bank. By feeding financial transaction from many different users into an unsupervised learning model and asking it to perform clustering of the data, it might be possible to find a group of users whose transactions differ substantially from the rest, which might indicate that their transactions are fraudulent.

2.1.3 Reinforcement Learning

Reinforcement Learning deals with problems where we do not know exactly what the correct solution is, but we are able to assess whether a given solution is good or not. For example, when controlling a robot arm, it is difficult to say exactly what angles each joint should be for every millisecond when picking up an object, but if the arm does not pick up the object, we know the algorithm has failed. In these problems, the algorithm is trained through reinforcement, where good attempts are rewarded and bad attempts are punished.

2.2 Neural Networks

Neural Networks constitute a class of machine learning algorithms which have become the clear SoTA in almost all fields where machine learning is applied. Notable examples are computer vision, image classification, speech recognition, text and image generation and machine translation. Neural networks are loosely inspired by our own brains, where neurons are connected together and send information between each other. By connecting thousands of neurons together, neural networks are able to learn complicated relationships between the input and output.

2.2.1 Feed Forward Neural Networks

The Feed Forward Neural Network (FFNN), also known as a Multilayer Perceptron, traces its roots to the very beginning of machine learning, through the work of Frank Rosenblatt [13]. It forms the basic structure for neural networks which has been adapted and modified over the years to form more complex architectures such as convolutional, recurrent or residual neural networks. In an FFNN the input values are passed through an affine transformation (a matrix multiplication followed by the addition of a bias), and then passed through an activation function. The output of this activation function can then go through the same process again, which constitutes a single "layer". By stacking several of these layers, with non-linear activation functions, an FFNN is able to learn

arbitrarily complex mappings between inputs and outputs[1]. Figure 2.1[2] shows a simple FFNN architecture with three hidden layers. In this case, the bias has been omitted for brevity. Here, we can see how all nodes of a layer are connected to the following layer. By using an activation function on the nodes of the hidden layer, before their values are sent to the next layer, we achieve the non-linearity required to learn complex patterns.

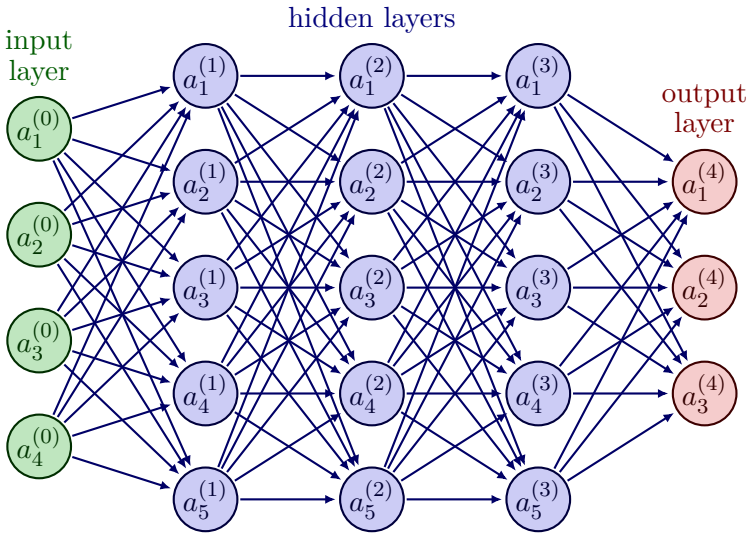

Figure 2.1: Figure showing a simple Feed Forward Neural Network, with nodes labeled. The number in parentheses indicates the layer number while the subscript indicates the node number within the layer.

Mathematically, a single layer can then be described as follows:

$$\boldsymbol{x_{i+1}} = \sigma_i(A_i\boldsymbol{x_i} + \boldsymbol{b_i}) \tag{2.1}$$

Here, the input $\boldsymbol{x_i}$ is linearly transformed by the weights of the matrix A_i from the input space to the output space, then each value of the new vector in the output space is adjusted by an addition of a bias term, and finally an activation function (σ_i) is applied to each value. The size of the input and output layers is determined by the number of input and output features, respectively. The activation function of the output layer is determined by the application, most commonly *sigmoid* for binary classification, *softmax* for multi-class classification and simply identity for regression.

2.2.2 Convolutional Neural Networks

FFNNs have some inherent flaws which make them unsuitable for working with high dimensional, spatially connected data, such as the pixels which make up an image. Firstly, each input of a FFNN is connected to every output of the following layer. If we want to connect the input pixels of a 224 by 224 image to a layer of 100 nodes, our first layer will have over 5 million weights, which is already quite a lot for a relatively small image. Furthermore, these weights will have to encode redundant information, because

[1] In fact, by the Universal Approximation Theorem [14], only a single hidden layer between the input and output is necessary, although this theorem does not give a way to construct such a network for any given function

[2] Figure by Izaak Neutelings, "Neural Network with coefficients, arrows", TikZ.net, licensed under CC BY-SA 4.0, [https://tikz.net/neural_networks/].

each pixel is considered separately. Consider a network attempting to detect the presence of a cat in an image: We would want the network to detect the cat regardless of whether it is in the middle, the right corner, or any other position in the image. In an FFNN, the weights connected to any of these positions in the image would then have to encode a cat detector separately from all the others.

CNNs solve both these issues by using small kernels of weights which are "slid" across the entire input. By using the same weights across all positions of the image, we do not need to train separate detectors for different positions, giving us translation invariance. Figure 2.2 shows the functioning of a convolutional kernel on a 3-channel image. Each value in the output is a weighted sum of a neighbourhood of values in the input image, where the weights are defined by the kernel. As we can see, the same weights are used on all positions, drastically reducing the number of parameters that need to be tuned. In a 2d-convolution, the kernel has the same number of channels as the input, and is only slid across the height and width dimension. The kernel in this figure is a *Sobel Operator*, and detects vertical edges. In a CNN, the weights of each kernel are not specified manually, but rather learned through backpropagation.

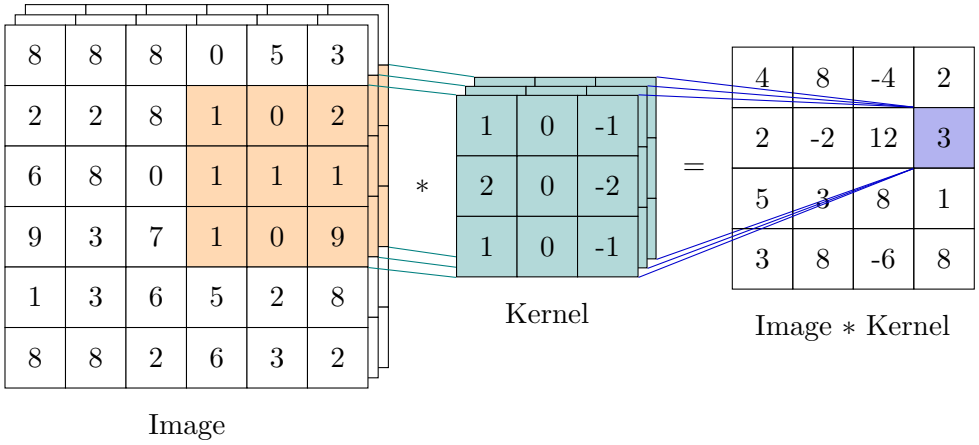

Figure 2.2: Figure showing a convolutional kernel applied to a 3-channel image.

By using several different kernels, we can detect many different patterns despite each kernel only detecting a single type. By using the outputs of all the kernels as inputs to a new set of kernels, we can use the same type layer structure as in an FFNN, allowing us to extract information in a hierarchical manner. It is common to see that trained CNNs have early layers that detect edges and texture, later layers that use these edge and pattern detections to detect larger shapes, while the final layers combine the shapes to detect entire objects [15].

By not evaluating every possible position in the input image, CNNs downsample the image, and are able to reduce the number of operations considerably. Simultaneously, this downsampling enables each subsequent layer to consider a larger area of the input image than the previous (a larger field-of-view), which allows larger patterns to be discovered. Simultaneously, it is common to use a larger and larger amounts of kernels on the new input, thus increasing the channel depth while the spatial dimensions are reduced. Between each layer, we use non-linear activation functions, similarly to how the are used in FFNNs.

After several such convolutions, we can flatten the output, either by aggregating each

channel using Global Average Pooling (GAP) or a similar method, or we can simply flatten all dimensions and consider the three dimensional feature map as a long vector of shape $C \times H \times W$. By doing this, we can pass the output to one or more linear layers, which can perform classification or regression on the extracted features and give us a final prediction. Figure 2.3 shows a high level overview of this process. Here, the input, which has only 3 channels, has its spatial dimensions reduced while its channel depth increases through consecutive convolutions. Finally, we have a certain number of channels in our final feature map, which are flattened (in this case with GAP) and processed through a linear layer to give a final prediction.

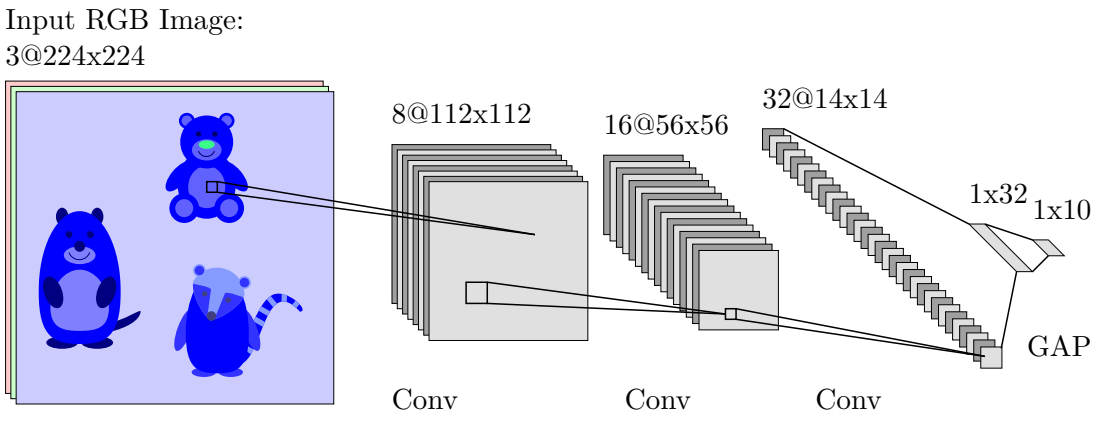

Figure 2.3: Figure showing a high level overview of how a CNN functions

2.3 Model evaluation

OOD detection is essentially a binary classification problem. Thus, the metrics I will use in this thesis are those used for such problems. In the field of OOD detection, AUROC and False Positive Rate (FPR) are most commonly used. As such I shall focus on these metrics, as opposed to for example the Area Under Precision Recall Curve (AUPR).

2.3.1 Accuracy

Accuracy is the simplest metric used in binary classification. It is simply the ratio of correct predictions over all instances in the data set.

Accuracy has the advantage of being simple to understand and calculate, but it is very often insufficient. A simple (and quite common) scenario where accuracy fails to capture the performance of a model is any situation where there are large class imbalances. For example, imagine we have trained a CNN to predict whether a person has lung cancer or not, based on CT-scans of their lungs. As most people do not have lung cancer, we can imagine that such a dataset is highly imbalanced, for example that only 1 in 100 people actually have cancer. If a model simply predicts that no one ever has lung cancer, it will be correct in 99% of cases, and will thus have an accuracy of 99%, although it has missed every instance of cancer and is completely unusable in any real context.

For the purposes of OOD detection, accuracy is thus insufficient, as we have no guarantees that ID and OOD will be balanced. In fact, we expect OOD data to be relatively rare, given that the goal of developing an AI model is to ensure high performance during deployment, which necessitates having training data which covers as much as possible of the data seen during inference.

2.3.2 Metrics utilizing the binary classification confusion matrix

Instead of simply considering whether a prediction was correct or not, we should take into account the different combinations of prediction and ground truth, considering positive and negative classes separately. Figure 2.4 shows the possible four possible combinations given a ground truth class and a predicted class.

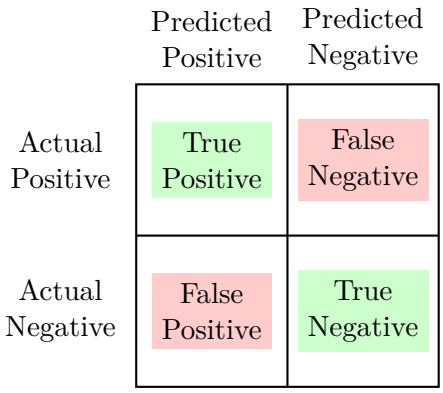

Figure 2.4: Figure showing the binary classification confusion matrix, denoting the four possible combinations created by the ground truth and predicted class. Green cells denote correct predictions, while red cells denote wrong predictions.

With these possibilities defined, we can begin to gain a clearer picture of the performance of a model.

Precision and Recall

Precision is the share of positive predictions that were actually positive. With a high false positive rate, we have a low precision, which means that the model is erroneously flagging many negative classes as positive. In an OOD detection setting, a model which flags many ID samples as OOD would have a low precision, if we treat OOD samples as the positive class.

Recall is the share of actual positive samples that were predicted positive. Recall tells us how many of positive samples we missed. In and OOD detection context, a model which lets many OOD samples slip by undetected will have a low recall score.

Precision and recall are often used together because evaluate the model in different ways that complement each other. If a model has both high precision and high recall, it does not erroneously flag many negative classes as positive, nor does it miss many positive classes.

Sensitivity and Specificity

Sensitivity and specificity is another pair of metrics that is commonly used for evaluating binary classification. Sensitivity is equivalent to recall; the share of positive samples that were correctly predicted as positive. Specificity is the share of the negative samples that were correctly predicted as negative. Sensitivity and specificity are also known as True Positive Rate (TPR) and *True Negative Rate*.

2.3.3 Threshold Independent Metrics

The previous metrics are a clear improvement over simply using accuracy. However, they still have the problem that they are all dependent on what threshold one sets when predicting something to be a negative or positive class. Thus, it becomes harder to compare different models by using these metrics. Indeed, by simply increasing the threshold of any classifier, we can increase the true negative rate. Similarly, by decreasing the threshold, we can increase the true positive rate.

Area under Receiver Operating Characteristic

AUROC remedies this problem by looking at all possible thresholds, and calculating the TPR (equivalent to sensitivity, recall), and the FPR (equivalent to 1− specificity) for each possible threshold. With these values calculated, we can plot each point on a graph, giving us an Receiver Operating Characteristic (ROC) plot. Figure 2.5 shows this plot, for three different models.

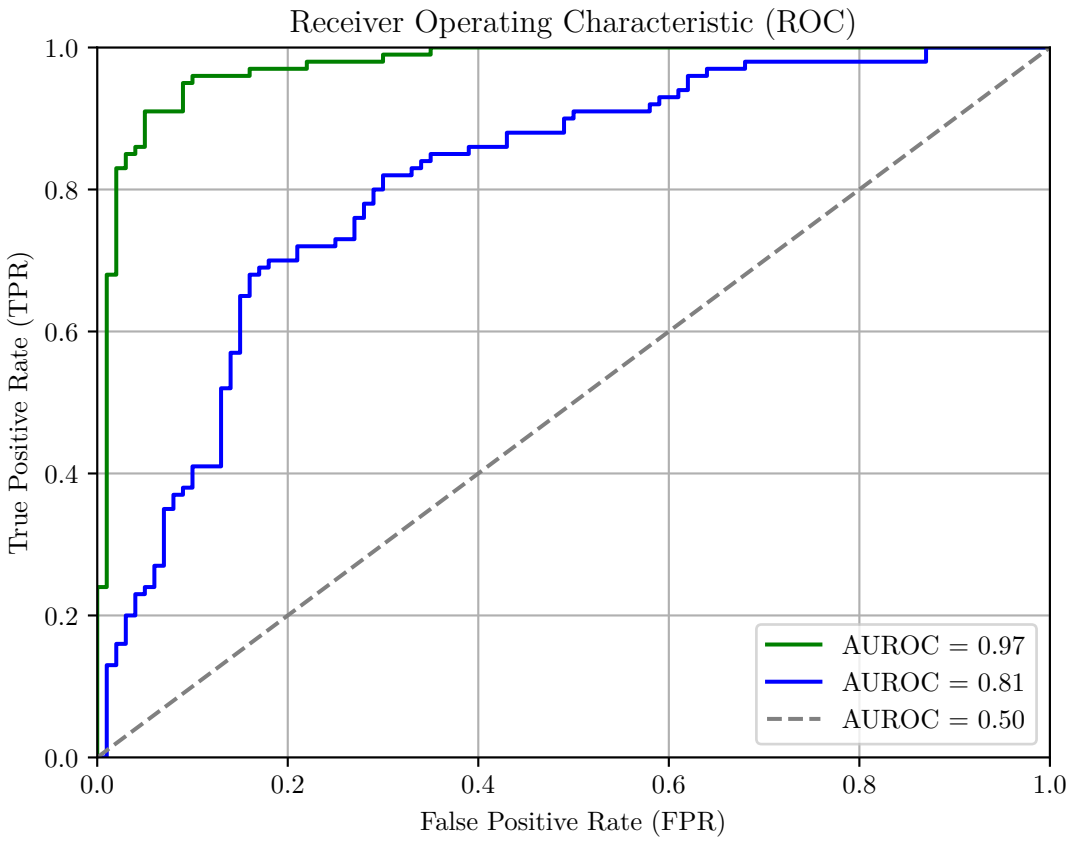

Figure 2.5: Figure showing the AUROC curve for two imagined classifiers; one which has an AUROC of 0.97 and one which has an AUROC of 0.80. In addition, the AUROC curve of pure guessing is shown in gray.

Once we have done this, we can calculate the integral under this curve, giving us the AUROC. If a binary classification model can perfectly separate the two classes, then

all possible thresholds will either have 100% TPR or 0% FPR, giving an area under the curve of 1. If instead a model has no discriminative power, then the predicted values of positive and negative classes are entirely random, and all changes to the threshold will increase one of the metrics at the expense of the other. Such a model would have TPRs and FPRs making a straight line of points, and an AUROC of 0.5. In between these extremes, we can evaluate different models, without having to consider different thresholds. Figure 2.5 shows what the ROC-curve of a model with either 0.97 or 0.80 AUROC looks like, as well as that of a random classifier with AUROC = 0.50.

One important thing to note about the AUROC is that values lower than 0.50 do not mean that a model is worse than random guessing. This is because if a model is consistently wrong, we can simply choose the opposite category of what the model outputs, and gain a new model which is better than random guessing. For example, if a cat versus dog detector gave an actual image dog a higher chance of being a cat than an actual image of a cat in 95% of cases, it would have an AUROC of only 0.05. However, if we simply multiplied all outputs of the model by -1, we would suddenly have a model which correctly gives a cat a higher chance of being a cat in 95% of the cases, and an AUROC of 0.95. Thus, we really only care about getting AUROC scores far away from 0.50.

False Positive Rate at 95% Recall

False Positive Rate at 95% Recall (FPR95) is another way of comparing models without having to consider specific thresholds. Instead, we simply select the threshold which gives a recall (equivalent to TPR) of 0.95, and calculate the FPR at this threshold. The drawback to this metric as opposed to AUROC is that we do not get a general view of how the model performs. However, if we have a requirement that the model has a very high true positive rate, we may not care about how the model performs at any other threshold, and thus this metric is suitable. It is of course also possible to calculate this metric at any other recall value, depending on the application. However, in the field of OOD detection, FPR95 is the metric that is used in the vast majority of cases [7, 11, 16–18].

2.4 Explainable Artificial Intelligence

Below follows a thorough introduction to XAI, as well as detailed look at some important methods for explainability for neural networks applied to images. Specifically, saliency methods will be explained in detail, as they constitute a core part of my thesis.

2.4.1 The motivation for Explainable Artificial Intelligence

Given the impressive performance of DL methods, one might be convinced that these models do not need to be explainable or interpretable, and that we instead should just place our faith in the model without knowing exactly how it came to a decision. However, as [19] points out, "a single metric, such as classification accuracy, is an incomplete description of most real-world tasks". Small differences between the data distribution when the test data was collected and when the model is deployed may have a large impact on the model's performance, or the model may have learned artifacts or specificities in the training dataset which were also present in the test dataset, leading to a false belief that the model has gained generalizable knowledge when it has not. By using explainable

methods, we may reveal these shortcomings. Relevant to this thesis, this may also have the secondary effect of separating ID and OOD data points.

XAI is also especially important whenever the model is used in settings where its decisions have a high impact. If a model is used by a hospital for disease detection, both the patient and doctor will probably want to be able to understand why the model has found that a disease is present. For them, high performance on a test set of different cases may not be enough. As [2] states, "for the regulated healthcare domain, it is utmost important to comprehend, justify, and explain the AI model predictions for a wider adoption of automated diagnosis". In other high impact areas, such as autonomous driving, the impact of wrong decisions by the network can have fatal consequences, and customers and regulators will want to be absolutely sure that the models used are robust and base their decisions on relevant factors as opposed to quirks in the training data. Furthermore, the right to an explanation of an automated decision affecting a person is included in the EU's General Data Protection Regulation, which states that "In any case, such processing should be subject to suitable safeguards, which should include specific information to the data subject and the right [...] to obtain an explanation of the decision reached after such assessment and to challenge the decision." [20].

2.4.2 Taxonomy of Explainable Artificial Intelligence

This section goes through three axes which define an XAI method:

- Intrinsically explainable models versus post hoc methods
- Model dependent versus model agnostic methods
- Global versus local explanations

Intrinsically explainable models versus post hoc methods

Intrinsically explainable models are models which have sufficiently low complexity, such that it is feasible for a human to understand them without further modifications. Examples of such methods are linear regression, logistic regression and decision trees [21].

Post hoc methods are methods which are applied to the model after training. These methods do not aim to constrain the model to be interpretable, but inspect the model after training. For example, after using a convolutional neural network to classify a CT-scan of a tumour (which gave a prediction of malignant), we could run post hoc algorithms on the network which are able to extract which part of the image contributed the most to the prediction. Thus, post hoc methods remove the need for the model to be simple enough for a human to understand by extracting the relevant information for us.

Model dependent versus model agnostic methods

Model dependence/agnosticity denotes whether an XAI method uses specifics of a particular type of model to generate the explanation, or whether the method can generate an explanation without using specifics of the model at all. Explanations based intrinsically explainable models are clearly model dependent, while methods that only use the input and output of the model instead of looking at the internal operations are model agnostic. An example of a model dependent method (which is not simply

an intrinsically explainable model) is Class Activation Mapping, which requires a CNN with a specific architecture to function, while an example of a model agnostic method is Shapley [22], which treats the underlying model as a black box and uses the inputs and outputs to calculate the marginal effect of a single feature on the output value.

Global versus local explanations

Global explanations provide general relationships between the input features and outputs learned by the model over the entire dataset [23]. In this way, they can show how a specific feature affects the output in general, instead of just how it affects the output of a single point. These methods are ideal for finding trends in the data, but may not be suitable for a patient wanting an explanation for their specific case.

Local explanations do not describe general trends, but focus only on a single data point. These methods give insight into how the features influenced the prediction of a single data point, but these relationships may not hold for other data points, and as such these methods do not give the same insight into the general behaviour of the model.

2.4.3 XAI methods adapted to images: Saliency maps and segmentation

As explained in section 1.3, the field of OOD detection is primarily focused on image data, and as such this is the focus of this thesis as well. Thus, before delving into specific XAI methods, it is beneficial to elaborate on how XAI methods are adapted to images. When explaining tabular data made up of categorical and numerical values, it is often common to explain each feature by associating it with some change in the output prediction. For example, one might say that increasing the number of rooms in a house by one increases the predicted sale price by 30 000 NOK, or that the absence of a balcony decreases it by 25 000 NOK. But, given that images are made up of tens or hundreds of thousands of features (RGB pixel values), such an approach may not be suitable.

Firstly, given that we have so many features, it may no longer be interesting to know exactly how much each feature contributes to a prediction, given that we don't expect any single pixel to have a very large impact. Furthermore, inspecting the contribution of each individual pixel, like one might do for tabular data, is not even realistically feasible, due the overwhelming number of features. Given that our input is in the form of an image, which is best understood visually as opposed to numerically, it is then natural to instead present the explanation in the same way.

These visual explanations take the form of saliency maps, as shown in figure 2.6. Here, we display the saliency values of all pixels in a heatmap. Instead of considering the absolute values, we instead display the saliencies in relation to each other, such that the most important and least important pixel has colors on opposite sides of the colormap. In this way, it is easy and intuitive to see where the important regions of the image are, and we do not need to consider the absolute values of each pixel.

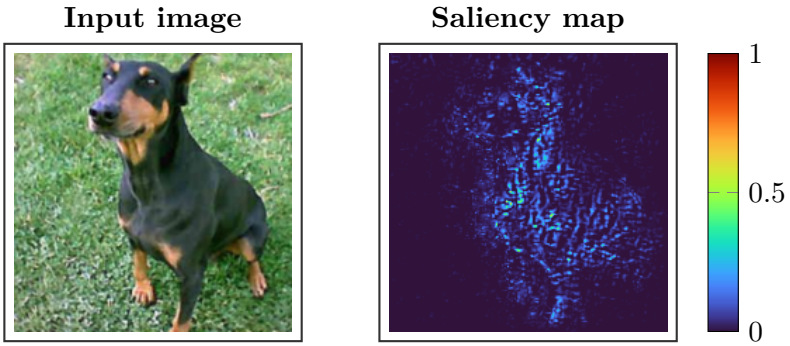

Figure 2.6: Figure showing an input image and the corresponding saliency map, generated by an XAI algorithm which shows the important pixels for the predicted class (Rhodesian Ridgeback). The colorbar and heatmap show the relative saliency for all pixels, with 0 being the least important pixel and 1 being the most important pixel

Secondly, many methods which are designed for data tabular data (which have may have 10-20 features) are far too slow when the number of features may be over a hundred thousand. Methods such as LIME, Occlusion, or Shapley fall into this category, as they permute the input at a rate which scales with the number of inputs.

A solution to this is to segment the image into larger regions, which are treated as one. As opposed to asking "how does the prediction change if we change this pixel", we instead ask "how does the prediction change if we change this *region*". All pixels in a region are then awarded the same amount of importance. This drastically reduces the dimensionality of the problem, and allows us to use a much larger range of methods. The output of such methods is thus also a saliency map, but one where each pixel is not given an individual saliency score, but rather a score derived from the larger region in which it is contained.

Regions of pixels can be created in different ways. The simplest option is consider a window of a specific height and width, and slide this window over the image with a specific stride. By adjusting the stride and size, the number of dimensions can be adjusted. The benefit of such a simple approach is that the segmentation itself introduces no extra computation. However, a substantial downside is that each region can contain completely unrelated objects, because the regions are created without considering the underlying image. For example, if a 60 by 60 pixel region contains in its lower right corner a part of a dog, occluding this region could lead to a large change in confidence and thus high importance. However, it is not just the lower right corner that is awarded high importance, but the entire region, which may contain other objects or parts of the background which are not actually important.

By using more sophisticated segmentation methods, we can avoid this problem. Methods such as Simple Linear Iterative Clustering (SLIC) [24] create regions which can be considered more intuitive than simply using a rectangular sliding window. SLIC performs an iterative clustering of pixels, grouping similar pixels into larger regions called superpixels. The downside to this method is that the iterative, CPU-bound process introduces a considerable computational overhead. Figure 2.7 shows a comparison between the saliency map of the LIME algorithm applied to an image segmented using rectangular regions and to an image segmented using SLIC. As we can see, the rectangular segmentation means that the some of the grass in the background is given a high saliency, because it happens to be in the same region as the dog's head. Using

SLIC, there are few regions which contain both grass and the dog, and thus the high saliency of the dog's face does not affect other parts of the image.

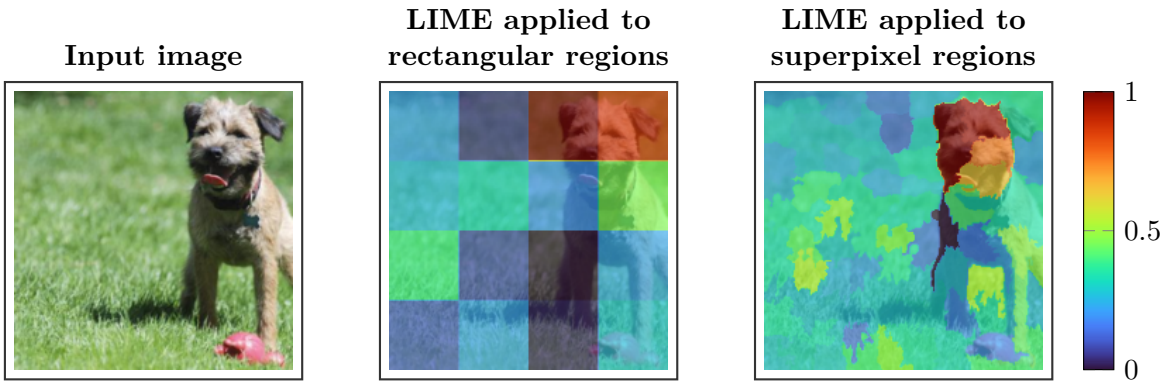

Figure 2.7: Figure showing a comparison between generating saliency maps using the LIME algorithm on rectangular and superpixel regions. The colorbar and heatmap show the relative saliency for all regions, with 0 being the least important region and 1 being the most important region

2.4.4 Specific methods

The following section goes through several specific XAI methods. First, the methods which are integral to the methods introduced in chapter 3 are described in detail, so that we have the necessary information to understand and analyze the behaviour of the methods developed as part of this thesis. Following this, I describe a selection of other XAI methods, to give a broader overview of the field.

Local Interpretable Model-Agnostic Explanations (LIME)

LIME [25] is a post-hoc, model independent XAI method. The method is built on the idea that while the decision function of a large neural network (or any other large model) might be far too complex to easily interpret, it can most likely be approximated quite well by a simpler function, as long as we only look at the feature space around a single data point. For example, we could approximate a large feed forward neural network with a simple linear regression model, which can be intrinsically explained due to its low complexity.

To create a locally interpretable model, we need a neighbourhood of data points around our point of interest. To do this, we can sample a number of points from our dataset and weigh them by their distance to our original point. This sampling can be done in many ways, for example by calculating a mean and variance for each feature and sampling from a normal distribution. For image data, we can create new points similar to the image by masking out different regions of the image [21]. The distance measure depends on the type of data we are dealing with. Regardless, the distance values are passed through a smoothing kernel which can be tuned to adjust the size of the "neighbourhood".

With these new data points, we can generate new predictions using the original, complex model. Thus, we now have a series of points, each with a weighting based on their distance to our original point, and each with a predicted score from our original model. With such a dataset, we can train a simpler model, which will then approximate

the complex model around the point of interest. By inspecting this simple model (for example: the learned weights of a linear regression model, or the structure of a decision tree), we can learn approximately how the complex model functions in a region around this single data point.

Occlusion methods

Occlusion methods are a family of post-hoc, model independent XAI methods. They function by masking different parts of the image and inspecting the change in output score. If an area leads to a large drop in softmax score for the predicted class when masked, this area must have been important for the network when making the prediction. The mask can be as simple as replacing all masked pixels with a single color, such as gray [26], or they could use more advanced inpainting methods using generative models, for example by replacing a masked tumor with generated healthy tissue.

Regardless of the mask, one can easily calculate the importance of any pixel for a prediction by calculating the average change in the output score for all masks which contain the specific pixel [27]. Occlusion methods have the advantage of being completely model independent, since they do not consider the internals of the model. However, the computation can be expensive, because we need to run a forward pass for each position of the mask on the image.

Class Activation Mapping (CAM)

Class Activation Mapping (CAM) [28] is a model dependent, post hoc XAI method, which is used on Convolutional Neural Nets (CNNs). For a specific output node of a model (for example, the one denoting the presence of a specific class, such as "cat"), CAM outputs a heat map showing which areas of the input image contributed to this node. In this way, CAM gives a visual explanation to which parts of an image the model focused on when making a decision to classify an image to a specific class. This method is model dependent, because it requires a specific architecture in the final layers of the network to work.

CAM is a relatively simple method to understand. It exploits the fact that various convolutional layers of CNNs actually behave as object detectors, even when the training objective is classification [28]. As [15] explains, the earlier layers "extract elementary visual features such as oriented edges, end-points [or] corners", which can be used by subsequent layers to detect higher-order features. In this manner, the final convolutional layer will detect very high level visual features, combining the extracted information from all the previous layers. This layer is composed of several feature maps, where each map can be thought of as denoting the presence of some specific feature across the original image. The authors perform global average pooling (GAP) on these feature maps, giving a single value for each map, which is followed by a single dense layer and the Softmax activation function. In this way, each output node in the final layer is a weighted sum of all the global average pooled feature maps from the final convolutional layer. This means that we can represent the areas of the image which were used to perform the classification by performing the same weighted sum on the actual feature maps instead, which gives us a heat map which we can overlay on the original image (after upsampling the feature maps).

Figure 2.8 shows the process visually. From this we can see that the resulting Class Activation Map (bottom right) gives an intuitive explanation for why the image in the top left gives a high score for the presence of the class "Australian Terrier".

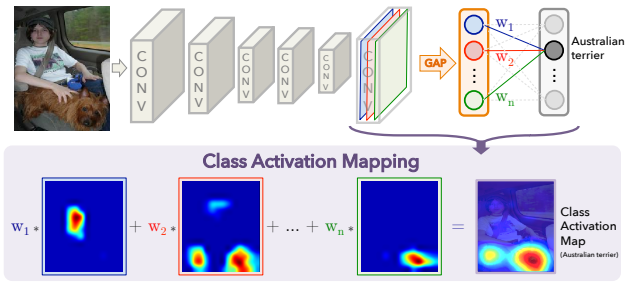

Figure 2.8: Figure taken from [28], showing the steps required to create a Class Activation Map

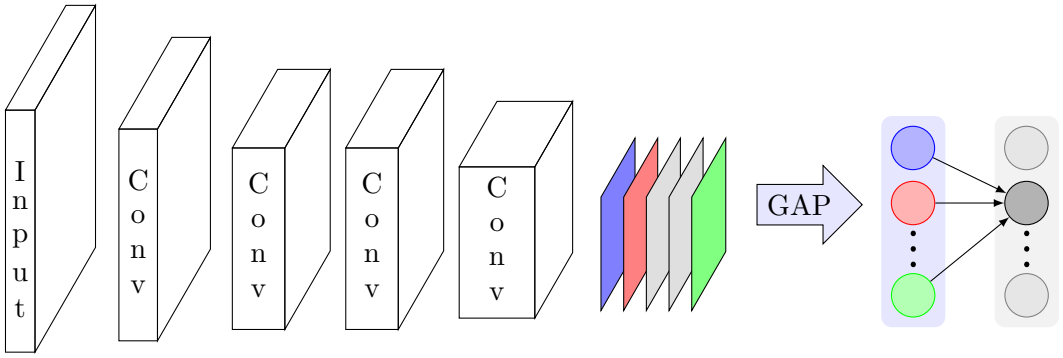

Figure 2.9: Figure showing a high level overview of how a CNN functions

Although CAM is an intuitive and effective method of visualizing the inner workings of a CNN, it has some downsides. Firstly, it is highly model dependent, requiring that the model only have a single dense layer after the convolutions. Although there are some SoTA models which only use a single dense layer, this still places a limit on what models can be used, or requires the simplification of models that use more than a single dense layer. [28, p. 4] notes a 1-2% drop in classification performance when performing this simplification. Secondly, the output of CAM is simply a weighted sum of all the feature maps after the final convolutional layer. As we move deeper in a CNN, we reduce the spatial resolution by downsampling, while increasing the number of channels (increasing the depth of the output while reducing the height and width). Because of this, the CAM will have a drastically lower resolution than the original image, often less than 10×10, while the input image may be hundreds of pixels in both dimensions. Because of this, CAM can only show general areas, as opposed to pixel wise explanations.

Gradient Class Activation Mapping (GradCAM)

GradCAM [29] is an improvement on CAM, which generalizes the method to function with any CNN architecture, thus making the method much less model dependent and avoiding the performance drop incurred when simplifying the model with CAM. Instead of using the weights of a final layer to calculate a weighted sum of feature maps in the last convolutional layer, GradCAM uses gradients flowing from the relevant output node to the activation maps to calculate the weights for each feature map. Furthermore, the authors prove that this method is a strict generalization of CAM [29, p. 5], so that no information is lost by using gradients instead of weights.

Like the simplicity of the CAM method, the calculation of the weights using the gradients is also quite simple, as seen in Equation 2.2.

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\delta y^c}{\delta A_{ij}^k} \tag{2.2}$$

Here, c represents the index of the class we are interested in, k the index of the feature map, and i and j the width and height of the image. y^c is the element of the output vector y which corresponds to the class c, while A^k is the k'th feature map. Z is equal to the number of elements in each channel of the feature map, and simply normalizes the sum. Thus, we are actually just performing global average pooling of the gradients of A^k with respect to y^c, which gives us a single value we can use as the weight for this feature map. Doing this for all feature maps for a specific class gives us all the weights we need to calculate a weighted sum, which we can upsample and visualize to get an explanation for the decision of the CNN.

Thus, GradCAM improves upon CAM by making the method less model dependent. However, the explanations are still the same low resolution, which may not be ideal in all cases.

Guided Backpropagation

GBP [30] is another XAI method which utilizes the gradients of the network to calculate saliencies. In this case, we do not stop at the final feature map, but backpropagate through the entire network to the input image. Simply backpropagating in this way produces a saliency map for the entire input image. However, [30] finds that by only backpropagating positive gradients through ReLU functions, they are able to produce "sharper visualizations of descriptive image regions than the previously known methods". Although the choice to simply neglect negative gradients when backpropagating lacks theoretical justification, GBP has been shown to be accurate and trustworthy compared to other methods [31, 32]. Compared to the previous methods, GBP differs in the sense that it produces a saliency value for every single input feature (every pixel, for example) as opposed to regions.

Integrated Gradients

Integrated Gradients [33] is another gradient based XAI method. The method was developed as part of an effort to create an XAI method which satisfies two axioms; sensitivity, and implementation invariance. Sensitivity states that if an input and a baseline differ in only one input feature, and have different outputs, then this input feature should have non-zero saliency. [33] shows that gradients violate this property because "the prediction function may flatten at the input and thus have zero gradient despite the function value at the input being different from that at the baseline". Implementation invariance states that if two models are functionally identical (their outputs are equal for all inputs), then their attributions should be identical as well. Gradients satisfy this property, but methods which modify the gradients, such as GBP and Layer Relevance Propagation (LRP) [34], break this axiom.

Both these axioms represent desirable qualities for an XAI method; we don't want our attribution method to miss any features which lead to a change in the model prediction, and we don't want the internal structure of the model to affect the attribution if it does not affect the output score in any way. Thus, it is problematic that there are few

methods which satisfy both criteria. [33] finds that by integrating the gradients of the network in the straight line path between a baseline and the input, both these axioms are satisfied. Mathematically, the saliency for a specific input feature $\mathbf{x}_i$ is defined as follows, given an input $\mathbf{x}$, a baseline $\mathbf{x}'$ and a deep learning model f:

$$\text{IntegratedGradients}_i(\mathbf{x}) := (\mathbf{x}_i - \mathbf{x}'_i) \times \int_{\alpha=0}^{1} \frac{\delta f(\mathbf{x}' + \alpha \times (\mathbf{x} - \mathbf{x}'))}{\delta \mathbf{x}_i} d\alpha \tag{2.3}$$

By cumulating the gradients of the network at all points between the baseline and the input, [33] manages to combine the implementation invariance of gradients with the sensitivity of techniques like LRP. In addition, they prove that if the model f is differentiable "almost everywhere", then the sum of all attributions is equal to the difference in output between the baseline and the input:

$$\sum_{i=1}^{n} \text{IntegratedGradients}_i(\mathbf{x}) = f(\mathbf{x}) - f(\mathbf{x}') \tag{2.4}$$

If we then choose a baseline which has $f(\mathbf{x}') \approx 0$, we see that integrated gradients is equivalent to distributing the output prediction amongst the input features, a very desirable interpretation for an XAI attribution method.

Other methods

A part from the methods mentioned above, there are many different XAI methods. Shapley values [22] is a post-hoc model agnostic method which calculates the marginal contribution of each feature for the prediction by perturbing the input. GradCAMPlusPlus [35], HiResCAM [36] and Guided GradCAM [29] are post-hoc model-dependent XAI methods which all build on the GradCAM architecture and modify the calculation of saliencies in ways that can be beneficial in specific scenarios. Partial Dependence plots [37] and Accumulated Local Effects plots [38] are global, model agnostic methods which aggregate information over the entire dataset to give information about the relationships between features across many samples.

2.5 Out-of-Distribution Detection

This section discusses OOD detection, the field which attempts to tackle the second problem discussed in the introduction and which is the main focus of this thesis; that ML models have significantly worse performance on OOD data points and will often "fail silently", making completely wrong predictions with apparent high confidence [39]. OOD detection is a developing field, and still in an initial stage [40]. In 2017, [16] proposed a baseline OOD detection method. This section will discuss this method and the methods which follow it.

2.5.1 Motivation for Out-of-Distribution Detection

When training a model using supervised learning, we implicitly use the "closed-world assumption", which means that we assume that test data will be drawn from the same distribution as the training data [18]. However, when a model is deployed, the data we see may not obey this assumption. Without OOD detection, the model will behave in the exact same way when encountering OOD samples or in distribution (ID) samples,

and may even claim to be highly confident in its prediction although the sample is far away from the distribution of the training data [41, p. 1]. In any system where models make high impact decisions, this is a huge problem. We do not want a medical AI system to attempt to classify a rare disease that was not part of the training data, nor do we want an autonomous car to continue to drive on snowy dirt roads if its training data only contains the sunny streets of San Francisco. Thus, OOD detection methods are necessary, so that OOD samples can be caught before the model takes any (potentially catastrophic) action.

Intuitively, one might assume that distinguishing ID and OOD samples from each other can be solved by simple binary classification using a dataset of ID samples and one of OOD samples. Indeed, if one has sufficient amount of high quality OOD samples, this can be done. However, this can be difficult to obtain in practice [18, p. 15]. A key problem is that OOD data is, almost by definition, data which we do not have at development and training time of a specific model. OOD data is unexpected and surprising data, which cannot be predicted before hand; we cannot possibly predict all situations a self-driving car could end up in, nor account for every single disease in the world when developing a medicinal AI system. Thus, we require more sophisticated methods of OOD detection, which catch OOD data points in a general manner and do not rely on explicit examples of OOD data. Indeed, the majority of methods developed in the field of OOD detection do not rely on such auxiliary datasets [7].

2.5.2 Semantic versus covariate shift

The first distinction to make in OOD detection tasks is whether an OOD sample is OOD because of *semantic* or *covariate* shift. Semantic shift refers to samples with different classes than the ones the model is trained on. A picture of a giraffe would represent a semantic shift for a model trained to differentiate between different breeds of dogs, as a giraffe does not belong to any breed of dog. Covariate shift refers to samples which come from a different distribution while still belonging to one of the classes of the original data set. An image of a Beagle puppy could represent covariate shift for a dog breed classifier despite Beagle being one of the classes, if all ID images were of adult dogs. Likewise, an image of a dog in a dark room could represent covariate shift, if all the ID images were of dogs outside, in well lit conditions. Figure 2.10 shows this distinction visually. Here, we can easily see the difference between covariate and semantic shift; covariate shifted images come from the same classes as ID samples, while semantically shifted images come from completely unknown classes.

Figure 2.10: Figure showing in-distribution, covariate shifted and semantically shifted images for an imagined dog breed classifier. Images are taken from the ImageNet dataset

The detection of semantic shift, as opposed to covariate shift, is the main focus of most OOD detection tasks [18]. In many applications, it is expected that the model should be able to generalize its prediction to covariate-shifted data, and therefore the focus is on detecting semantic shift. However, the field of medical image classification is one where detecting covariate shift is also important, as the model should only make predictions on data points which are very similar to its training data [18].

Given that the detection of semantic shift has been the main focus of most OOD literature, my work will primarily deal with semantic shift as well. Thus, unless otherwise specified, when I refer to OOD data points, I mean data points which are semantically shifted, i.e that come from another class than those the model has been trained on.

2.5.3 Benchmarking

The performance of an XAI is hard to quantify, because the quality of an explanation is not easily reduced to a number. For OOD detection, performance is much easier to measure, as the problem can be described as a binary classification problem, with OOD and ID samples as the positive and negative class, respectively. Thus, we can calculate many different metrics and compare methods against each other. For OOD methods, the two most common metrics to report are FPR95 and the AUROC (see section 2.3.2). It is common to use ImageNet or CIFAR as the ID dataset, and calculate FPR95 and AUROC on other datasets which contain no overlapping class labels. When selecting OOD datasets, it is common to differentiate between **Near-OOD** and **Far-OOD**. Far-OOD samples are samples which are drastically different from the ID samples, while Near-OOD samples only differ slightly. For a cat-versus-dog classifier, a tiger and a wolf would represent Near-OOD semantic shift, while a plane and a car would represent Far-OOD semantic shift. As one might expect, detecting Near-OOD samples is much harder than Far-OOD.

In 2021, [18] defined a generalized OOD detection framework, and in 2023 [7] introduced a comprehensive benchmark called OpenOOD, which evaluates all relevant OOD methods under this framework. Prior to this work, different methods were tested on different benchmarks, with different image preprocessing procedures, and with other externalities which inhibited effective comparison between methods [7]. In 2024 OpenOOD was further improved, with the addition of more benchmarks, more methods and the inclusion of vision transformer models [6]. OpenOOD is "is the only work that comprehensively evaluates a wide range of OOD detection methods on multiple benchmarks of various sizes" [6], thus making it an obvious choice for this thesis.

OpenOOD includes 11 different benchmarks across Anomaly Detection, Open Set Recognition and OOD detection, three fields which are very closely related. Of these, 4 benchmarks are used for standard OOD detection, which are the ones I will be concerned with. Each benchmark is defined by an ID dataset, with 6 or more corresponding OOD datasets, separated into Near-OOD and Far-OOD. For each benchmark, AUROC, AUPR and FPR95 is reported over all OOD datasets, and Near-OOD AUROC is used to rank methods against each other.

2.5.4 Methods

This section will follow the same outline as section 2.4; firstly, the overarching categories of methods will be discussed, followed by a more detailed look at a selection of specific methods within the field.

The field of OOD is separated into four categories of methods [18]:

- Classification-based methods
- Density-based methods
- Distance-based methods
- Reconstruction-based methods

All methods can also be categorized by whether they are post-hoc or training based. Post-hoc methods take an already trained network and attempt to extract information which separates ID and OOD samples out of the network during inference. These

methods have the obvious advantage that they can work out of the box with large pre-trained network without requiring expensive training from scratch. Training based methods train the network in ways which maximize the difference between ID and OOD samples. These methods do not necessarily require OOD samples, but can train using auxiliary loss functions which amplify the differences in network behaviour when faced with OOD data as opposed to ID. Regardless, these methods come with a much higher computational requirement than post-hoc methods, as they require training from scratch or at least retraining using the new loss criterion. Given the fact that post-hoc methods can be applied to trained networks out of the box, it is quite common to combine both post-hoc and training strategies to achieve the best performance.

Below follows a short explanation of each the four categories mentioned above.

Classification-based methods

Classification-based methods usually use the softmax score or logits of a model to attempt to distinguish OOD and ID samples. [16] made the observation that while the softmax score may be a poor indication of the actual confidence of the model on a single data point, it is still higher on average for ID samples as opposed to OOD samples. By using this simple distinction, they created a baseline model which separated OOD and ID samples. Using input perturbations and temperature scaling, [17] further improved on this method, by amplifying the difference in softmax score of ID and OOD data.

More generally, classification-based methods do not need to use the softmax score, but may attempt to find any metric which separates the distribution of ID samples from OOD samples. Figure 2.11 shows the probability density for an unspecified metric for both OOD and ID samples. The goal of classification-based OOD detection is to find metrics or training methods which make these probability densities have as little overlap as possible, such that they are easily separated by a threshold. The threshold, often denoted as δ, is a parameter that needs to be set at a specific value based on the values of a validation ID and validation OOD data set. Often, one sets the δ such that a certain percentage of OOD samples in the validation set are correctly detected, for example 95%.

Density-based methods

Density-based methods explicitly try to model the in-distribution, which is then used to detect outliers in low likelihood regions. Although the idea is intuitive, learning the distribution of the data set can often be prohibitively expensive, and thus these methods often lag behind classification-based methods [18].

Distance-based methods

Distance-based methods attempt to detect OOD samples by calculating their distance to ID samples. Many different distance measures are used, such as the Mahalanobis distance to estimated Gaussian distributions of the ID classes, cosine distance to the first singular vector of the ID dataset or the Euclidean distance in an embedding space.

Reconstruction-based methods

Reconstruction-based methods are based on encoder-decoder frameworks, where the core idea is that the model will be much worse at reconstructing OOD data than ID. By

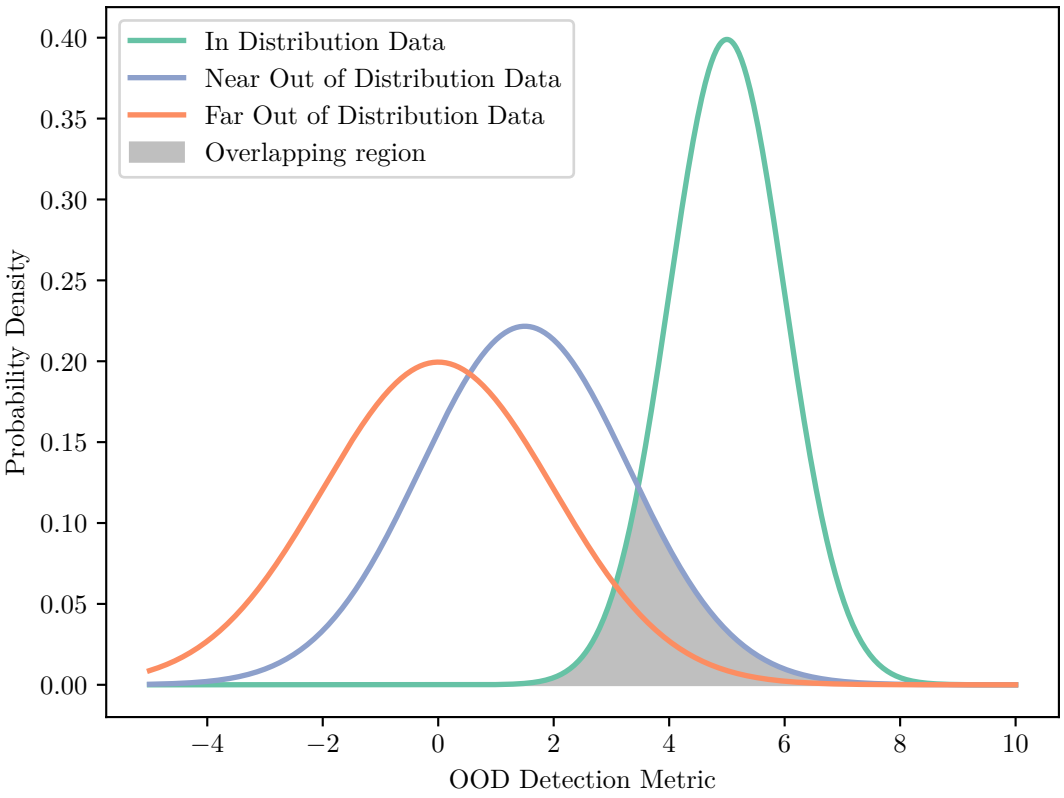

Figure 2.11: Graph showing the distribution of hypothetical ID, Near-OOD and Far-OOD data for an unspecified metric. The shaded region shows the overlap between the ID and OOD samples.

measuring the reconstruction loss, we can detect OOD samples.

2.5.5 Specific methods

Like in section 2.4.4, I here describe a selection of OOD detection algorithms which have special relevance for the methods introduced in chapter 3. These four methods are MSP, MLS, VIM and `COMBOOD`. After this, I give a short overview of methods which are SoTA in the field.

Baseline models

The baseline model created by [16] is extremely simple, yet effective. It simply compares the softmax score the predicted class to a threshold, and labels it as OOD if it falls below this threshold. Let us assume we have a model $f : \boldsymbol{x} \to \mathbb{R}^C$ that takes an input $\boldsymbol{x}$ (which may be an image, a vector of values, or something else) and returns a vector of logits with length equal to the number of classes C. If we define a softmax score function

$$S_i(\boldsymbol{x}) = \frac{\exp\left(f_i(\boldsymbol{x})\right)}{\sum_{j=1}^{N} \exp\left(f_j(\boldsymbol{x})\right)}, \tag{2.5}$$

then the OOD detector has the following simple form, given a threshold δ:

$$g(\boldsymbol{x}; \delta) = \begin{cases} \text{in} & \max_i S(\boldsymbol{x}) \geq \delta \\ \text{out} & \max_i S(\boldsymbol{x}) < \delta \end{cases}, \tag{2.6}$$

As explained previously, the δ must be set by the user of the system based on results from a validation set of ID and OOD data.

This method reasonably well, because the softmax scores for ID data generally is higher than for OOD data. Two years later, [42] showed that this baseline could be improved in settings with larger datasets by forgoing the softmax normalization and instead only looking at the maximum logit, with the even simpler form

$$g(\boldsymbol{x}; \delta) = \begin{cases} \text{in} & \max_i f(\boldsymbol{x}) \geq \delta \\ \text{out} & \max_i f(\boldsymbol{x}) < \delta \end{cases}, \tag{2.7}$$

Perhaps surprisingly, both these methods are quite good at detecting OOD samples. In fact, when combined with outlier exposure, MSP is SoTA on several benchmarks.

Virtual Logit Matching (ViM)

[11] attempts to improve OOD detection by calculating a score based on the feature, the logit and the softmax probability at once, as opposed to just one of them. By looking at all three elements in conjunction, they see an increase in performance over models which only rely on a single input source (such as the previously mentioned ODIN).

The reasoning behind not just looking at the logits or softmax probability is that there is a lot of information that is lost when going from features to logits [11]. Once we project the features down to logits, we have only class dependent information, and have lost the class agnostic information which is contained within the features. To show how this information is lost, the authors give an example based on null space analysis [43]:

Let us assume that we have a simplified network with only a single layer. Then, we have $\hat{\boldsymbol{y}} = W\boldsymbol{x}$, where $\hat{\boldsymbol{y}}$ is the vector containing the logits, $\boldsymbol{x}$ is the feature vector of the input (with an additional 1 for the bias term) and W is the matrix containing the weights and biases transforming the feature vector into logits. A null space $\text{Null}(W)$ of a matrix W is the set of all vectors that map to the zero vector, such that $W\boldsymbol{a} = \boldsymbol{0} \iff \boldsymbol{a} \in \text{Null}(W)$. The null space of a matrix may be trivial (empty), but a matrix which projects vectors to a lower dimension have non-trivial null spaces. Given that the final layer of a neural network projects down to logits, which are the same dimension as the number of classes, this will almost always be the case. Because of the distributivity of matrix multiplication, we have the following:

$$W(\boldsymbol{x} + \boldsymbol{a}) = W\boldsymbol{x} + W\boldsymbol{a} = W\boldsymbol{x} + \boldsymbol{0} = W\boldsymbol{x} \tag{2.8}$$

The vector $\boldsymbol{x}$ can be decomposed into $\boldsymbol{x}^W + \boldsymbol{x}^{\text{Null}(W)}$, where $\boldsymbol{x}^W$ is the projection of $\boldsymbol{x}$ onto the column space of W and $\boldsymbol{x}^{\text{Null}(W)}$ is the projection of $\boldsymbol{x}$ onto the null space of W. It follows from this and equation 2.8 that when going from features to logits using the projection $W\boldsymbol{x}$, we lose all information contained in $\boldsymbol{x}^{\text{Null}(W)}$. [43] shows how this

can be exploited by adversarial methods, by creating images with added noise derived from the null space of a matrix within the network, which are classified as if the noise was not present, despite having no resemblance to the original image. See figure 2.12.

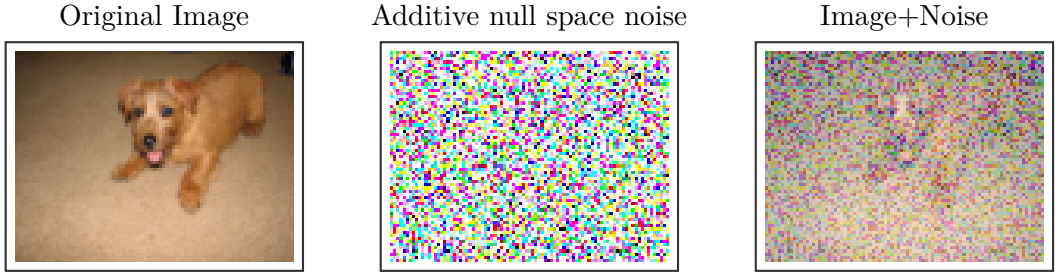

Figure 2.12: Diagram showing the findings of [43] visually. By sampling null space noise from the network, we can create an image (the image to the right) which is completely distorted, but which is indistinguishable for the network and given the exact same prediction as if the noise was not present (the image to the left).

From this, we can see that potentially large amounts of information can be lost when going from features to logits. Using this information, it is also possible to perform OOD detection, as shown by [43]. Another method which uses the features performs Principal Component Analysis (PCA) and looks at the residual information lost when using the first N principal components [44]. However, the information in the features is still class agnostic, and [11] aims to go beyond using just one input source and combine several elements of the network.

To do this, they propose using a *Virtual Logit*. The Virtual Logit is calculated as follows: First, they center the feature space, so that "it is bias free in the computation of logits" [11]. They then perform PCA as in [44], and calculate the residual of $\boldsymbol{x}$ with regards to the principal components, which is the projection $\boldsymbol{x}$ onto the null space of the principal subspace P. The residual represents the information lost when using the projection P.

$$\text{Residual}(x) = ||\boldsymbol{x}^{\text{Null}(P)}|| \tag{2.9}$$

This value is scaled based on the average values of the maximum logit across the dataset, and is appended to the rest of the logits as a Virtual Logit:

$$l_0 := \alpha||\boldsymbol{x}^{\text{Null}(P)}|| \tag{2.10}$$

This now takes part in the computation of the softmax values, and thus is affected by the size of the rest of the logits. They call the softmax value of the Virtual Logit the *ViM score*. In this way, the ViM score represents the size of the residual in comparison with the predictions of the model. If the model is very confident, then the norm of the residual will be small in comparison, and the ViM score will be low. If the residual is very large, the ViM score will be higher, and more indicative of an OOD sample. In this way, [11] have combined information from the feature, the logit and the softmax probability level to perform OOD detection.

COMBOOD

`COMBOOD` [10] is another OOD detection method which combines information from different sources to increase performance. Unlike VIM, which combines different internal

signals from the network, COMBOOD combines information from two different metrics calculated from the feature space. The two metrics are Mahalanobis distance and nearest neighbour distance, which have both seen decent performance on their own [45, 46]. [10] builds on these works by showing that their combination into one single score can increase performance far above either one. Indeed, the performance of COMBOOD is State-of-the-Art, being the highest performing OOD detector on the ImageNet200 and ImageNet1K benchmarks in the OpenOOD framework.[3]

To understand how COMBOOD works, we must first understand how Mahalanobis and nearest neighbour distance work as OOD detectors. Mahalanobis distance assumes that the features generated by the network are distributed as a multivariate Gaussian, and calculates the distance based on the mean and standard deviations of this Gaussian. This method is a generalization of calculating the Z-score for a univariate Gaussian, and has the following form, given a mean $\boldsymbol{\mu}$ and covariance matrix Σ, calculated over the ID training set:

$$\text{Mahalanobis}(\boldsymbol{x}) = \sqrt{(\boldsymbol{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{x} - \boldsymbol{\mu})} \tag{2.11}$$

Nearest neighbour distance has the advantage that it does not impose any assumptions about the distribution of the feature space, and is thus non-parametric. The k-nearest neighbour distance, is simply the distance from a given datapoint to the k'th nearest neighbour in feature space. [45], which used this distance metric for OOD detection, normalized the features and used Euclidean distance as the distance metric, giving us the following equation:

$$\text{NN}(\boldsymbol{x}) = \|\boldsymbol{x}^* - \boldsymbol{z}_k\|_2, \tag{2.12}$$

where $\boldsymbol{x}^*$ is the normalized feature $\boldsymbol{x}$ and $\boldsymbol{z}_k$ is the k'th nearest neighbour from the ID training set.

COMBOOD combines these metrics by computing "confidence scores" for each distance, which are then added together to produce a final score. In addition, they find that by using different feature extracting methods for each of the different methods, their combined performance can be enhanced considerably, concluding that "COMBOOD performs best when the nonparametric and the parametric components use different feature extraction strategies, penultimate layer embeddings for the former and global extrema of the features for the latter" [10].

Other State-of-the-Art methods

As mentioned previously, OOD detection is a developing field, with no method which definitely defeats all others across different benchmarks. I will now describe shortly a selection of methods which performs well on one or more benchmarks. AdaSCALE [47] is a scaling method which scales the activations of a network based on the OOD likelihood as calculated by perturbation and achieves SoTA results on the ImageNet200 and ImageNet1K benchmarks. ASH [48] creates a simplified representation of a specific layer and feeds it through the network, and uses the energy score of the modified prediction for OOD detection. Augmentation based methods such as RotPred [49] and PixMix [50] have also been used in conjunction with other detectors to achieve SoTA results on CIFAR10. Energy based OOD detection [41], which uses an energy score as opposed to the MSP also achieves high results on both CIFAR10 and CIFAR100. The

[3] https://zjysteven.github.io/OpenOOD/

methods mentioned above, such as MSP and `COMBOOD`, are also SoTA within the field of OOD detection.

2.6 Related work

In a broad sense, this work is about OOD detection. In this way, the field of OOD detection, and methods described in the preceding sections, can be thought of as constituting the related work. However, more specifically, I attempt to use XAI methods to enhance OOD detection performance, and thus we may look towards previous work which has attempted a similar combination.

While the combination of XAI and OOD detection has been explored in many previous works, the majority of them focus on explaining why a data point was marked as OOD, as opposed to using XAI to aid the detection itself. [51], [52] and [53] are papers which combine XAI and OOD for this purpose. Within network security, XAI has been as part of anomaly detection systems to detect malicious or faulty network traffic. Here, it has been used to explain detections [54, 55], but also to aid in detection itself by inspecting the explanations of the detection system [56, 57]. These methods thus use XAI to aid OOD detection in a similar manner to my work, however they are strictly focused on sequential network traffic data as opposed to images, and are mostly concerned with detection "unnatural" data samples such as intentionally malicious traffic or that generated by faulty equipment, as opposed to natural OOD data caused by semantic or covariate shift occurring when a model is deployed.

[9] is the most relevant previous work. Here, the authors explicitly aim to use XAI to improve OOD detection on images. They do this by looking at saliency maps produced by a GradCAM-based XAI method (section 2.4.4) during inference, i.e the heatmaps that explain which parts of the image was most influential to classify the image as a specific class. Using these heatmaps, they perform distance-based OOD detection (section 2.5.4): By collecting all explanations for each image in the ID dataset, they are able to construct archetypical explanations, and can make clusters of explanations. To perform OOD detection, they simply compare the explanation of a new data point to the clusters of archetypical explanations, and mark it as OOD if it has a distance which is over a certain threshold.

This method performs decently on toy benchmarks, achieving scores similar to SoTA methods when using *Fashion MNIST* as ID and *MNIST* as OOD. However, this method fails catastrophically in more complicated scenarios, achieving an AUROC score of only 52% on *CIFAR10* vs *SVHN*, which is no better than pure guessing and far below any other method. The paper thus ends with the authors concluding that "OoD detection approaches that are specifically designed for the purpose achieve in general better detection scores at the cost of an additional computational burden in the model's construction" [9].

For more potential related work, we can look to OpenOOD [6, 7], which aims to provide a comprehensive benchmark of all relevant methods in the field of OOD detection. Out of all 41 OOD detection methods included in this benchmark, there are no methods which use XAI. However, as many XAI methods utilize the gradients of the network to generate saliency values, we could also consider OOD detection models which utilize gradients in some form as tangentially related to this thesis. In this regard GradNorm [58] is related, as they utilize the norm of the gradients of the network with respect to the Kullback-Leibler distance between the outputs and a uniform distribution to perform OOD detection.

From the absence of any relevant method utilizing XAI in OpenOOD and from the poor results of [9], we can see that the potential for a truly effective OOD detection system using XAI has not been fully realized in any previous work.

2.7 Summary

In this chapter, I have given a short introduction to machine learning in general, as well as thorough introductions to the fields of XAI and OOD detection. Specifically, I have also given detailed descriptions of the XAI and OOD detection methods which are relevant to the development of the three frameworks I will introduce in the next section.

Chapter 3

Methodology

As shown in the preceding chapter, there exists a wide range of XAI methods. These methods exploit gradient information, differences in output scores when altering model inputs, marginal contributions of input features, as well as many other intricacies of deep learning models. The core idea of this master thesis is that these methods, in their attempt to explain a model decision, may also inadvertently extract information which is valuable for OOD detection. Thus, there may be an unexplored potential in these methods to function not just as explanations, but also as classifiers which allow us to separate ID and OOD data. Intuitively, we might expect the explanations to be more spread out on OOD images, given that there are (by definition) no objects of interest in the image that the model can definitely be said to focus on. In contrast, we might expect an explanation on an ID image to be more focused on a specific area, which contains an object of interest. Furthermore, given that saliency methods give a numerical value to each region of the image, we might be able to extract information about the "OOD-ness" of an image by inspecting the magnitudes of these values. Intuitively, it may be the case that such values should be lower for OOD than ID, reflecting the higher uncertainty present in OOD data.

3.1 Proposed XAI frameworks for OOD detection

With the preceding intuitions in mind, I present three frameworks for OOD detection which utilize XAI methods as part of their detection pipeline.

3.1.1 Stand-alone saliency framework: Saliency Aggregation

As we have seen from section 2.6, there has been little research into using explanations for OOD detection, aside from the work of [9]. Thus, I begin by presenting a simple framework which uses saliency values generated by XAI methods to calculate an OOD score.

As mentioned previously, the field of OOD detection started with the simple baseline introduced by [16], which simply uses the MSP (i.e the confidence score of the predicted class) as a way to measure OOD. Later, it was shown that using the MLS could also serve as an effective baseline. As such, I propose a similarly simple baseline when using explanations for OOD detection. The analogue of the maximum logit in an XAI context can reasonably be said to be the explanation generated of the predicted class (the class corresponding to the maximum logit). As explained previously, this explanation will take the form of an $N \times M$ saliency map. This saliency map is not a single scalar value,

and does thus not make a suitable OOD score by itself. Instead, we may perform some form of aggregation on the saliency map (such as taking the mean, the vector norm, the variance or some other metric), and use this as the OOD score.

The intuition for this method is informed by the fact that there are many forms of aggregation over saliencies which one might reasonably expect to be different for ID and OOD data. As an example, let us consider the implications of aggregating in some way which captures the magnitude of the saliencies, such as the *mean*, the *vector norm*, or the *max value*. When we generate a saliency map using the predicted class, the XAI saliency method attempts to calculate a measure of importance for each region of the input image, with regard to this class. For ID data, as long as the model predicted correctly, we know that there really are regions of the input image which contain the predicted class. If we instead are looking at semantically shifted OOD data, we know that no input image contains any of the ID classes. Thus, when a neural network makes a prediction on such a data point, it will always be wrong, because it will always predict one of the ID classes. By generating a saliency map of this prediction, we are asking a method to decide how each region contributed to a false decision. Given that there are no objects of the predicted class, in any region, we may reasonably assume that the saliency values are very different than in an ID case, where such objects actually are present.

Saliency Aggregation based on magnitude

The first intuition which motivates the Saliency Aggregation framework is the idea that the magnitudes of XAI saliencies generated from ID and OOD data may be different. In this section, I will present two hypothetical scenarios which motivate why this might be the case. First, I present a simple example scenario (figure 3.1) where one might expect the ID saliencies to be higher than the OOD saliencies. Here, I imagine a model which has been trained to differentiate between different breeds of dogs. In the first case, it is given an image of a dog, and a prediction of "English Foxhound" is made, which happens to be correct. Generating an explanation for this prediction, each region of the image is given a measure of importance, calculated using gradient information, differences in prediction score on counterfactual examples or by other means. As there actually is an English Foxhound in the image, we may expect that these methods generate saliencies which have a magnitude which is higher than if there was no dog present.

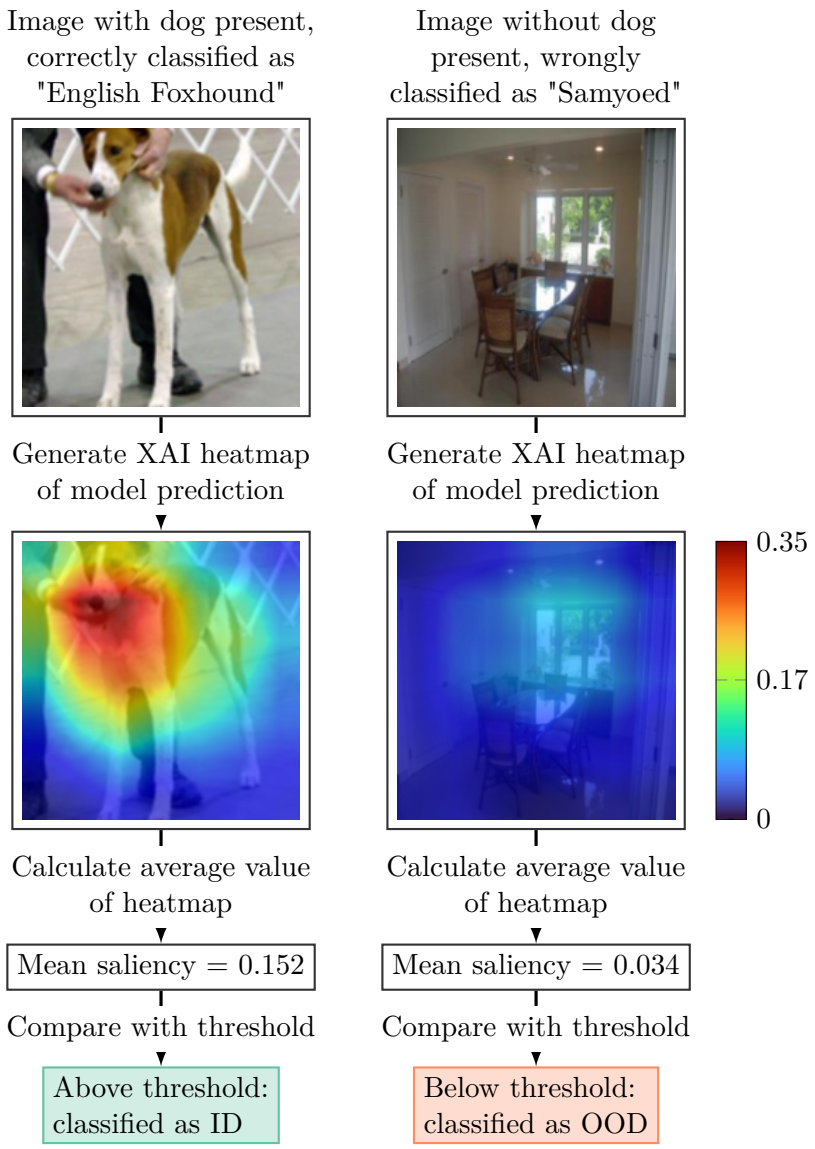

Figure 3.1: Figure showing the functioning of the Aggregate of Saliency OOD detector, using mean as the aggregation, in a hypothetical scenario where a model trained on images of dogs is shown an image with no dogs present. The heatmaps here show the maximum value of both saliency maps as dark red, reflecting the lack of normalization

In the second case, the model is given an image without any dogs present. Given that there is no class for images without dogs in them, the model will classify the image as one of the possible dog breeds. In this case, the model predicted the class of "Samoyed", a decision which can be considered essentially arbitrary. When a explanation is generated for this decision, the methods for calculating importance scores will most likely assign saliencies to most regions, given that no region contains a dog. As such, if we calculate mean saliencies for both the image with the dog and the one without, we expect the image with the dog to have a higher mean saliency. As long as we work with semantically shifted OOD data, it will always be the case that the prediction on OOD data is wrong, and thus we may also expect that the generated explanations in general output smaller saliencies.

On the contrary, we could also expect that OOD saliency maps have *higher*

magnitudes than ID saliency maps. As has been well documented, neural networks behave unpredictably when exposed to examples far from their training distribution [59–61]. Thus, it is not unreasonable to expect that explanations based on this unpredictable behaviour may also be unstable and unpredictable. This instability could lead to large outliers in saliency maps, which would give larger magnitudes when compared to ID data points. This lead to a second hypothetical scenario. Figure 3.2 shows such a scenario. Here, we imagine that the same dog breed classifier is shown two images: the first; an image of a Border Terrier in a park, a scenario which we could assume is quite common in the training dataset. The second is a night-time picture of a illuminated archway, an image which is very far away from the training distribution of the model, and contains many sharp changes in pixel intensity. In such a scenario, it is possible that a network will behave unpredictably, which may be reflected in the saliency map of the model. As explained in section 2.4.4, methods such as GradCAM, GBP and integrated gradients use gradient information from the network, which may be affected by sharp activations in the internals of the model.

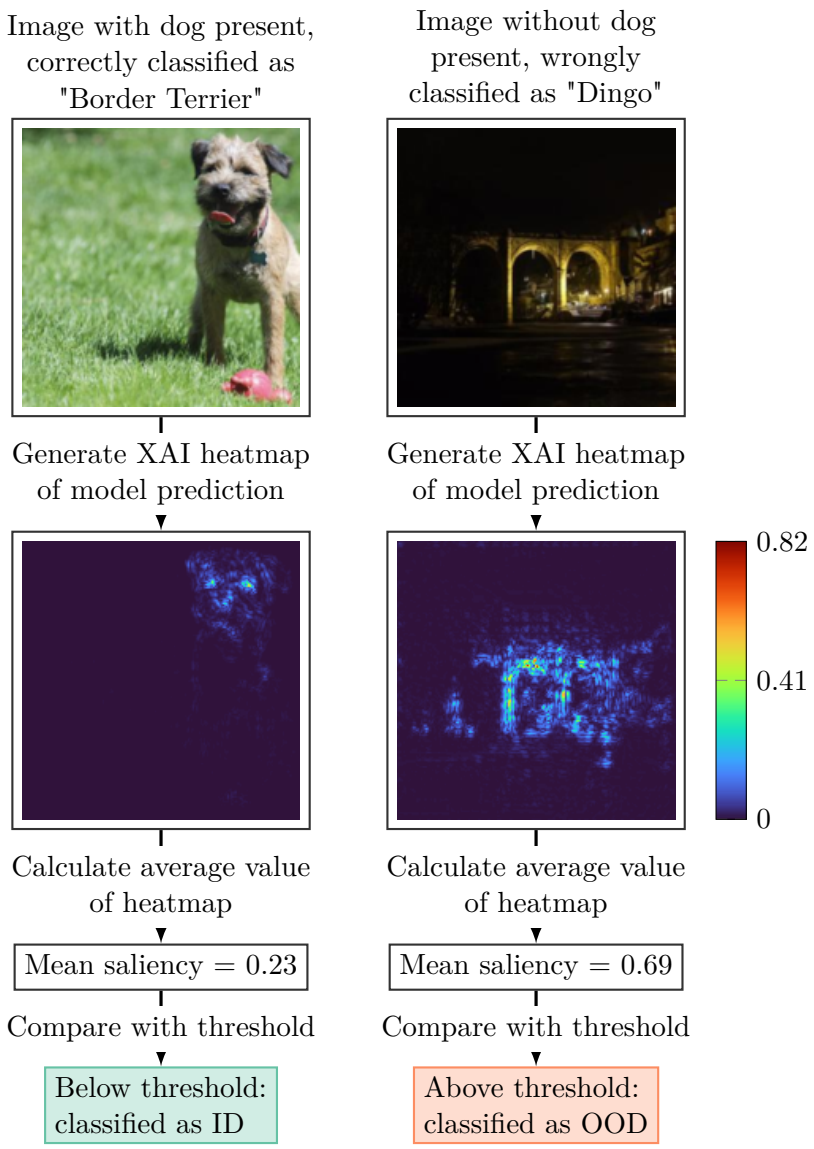

Figure 3.2: Figure showing the functioning of the Aggregate of Saliency OOD detector, using mean as the aggregation, in a hypothetical scenario where a model trained on images of dogs is shown an image with no dogs present. As opposed to the previous figure, here we assume that ID samples have lower mean values on average, and as such the thresholding is inverted.

Saliency Aggregation based on Statistical Dispersion

Apart from aggregations which convey information about the magnitude of the saliencies, we may also expect the variation or dispersion of the saliencies to be different between ID and OOD data. We might expect the heatmap on OOD data to be less concentrated and more evenly spread out, given that there are no actual objects of interest present. This would give OOD data points a low variance, and ID data points a high variance. Alternatively, we might expect OOD saliency maps to have large outliers, which could give OOD saliency maps high variance.

Figure 3.3 shows a hypothetical scenario in which calculating the spread could be beneficial in OOD detection. Like in the previous case, we imagine a model trained on dogs, which is fed two images, one which contains a dog and one which does not. In this

case, we do not care about the magnitudes, but instead only the spread of the values in relation to each other, and thus the heatmap is normalized. As we can see, in this case the heatmap is more spread out in the explanation where there is no dog present than for the image with a dog. By calculating a suitable metric, such as the Relative Mean Absolute Difference (RMD), Coefficient of Variation (CV) or another measure of statistical dispersion, we can get a single number which represents how spread out the heatmap is, regardless of its magnitude. Using these values, we can define a threshold which is below most ID data and above most OOD data, allowing us to separate these distributions.

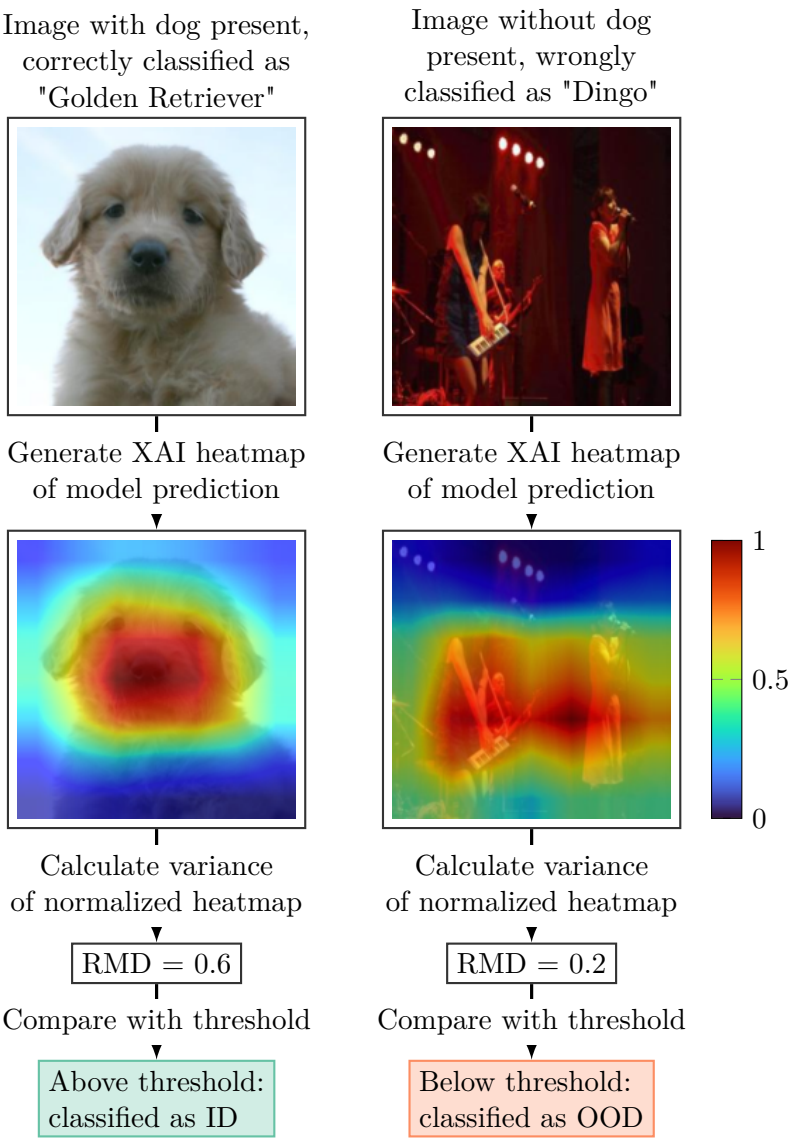

Figure 3.3: Figure showing the functioning of the Spread of Saliency OOD detector in a hypothetical scenario where a model trained on images of dogs is shown an image with no dogs present. The heatmaps are here based on the normalized values of each saliency map, meaning that magnitude information is ignored

On the contrary, we may expect the spread of saliencies to be higher for OOD data, if the saliency mapping methods are unstable and lack robustness. In such a scenario,

we could expect that the saliencies outputted on OOD data could be highly varied, while the saliencies of ID data is more stable.

Saliency Aggregation OOD detection framework: formal definition

After having introduced some motivating intuitions, we may formalize the framework. To define this detection framework mathematically, let us first define the necessary components. As in chapter 2, we assume we have a model $f : \boldsymbol{x} \to \mathbb{R}^C$. In this case, $\boldsymbol{x} \in \mathbb{R}^{D \times H \times W}$, i.e a D channel image of height H and width W. In addition, we define a general XAI saliency mapping method $s : (f, \boldsymbol{x}) \to \mathbb{R}^{N \times M}$. This function takes the model f and an input $\boldsymbol{x}$ and returns a N by M saliency map for the predicted class, i.e. the class corresponding to the highest logit. We also define a general aggregation function $A : \boldsymbol{x} \to \mathbb{R}$, where $\boldsymbol{x}$ can be of any shape. Given the fact that we do not know whether the saliency aggregation of a specific XAI method s and a specific model f will be larger or smaller for ID data, we must also decide whether larger or smaller values should be considered ID. To do this, we can calculate the mean value of a given saliency method and aggregation over a validation ID and validation OOD dataset, and compare their values. Requiring the presence of a validation set does not impose any actual limitations on the method, as a validation set is required by all OOD detection methods to be able to set the threshold δ. If we denote the mean value of the aggregation over the ID validation dataset as μ_{id}, and over the OOD dataset as μ_{ood}, we can use $\text{sign}(\mu_{id} - \mu_{ood})$ to multiply the OOD detection score by 1 or -1, respectively. This ensures that the OOD detector follows the convention of the OOD detection field, which is that ID samples should have higher scores than OOD samples on a given OOD detection metric.

Thus, the OOD detector has the following form, given a threshold δ:

$$g(\boldsymbol{x}; s, A, \delta) = \begin{cases} \text{in} & \text{sign}(\mu_{id} - \mu_{ood}) \cdot A(s(\boldsymbol{x}, f)) \geq \delta \\ \text{out} & \text{sign}(\mu_{id} - \mu_{ood}) \cdot A(s(\boldsymbol{x}, f)) < \delta \end{cases} \tag{3.1}$$

An astute reader may note that aggregation functions are permutation invariant, meaning that all positional information from the two-dimensional saliency maps is lost when aggregating. This may seem strange, as it is primarily the positions of the different values that is important when using XAI methods for explaining model predictions on images. However, there is good reason to believe that for many image classification tasks, the positions of the points of interest in an XAI saliency map does not carry much discriminative potential. For datasets such as CIFAR or ImageNet, there is a huge variety in the positions of the ground truth class (a dog may appear in the middle, the top right corner, or any other position and still be of the class 'dog'). As such, it is not a given that the removal of positional information will be massively detrimental. In fact, when [9] reflects upon the poor results of their saliency heatmap clustering for detecting OOD samples on CIFAR10, it is exactly this variability in position they highlight: "Indeed, the CIFAR10-C dataset does not afford the positional bias and low intra-class variability observed in the previous case studies: informative objects for the classes to be predicted appear in arbitrary parts of the image and have a high degree of compositional variability".

Given the exploratory nature of this thesis, it is reasonable to try many different forms of aggregation. Even when just considering the magnitude, it would be insufficient to just use the mean or maximum value of each saliency map, as each form of aggregation captures different qualities about the underlying data. The maximum value, for example,

is sensitive to outliers, which may be detrimental. The median value is far less sensitive to outliers, but given that one might expect ID data to have regions which are very important while most regions are relatively unimportant, outlier insensitivity may actually be undesirable. The vector norm and range are invariant to the sign of the saliencies, which means that if there are both large positive and negative values, these aggregates will return large scores. This is in contrast to the mean, median and third quartile, which will be lower if there are many negative values. In summary, we do not have the prerequisite knowledge about the distribution of saliency maps on ID and OOD data to make an informed selection, and as such we should cast a wide net when choosing which forms of magnitude aggregation to use.

Similarly, when choosing statistical dispersion aggregations, we should also include several different metrics. In addition, these metrics should also reduce the effects of magnitude, so that the two hypotheses can be tested individually. Metrics like the variance or standard deviation capture the spread of a distribution of values, but are also correlated with the magnitude: the variance of human arm length is far higher than the variance of human finger length, but this difference is mostly because arms are longer than fingers. Among dispersion metrics which are less dependent on magnitude, there are several to choose from: The CV is simply the standard deviation divided by the mean, and is a simple and intuitive method of standardizing dispersion values. The Quartile Coefficient of Determination (QCD) is another dispersion metric, which uses quartile information as opposed to the mean and standard deviation. This reduces the sensitivity to outliers when compared to the CV, but as stated previously, we do not have the requisite knowledge to know whether insensitivity to outliers is a benefit. As such, we should use both metrics. The QCD is equal to half of the interquartile range divided by the midhinge, which can be simplified to the difference between the third and first quartile divided by their sum. A third dispersion metric is the RMD. The RMD is equal to the mean absolute difference divided by the arithmetic mean. This metric has the desirable quality of being invariant to positive scaling. In addition, the RMD can be defined in terms of the second L-moment, while the standard deviation can be defined in terms of the second conventional moment. This makes the RMD a suitable complementary dispersion metric to the CV and QCD.

To summarize, the following aggregations have been used for the experiments in chapter 4.

Magnitude:

- Mean
- Median
- Vector norm
- Range
- Maximum
- Third Quartile

Statistical dispersion:

- Coefficient of Variation (CV)
- Quartile Coefficient of Determination (QCD)
- Relative Mean Absolute Difference (RMD)

These two categories correspond to the two hypotheses described in the preceding paragraphs, and will test whether either the magnitude or statistical spread of ID saliency maps differ substantially from those of OOD saliency maps.

3.1.2 Saliency integrated into existing OOD detection algorithms

Given the poor results of [9], one might expect that saliency maps on their own are insufficient to differentiate ID and OOD data. [10] has shown that by combining the OOD detection scores of two different methods, the total performance can be improved considerably. Furthermore, [11] has shown improvements by considering the softmax, logit and feature space in tandem. Thus, there is reason to believe that if XAI saliency maps have at least some discriminatory capabilities, these could be combined with traditional OOD detection methods, resulting in a performance gain. As such, I further present two frameworks which use combine XAI saliencies and traditional OOD detection metrics, to investigate if the addition of saliency values can enhance existing methods.

Saliency Aggregation plus Logit

In 2024, [10] introduced `COMBOOD`. This OOD detector combines the OOD detection scores of two different distance metrics; Mahalanobis distance and nearest neighbour distance. Each distance metric uses a different feature extraction method, which allows the two methods to collect different forms of information and complement each other. The combination is done by a simple unweighted addition of the confidence scores computed from the log distributions of the two metrics. `COMBOOD` achieves SoTA performance, and is (at the time of writing) by far the best performing method on ImageNet200 and ImageNet1K in the OpenOOD benchmark. Based on these results, I propose a similar method for combining XAI-based and traditional OOD detection metrics. If it is the case that XAI methods extract OOD information from the model in ways which are substantially different from traditional OOD detection strategies, we may see improvements similar to those observed by [10].

The field of OOD detection is vast, and in OpenOOD alone there are over 40 different OOD detection methods. All of these return a single metric and could thus be combined with an XAI OOD detection metric similar to how metrics are combined in [10]. However, investigating all such possible combinations would be a monumental undertaking, and is infeasible under the time constraints of a master thesis. Instead, I will constrain myself to introducing a simple baseline framework, combining Saliency Aggregation with the MLS.

Under this framework, the OOD score is thus a sum of a saliency aggregate and the maximum logit. However, due to the fact that both the logit and saliency values can be of arbitrary magnitude, we must normalize them before summing if we want each part to contribute equally to the final score. Thus, we can sum the Z-scores of each metric instead. This ensures that the values of the maximum logit and the saliency aggregate are distributed in about the same way. To calculate the Z-scores, we can simply subtract the mean and divide by the standard deviation over an entire ID validation dataset, for each metric. Thus, we calculate the mean and standard deviations of the maximum logit over an ID validation set μ^{id}_{MLS} and σ^{id}_{MLS}, as well as the mean and standard deviation of the aggregate of saliencies μ^{id}_{Agg} and σ^{id}_{Agg}. In addition, we must calculate the mean value of the aggregation metric over a validation OOD dataset, as we do not know whether a given aggregation is higher or lower for ID data. We denote this value as μ^{ood}_{Agg}. We now have the necessary values required to define this framework mathematically:

As in the previous section, we assume we have a model $f : \boldsymbol{x} \to \mathbb{R}^C$, an XAI saliency mapping method $s : (f, \boldsymbol{x}) \to \mathbb{R}^{N \times M}$, and an aggregation function $A : \boldsymbol{x} \to \mathbb{R}$. Let us denote $\text{sign}(\mu^{id}_{\text{Agg}} - \mu^{ood}_{\text{Agg}})$ as S, for readability. An OOD detector under this framework then has the following form, given a threshold δ:

$$g(\boldsymbol{x}; s, A, \delta) = \begin{cases} \text{in} & S \cdot \frac{A(s(\boldsymbol{x},f)) - \mu^{id}_{\text{Agg}}}{\sigma^{id}_{\text{Agg}}} + \frac{\max_i S(\boldsymbol{x}) - \mu^{id}_{\text{MLS}}}{\sigma^{id}_{\text{MLS}}} \geq \delta \\ \text{out} & S \cdot \frac{A(s(\boldsymbol{x},f)) - \mu^{id}_{\text{Agg}}}{\sigma^{id}_{\text{Agg}}} + \frac{\max_i S(\boldsymbol{x}) - \mu^{id}_{\text{MLS}}}{\sigma^{id}_{\text{MLS}}} < \delta \end{cases} \tag{3.2}$$

In fact, this detector can be simplified somewhat. Consider the following:

$$S \cdot \frac{A(s(\boldsymbol{x}, f)) - \mu^{id}_{\text{Agg}}}{\sigma^{id}_{\text{Agg}}} + \frac{\max_i S(\boldsymbol{x}) - \mu^{id}_{\text{MLS}}}{\sigma^{id}_{\text{MLS}}} = \tag{3.3}$$

$$S \left(\cdot \frac{A(s(\boldsymbol{x}, f))}{\sigma^{id}_{\text{Agg}}} - \frac{\mu^{id}_{\text{Agg}}}{\sigma^{id}_{\text{Agg}}} \right) + \frac{\max_i S(\boldsymbol{x})}{\sigma^{id}_{\text{MLS}}} - \frac{\mu^{id}_{\text{MLS}}}{\sigma^{id}_{\text{MLS}}} = \tag{3.4}$$

$$S \cdot \frac{A(s(\boldsymbol{x}, f))}{\sigma^{id}_{\text{Agg}}} + \frac{\max_i S(\boldsymbol{x})}{\sigma^{id}_{\text{MLS}}} - \left(S \cdot \frac{\mu^{id}_{\text{Agg}}}{\sigma^{id}_{\text{Agg}}} + \frac{\mu^{id}_{\text{MLS}}}{\sigma^{id}_{\text{MLS}}} \right) \tag{3.5}$$

Notice how all the values in the third term of the above summation are constants; they do not depend on $\boldsymbol{x}$. Thus, we can disregard these terms, as all they do is shift all outputs by a constant value. The final OOD detector thus has the following form:

$$g(\boldsymbol{x}; s, A, \delta) = \begin{cases} \text{in} & S \cdot \frac{A(s(\boldsymbol{x},f))}{\sigma^{id}_{\text{Agg}}} + \frac{\max_i S(\boldsymbol{x})}{\sigma^{id}_{\text{MLS}}} \geq \delta \\ \text{out} & S \cdot \frac{A(s(\boldsymbol{x},f))}{\sigma^{id}_{\text{Agg}}} + \frac{\max_i S(\boldsymbol{x})}{\sigma^{id}_{\text{MLS}}} < \delta \end{cases} \tag{3.6}$$

It should be reiterated that the second term in this OOD detection metric need not be the MLS. In theory, it can be any metric calculated from the network on a given input, for example SoTA methods such as AdaSCALE [47], RotPred [49] or EBO [41].

SaliencyVIM: Virtual Logit Matching with Saliencies

As a final proof-of-concept framework, I propose adding saliency values directly into preexisting OOD detection models. Like the framework introduced in the previous section, there are many OOD detection models which could be the basis for such a framework. However I will limit myself to a single method in this case as well, to allow for rigorous testing within the time constraints of this thesis. Out of the many possible methods, VIM [11] is a fitting choice, as VIM attempts to combine information from different sources, namely from the feature, logit and softmax probability space. As we can recall from section 2.5.5, VIM uses a "virtual logit" which is calculated from the output features on the penultimate layer and appended to the original logits. As a proof-of-concept integration of saliencies into this method, we may append XAI saliencies to the features prior to the generation of this virtual logit. The virtual logit in VIM is calculated as the information lost when performing a PCA feature reduction on the features. By appending the saliencies, the PCA takes into account not just how the penultimate features vary together, but also the interplay between the penultimate features and the

XAI saliencies, which may be substantially different between ID and OOD datasets. This increase in variance could increase the reconstruction loss when projecting OOD samples using the principal components calculated from the ID dataset, increasing the separability of ID and OOD data points. Because there is no equivalent to centering the feature space as described in [11] for saliencies, I keep the saliencies as they are while the features are centered.

To define this OOD detector mathematically, we follow the definition of [11] very closely. We assume we have a model $f : \boldsymbol{x} \to \mathbb{R}^C$ and a XAI saliency mapping method $s : (f, \boldsymbol{x}) \to \mathbb{R}^{N \times M}$. We further assume we can extract the penultimate features $h \in \mathbb{R}^K$, with a function $g : (f, \boldsymbol{x}) \to \mathbb{R}^K$. We denote these extracted features as $\boldsymbol{z}$, and the entire set of L feature vectors over the ID validation dataset as $Z \in \mathbb{R}^{L \times K}$. As in [11], we calculate an offset $\boldsymbol{o} = -(W^T)^+ \boldsymbol{b}$, where W and $\boldsymbol{b}$ are the weights and biases which transform the features into logits. The feature matrix Z is transformed by this offset: $Z^\dagger = Z + \boldsymbol{o}$. In addition to this feature matrix, we also calculate saliencies over the ID dataset, and flatten them to produce a saliency matrix $S \in \mathbb{R}^{L \times (N \cdot M)}$. These two matrices are concatenated, giving us the matrix $Y = \left[Z^\dagger, S\right]$. We then perform an eigendecomposition on $Y^T Y$, as part of the PCA and in accordance with the method described by [11]:

$$Y^T Y = Q \Lambda Q^{-1} \tag{3.7}$$

The first D columns of Q make up the D-dimensional principal subspace P. D is a hyperparameter that must be tuned, and is usually some fraction of K, the amount of logits in the penultimate layer (for example, if the number of logits is 1024, D may be 512 or 256). For this thesis, D has been fixed at 256, to avoid the extra computational overhead required for hyperparameter tuning. The remaining $K - D$ columns make up the null-space of P. Following the notation of [11], we denote the matrix making up these remaining columns as R. With this matrix, we can calculate the residual of $\boldsymbol{x}$ onto P as $\boldsymbol{x}^{P^T} = RR^T \boldsymbol{x}$. The virtual logit then has the following form:

$$l_0 := \alpha ||\boldsymbol{x}^{\text{Null}(P)}|| = \alpha \sqrt{\boldsymbol{x}^T R R^T \boldsymbol{x}} \tag{3.8}$$

This is exactly the same as the virtual logit for VIM, with the only difference being that R is calculated from the concatenated matrix Y, which includes both saliencies and logits, as opposed to only logits. To make it clear that R no longer depends solely on the network f and the input data, but also on the choice of saliency generator s, we can denote R as R_s. As with VIM, α is calculated from the average virtual and maximum logit over the ID dataset before the OOD detector is put into use.

The OOD detection score is the softmax score of the virtual logit when appended to the rest of the logits:

$$\text{SaliencyVIM}(\boldsymbol{x}) = \frac{e^{\alpha \sqrt{\boldsymbol{x}^T R_s R_s^T \boldsymbol{x}}}}{\sum_{i=1}^{C} e^{l_i} + e^{\alpha \sqrt{\boldsymbol{x}^T R_s R_s^T \boldsymbol{x}}}} \tag{3.9}$$

Thus, the formal definition of the SaliencyVIM OOD detector is as follows, given a threshold δ:

$$g(\boldsymbol{x}; \delta) = \begin{cases} \text{in} & \text{SaliencyVIM}(\boldsymbol{x}) \geq \delta \\ \text{out} & \text{SaliencyVIM}(\boldsymbol{x}) < \delta \end{cases} \tag{3.10}$$

Because we are no longer aggregating the saliencies, the choice of saliency method is no longer as free as with the previous two methods. When we wish to use the saliencies directly, we cannot use methods which output one value for every single pixel in the input image, as this would lead tens of thousands of dimensions over which to compute a Principal Component Analysis (PCA). This is computationally intractable. Instead, we must use methods which calculate saliencies over larger windows, such as occlusion, LIME or GradCAM.

3.2 Relation to existing methods

To further justify the idea that XAI saliency values can be used for OOD detection, I present proof that by choosing a specific XAI saliency mapping method and aggregation under the Saliency Aggregation framework, the resulting OOD detection score is equivalent to MLS, one of the baseline methods used for OOD detection. The MLS OOD detector simply uses the maximum logit as an indicator of OOD-ness, and has been described in more detail in section 2.5.5.

In this case, we choose mean aggregation, and GradCAM as the saliency mapping method. In addition, we assume that the classification stage of the CNN is a simple GAP over the feature map followed by a single linear layer. This is not an unreasonable assumption, as this is the classification head of all ResNet models. Finally, we choose to perform GradCAM on the final layer of the network, which is recommended by [29]. Under these conditions, the following is true:

Theorem 1. *The OOD score of Saliency Aggregation is equal to the MLS, up to a constant a:*

$$SalAgg(\boldsymbol{x}) = a \cdot MLS(\boldsymbol{x})$$

Proof. As defined in equation 3.1, the OOD detection score of Saliency Aggregation has the following form:

$$\text{SalAgg}(\boldsymbol{x}) = \text{sign}(\mu_{id} - \mu_{ood}) \cdot A(s(\boldsymbol{x}, f)) \tag{3.11}$$

In this special case, A is equal to *mean* and s is equal to *GradCAM*. Following [29], the saliency map generated by GradCAM has the following definition:

$$\text{GradCAM}(\boldsymbol{x}) = ReLU\left(\sum_k \left(\frac{1}{N \cdot M} \sum_i \sum_j \frac{\delta y^c}{\delta F^k_{ij}}\right) F^k\right). \tag{3.12}$$

Here, F^k is the k'th channel of the final convolutional feature map, while N and M are its dimensions.[1] The above equation simply describes averaging the gradients of the logit of class c for each channel, and using these values to perform a weighted sum of the channels in the feature map, as described in section 2.4.4. While c can be any class index, in this case, we define $c = \max_i f_i(\boldsymbol{x})$, i.e we calculate the saliency map for the predicted class, as defined by the framework. As will be explained in more detail later (section 3.5), we do not wish to perform ReLU rectification with any of our saliency mapping methods, as we wish to keep all information extracted by the saliency mapping, not just the values which increase the probability of the predicted class. Using mean as the aggregation, the OOD score is then:

[1] The attentive reader will notice that this is the same notation used for the dimensions of saliency maps throughout this thesis. This is intentional, as the saliency map generated by GradCAM has the same dimensions as the feature map on which the algorithm is performed.

$$\text{SalAgg}(\boldsymbol{x}) = \text{sign}(\mu_{id} - \mu_{ood}) \cdot \text{mean}\left(\sum_k \left(\frac{1}{N \cdot M} \sum_i \sum_j \frac{\delta y^c}{\delta F_{ij}^k}\right) F^k\right). \tag{3.13}$$

For the network as described above, the logit y^c for class c is calculated in the following manner:

$$y^c = \sum_k \text{mean}(F_k) \cdot W_{ck} \tag{3.14}$$

$$= \sum_k \left(\frac{\sum_i \sum_j F_{ij}^k}{N \cdot M} \cdot W_{ck}\right). \tag{3.15}$$

This equation simply describes GAP (all channels are averaged to a single number) followed by a single linear layer (each logit is a weighted sum of the average pooled channels, with the weights defined by a specific row/column in the weight matrix W). Given our definition of $c = \max_i f_i(\boldsymbol{x})$, $y^c = \text{MLS}(\boldsymbol{x})$. We return to the equation for $\text{SalAgg}(\boldsymbol{x})$:

$$\text{SalAgg}(\boldsymbol{x}) = \text{sign}(\mu_{id} - \mu_{ood}) \cdot \text{mean}\left(\sum_k \left(\frac{1}{N \cdot M} \sum_i \sum_j \frac{\delta y^c}{\delta F_{ij}^k}\right) F^k\right). \tag{3.16}$$

Given equation 3.15,

$$\frac{\delta y^c}{\delta F_{ij}^k} = \frac{W_{ck}}{N \cdot M}. \tag{3.17}$$

As we can see, the indices i and j have disappeared. This is to be expected, as global average pooling means that all values in each channel are multiplied by the same value when calculating the logit of a specific class. We may now substitute this derivative in equation 3.16:

$$\text{SalAgg}(\boldsymbol{x}) = \text{sign}(\mu_{id} - \mu_{ood}) \cdot \text{mean}\left(\sum_k \left(\frac{1}{N \cdot M} \sum_i \sum_j \frac{W_{ck}}{N \cdot M}\right) F^k\right). \tag{3.18}$$

We now perform some simple algebra, exploiting the fact that $\text{mean}(a \cdot \boldsymbol{x}) = a \cdot \text{mean}(\boldsymbol{x})$ and that $\sum_i c \cdot x_i = c \sum_i x_i$:

$$\text{SalAgg}(\boldsymbol{x}) = \text{sign}(\mu_{id} - \mu_{ood}) \cdot \text{mean}\left(\sum_k \left(\frac{1}{N \cdot M} \sum_i \sum_j \frac{W_{ck}}{N \cdot M}\right) F^k\right) \tag{3.19}$$

$$= \text{sign}(\mu_{id} - \mu_{ood}) \cdot \frac{1}{N \cdot M} \text{mean}\left(\sum_k \left(\sum_i \sum_j \frac{W_{ck}}{N \cdot M}\right) F^k\right) \tag{3.20}$$

$$= \text{sign}(\mu_{id} - \mu_{ood}) \cdot \frac{1}{N \cdot M} \text{mean}\left(\sum_k \left((N \cdot M) \frac{W_{ck}}{N \cdot M}\right) F^k\right) \tag{3.21}$$

$$= \text{sign}(\mu_{id} - \mu_{ood}) \cdot \frac{1}{N \cdot M} \text{mean}\left(\sum_k W_{ck} \cdot F^k\right) \tag{3.22}$$

$$= \left(\text{sign}(\mu_{id} - \mu_{ood}) \cdot \frac{1}{N \cdot M}\right) \cdot \left(\sum_k W_{ck} \cdot \text{mean}(F^k)\right). \tag{3.23}$$

The first factor above is a constant. It does not depend on $\boldsymbol{x}$, as all values are calculated before inference, or are themselves constants. As such, we can denote this factor as a. We recognize the second factor as $y^c = \text{MLS}(\boldsymbol{x})$ as described in equation 3.15. We then have

$$\text{SalAgg}(\boldsymbol{x}) = a \cdot \text{MLS}(\boldsymbol{x}), \tag{3.24}$$

which was what we wanted to prove.

□

The constant factor has no effect on the OOD detection, as it just means that the thresholds δ will differ by this factor between the two detectors, with all predictions being the same for either method. Thus, theorem 1 states that Saliency Aggregation and MLS OOD detection are functionally equivalent given the conditions described above.

We have now shown that XAI saliency mapping methods, although they have been developed for an entirely different purpose than OOD detection, also collect information from the network which can be used for OOD detection.

3.3 Benchmarks

After having introduced the three novel OOD detection methods that I plan to test, I will now introduce the OOD detection benchmarks that will be used. To best align this thesis with the broader OOD detection field, I will use the OpenOOD framework, and their collection of ID and OOD datasets. Each OOD detection benchmark is defined by an ID dataset, and a selection of corresponding OOD datasets. As described in [6], the four standard OOD detection benchmarks included in OpenOOD are CIFAR10, CIFAR100, ImageNet200, and ImageNet1K. I have used all four benchmarks in this thesis, to ensure that the results are robust and easily comparable to the SoTA within the field of OOD detection.

3.3.1 CIFAR10/CIFAR100

CIFAR10 and CIFAR100 are two general object recognition datasets collected by [62]. These datasets are commonly used in AI research [63]. Each RGB image has dimension 32×32, and there are 50 000 training images and 10 000 test images for both datasets. In CIFAR10, each of the ten classes has 5000 training samples and 1000 test samples, while for CIFAR100 each of the one hundred classes has 500 training samples and 100 test samples. For CIFAR10, the ten classes are as follows: airplane, car, bird, cat, deer, dog, frog, horse, ship, and truck. The classes in CIFAR100 are in the same vein, but without any overlap with CIFAR10. OpenOOD presents a series of Near-OOD and Far-OOD datasets which are used in conjunction with these datasets. The collection of an ID dataset and corresponding OOD datasets constitute a benchmark. Both datasets have the same Far-OOD datasets, and also use TinyImageNet for Near-OOD evaluation. In addition, the CIFAR10 benchmark uses CIFAR100 as a Near-OOD dataset, while the CIFAR100 benchmark uses CIFAR10. Table 3.1 presents a short description of each dataset in the benchmarks, as well as the number of samples in the set for each dataset.

Dataset	Size	Description
CIFAR10/CIFAR100	10 000	General objects
Near-OOD		
CIFAR100/CIFAR10	10 000	General objects
TinyImageNet	7 793	General objects, downsampled from ImageNet
Far-OOD		
MNIST	70 000	Handwritten digits from 0 to 9
SVHN	26 032	Street view house numbers
Texture	5 640	Textural patterns
Places365	35 195	Places, scenes, locations

Table 3.1: Table showing the datasets which constitute the CIFAR10 and CIFAR100 OpenOOD benchmarks

In addition, a sample of images from CIFAR10 and the corresponding OOD datasets are shown in figure 3.4. Given that the only difference between the CIFAR10 and CIFAR100 benchmark is that CIFAR10 becomes a Near-OOD dataset and CIFAR100 becomes the ID dataset, this figure will suffice to give an impression of both benchmarks.

Figure 3.4: Figure showing three example images from CIFAR10 and from the corresponding OOD datasets

3.3.2 ImageNet200/ImageNet1K

ImageNet is another ubiquitous image classification dataset, which is used in a large amount of computer vision works. In contrast to CIFAR, ImageNet images are substantially larger, at 256×256.[2] This increase in dimensionality could be used to argue that ImageNet tasks are more realistic, as it is rare that one would use 32×32 images in practical applications. The ImageNet dataset contains 1000 classes of general objects. In addition, there exists smaller versions, such as ImageNet200. As with the other ID datasets, OpenOOD contains a selection of OOD datasets which are chosen to test a model's ability to differentiate between different forms of OOD data. These OOD datasets, along with the ID dataset, constitute the OOD detection benchmark. Here, OpenOOD does not use ImageNet200 as a Near-OOD dataset for ImageNet1K, nor vice versa. Instead, both datasets use exactly the same OOD datasets, to facilitate straight comparisons between the performance on ImageNet1K and ImageNet200 [6]. Table 3.2 lists these datasets, and gives a short description for each one.

Dataset	Size	Description
ImageNet200/ImageNet1K	9 000/45 000	General objects
Near-OOD		
SSB-Hard	49 000	General objects taken from ImageNet21K
NINCO	5 879	General objects, manually curated by [64]
Far-OOD		
iNaturalist	10 000	Various plants
Texture	5 160	Textural patterns
OpenImage-O	15 869	General objects, manually curated by [11]

Table 3.2: Table showing the datasets which constitute the ImageNet200 and ImageNet1K OpenOOD benchmarks

In addition, figure 3.5 shows three example images for each dataset in the ImageNet200 benchmark. Like the preceding section, I will not show a separate figure for ImageNet1K, as the only change would be the ID dataset, which would show slightly different classes.

[2]It is common to center crop as part of preprocessing on ImageNet, and as such you will more often see the dimensions 224×224 mentioned in this thesis

Figure 3.5: Figure showing three example images from ImageNet200 and from the corresponding OOD datasets

3.3.3 Overview of testing environment

After having introduced the specific benchmarks, I will describe the testing environment that has been used, so that it is clear for the reader how the OOD detection metrics are calculated.

As we have seen, for a given benchmark, we have one ID dataset and a series of

OOD datasets, which are all semantically shifted in relation to the ID dataset. These OOD datasets are categorized into Near- and Far-OOD, depending on the degree of their semantic shift. The ID dataset is split into a train, validation and test set, while the OOD datasets are split into validation and test sets. A network is trained on the train ID set. Then, one OOD validation set is chosen to be used for hyperparameter tuning. Any hyperparameters that the OOD detection algorithm may have are tuned on this validation set and the ID validation dataset. After having trained the network and tuned the OOD detector, we now have every part needed to perform OOD detection and calculate performance metrics.

First, we calculate OOD detection scores over the entire ID test set. By convention, these are expected to be higher than for OOD samples. For example, if we use the MSP OOD detector as described in section 2.5.5, we would calculate the softmax score of the predicted class for all samples in the ID test set and store them. Then, for each OOD test dataset, we perform the same calculation of OOD detection scores. To calculate AUROC scores, each set of OOD test set scores is compared against the ID test set scores we calculated previously by denoting all ID samples as class 0 and all OOD samples as class 1. After having done this for each OOD test set individually, we have a series of Near- and Far-OOD AUROC scores. The average performance of our classifier on Near- and Far-OOD can then be calculated by averaging over these individual scores.

This process describes how the performance of an OOD detector can be calculated for a given ID dataset and set of OOD datasets. However, when developing a new method, one cannot repeatedly perform this process on the same ID and OOD datasets, as this will bias the final results. A common option in such a scenario is to simply use some benchmarks for development and some for the final tests. However, for this thesis, this is not sufficient. Firstly, to ensure comparisons with all methods tested under the OpenOOD framework, we should evaluate our performance on all benchmarks included within it. In addition, we have seen from the preceding section that the OOD datasets are shared between different ID benchmarks, which means that if we used the ImageNet200 benchmark as a development benchmark, we will have seen all OOD data of both ImageNet200 and ImageNet1K during development, which would disallow using ImageNet1K as a benchmark during testing. The same problem will arise on CIFAR100 if we use CIFAR10 as the development benchmark.

Because of this, I have separated all benchmarks into validation and test benchmarks. Specifically, all ID and OOD validation sets, and all ID and OOD test sets have been split in two equally sized subsets, of which one half has been used during development and the other half has been used during the final testing. This has been done for all four benchmarks, ensuring that the results reported in chapter 4 have not been biased by repeated testing on the same data. The development of new methods has been done on ImageNet200 and CIFAR10, as there is little benefit to using all four benchmarks during development. The final testing has been done on all benchmarks included in OpenOOD, as mentioned previously.

3.4 Networks

While it may be interesting to see how these new methods function on different network architectures, the combination of several different novel OOD detection algorithms and XAI saliency methods, as well as four different benchmarks, already presents a considerable amount of evaluation. Thus, to focus the thesis on comparing different OOD detection methods against each other, I believe it is best to fix other parameters

such as network architecture. With this in mind, I choose to limit myself to the ResNet [65] family of neural networks. In particular, I use ResNet18 for evaluation on CIFAR10, CIFAR100 and ImageNet200, and ResNet50 for evaluation on ImageNet1K. Given the scope of the of the thesis as described in section 1.3, I do not use methods which require retraining of the network. As such, all networks are pretrained and I do not perform any training as part of this thesis.

Aside from CNNs, Vision Transformers also perform exceptionally on computer vision tasks, and achieve SoTA results in many settings [66]. On ImageNet, they are even dominant, and the top 10 models when considering (top-1) accuracy are all based on vision transformers, as opposed to CNNs.[3] As such, one might question the choice of a CNN model, when newer and better models have been developed.

However, given that a large part of XAI methods have been developed under the CNN paradigm, many XAI methods are not easily adapted to vision transformers. Methods such as GradCAM and GBP exploit specific parts of CNN architectures when generating explanations [67], and are thus difficult to use with different architectures. To be able to use a broad section of the representative XAI methods in use today, it is thus preferable to use CNNs as opposed to vision transformers.

3.5 XAI Saliency Methods

The theory of each XAI saliency method used in this thesis has been described in section 2.4.4 of chapter 2. In the following section, I will describe the specifics of how I have applied each method in my efforts to better align them with the goal of separating ID and OOD data points.

For all saliency methods, one significant difference between my application and those commonly used for model explanation is that I modify all methods to return unnormalized and unrectified saliencies. For most XAI applications, it is common to rectify the saliencies such that negative saliencies are set to zero. In addition, some methods output normalized saliencies, as XAI saliency maps are intended to be inspected individually. In these cases, the absolute magnitude can be considered less relevant, and looking at saliencies relative to each other more informative. For my purposes, both negative values and the magnitude may convey important information which should not be discarded.

In addition, any parameters of each XAI method has been fixed, and no hyperparameter tuning has been performed. Given that I have introduced three different XAI OOD detection frameworks which have been tested on four different benchmarks, tuning five XAI methods was an additional computational burden that could not be afforded.

3.5.1 LIME

As is common when applying LIME to images, I have used segmentations of the input image to reduce the dimensionality of the input that is fed into the LIME algorithm. As explained in section 2.4.3, superpixel segmentation algorithms such as SLIC may increase the accuracy of the explanations, but are CPU-bound and thus introduce a considerable computational overhead (which is problematic when tens of thousands of samples are to be evaluated). Thus, I have chosen to stick to a simple rectangular segmentation

[3] https://paperswithcode.com/sota/image-classification-on-imagenet

instead. Each image is split into $N \times N$ equally sized rectangular regions, with N being set to 4.

3.5.2 Occlusion

Like with LIME, I have used a rectangular segmentation approach, with each image being split into $N \times N$ regions. As with LIME, N is set to 4, which ensures that accurate comparisons between LIME and occlusion can be made.

3.5.3 GradCAM

When utilizing GradCAM, a choice must be made about which convolutional layer one should calculate gradients from. As described in section 2.2.2, earlier layers have higher resolution (more precise spatial information) but lower field of view (less comprehensive semantic information) than later layers. In most cases, the final convolutional layer is seen as most informative, with [29] stating that "convolutional layers naturally retain spatial information which is lost in fully-connected layers, so we can expect the last convolutional layers to have the best compromise between high-level semantics and detailed spatial information". For this reason, I also use the final convolutional layer.

As mentioned previously, I do not apply any Rectified Linear Unit (ReLU) activation function after calculating the saliencies, despite this being an explicit part of the methodology of [29]. This is because information about what the model regards as detrimental to the prediction may also be informative for OOD detection.

3.5.4 Guided Backpropagation

GBP is a very simple method which is completely non-parametric, and as such no specific choices have had to be made. Like with all other methods, no normalization or rectification has been done to the output of GBP.

3.5.5 Integrated Gradients

For integrated gradients, the only choice that must be made is the choice of baseline. An ideal baseline should convey no information, and should ideally lead to an output of zero from the network. As [68] shows, there are no perfect choices for such a baseline, as all methods carry some assumption about our dataset and what it means for an input to "convey no information". However, the common choice for image classification tasks is the zero vector (a completely black image), as recommended by [33]. As such, this is the baseline I have used.

3.6 Evaluation

This section describes the details of how I have evaluated my methods on the four OpenOOD benchmarks.

3.6.1 Metrics

AUROC and FPR95 are the most common metrics used for OOD detection [7, 11, 16–18]. In OpenOOD [7], AUROC is chosen as the primary metric used to rank methods against each other, as mentioned previously. As [7] by far presents the most comprehensive complete benchmark of all OOD detection methods to date, I have followed their

methodology and used AUROC when evaluating my methods. Although FPR95 is reported quite often in OOD detection literature, I have chosen to omit this metric when reporting the results. Due to the large number of tests that have been performed as part of this thesis, this has been necessary to limit the amount of information conveyed in chapter 4, which would otherwise be extremely long. When comparing all OOD detection methods against each other, the FPR95 is similarly omitted by [6].

When performing OOD detection, we must make a choice about whether OOD samples belong to the positive or negative class. There is no correct answer, but [7] chooses to consider OOD samples as the positive class, to "align with ML convention": It is common to consider abnormalities, anomalies, or the unexpected as the positive class (for example, a cancer detection system would consider the presence of cancer to be part of the positive class). This aligns with the goal of OOD detection, which is to detect abnormal inputs with regard to the data the model is trained on. Thus, I have also followed this convention and considered OOD samples as positive, and ID samples as negative.

The AUROC calculations are done on each OOD dataset individually, as opposed to comparing all OOD samples to the ID dataset. Afterwards, the metrics of all Near-OOD and Far-OOD are averaged, giving us two general performance metrics which tell us how a method functions on either Near-OOD or Far-OOD. This aligns with the methodology of [7]. Similarly, when plotting density plots for a given OOD detection metric, the densities of all Near- and all Far-OOD data points are combined, giving a more comprehensible overview than if all OOD data sets are individually. Figure 3.6 shows density plots before and after combining Near- and Far-OOD, on the maximum logit score of images from the CIFAR10 dataset. As we can see, there is not that much internal variance between the OOD datasets, allowing us to combine them without losing too much information.

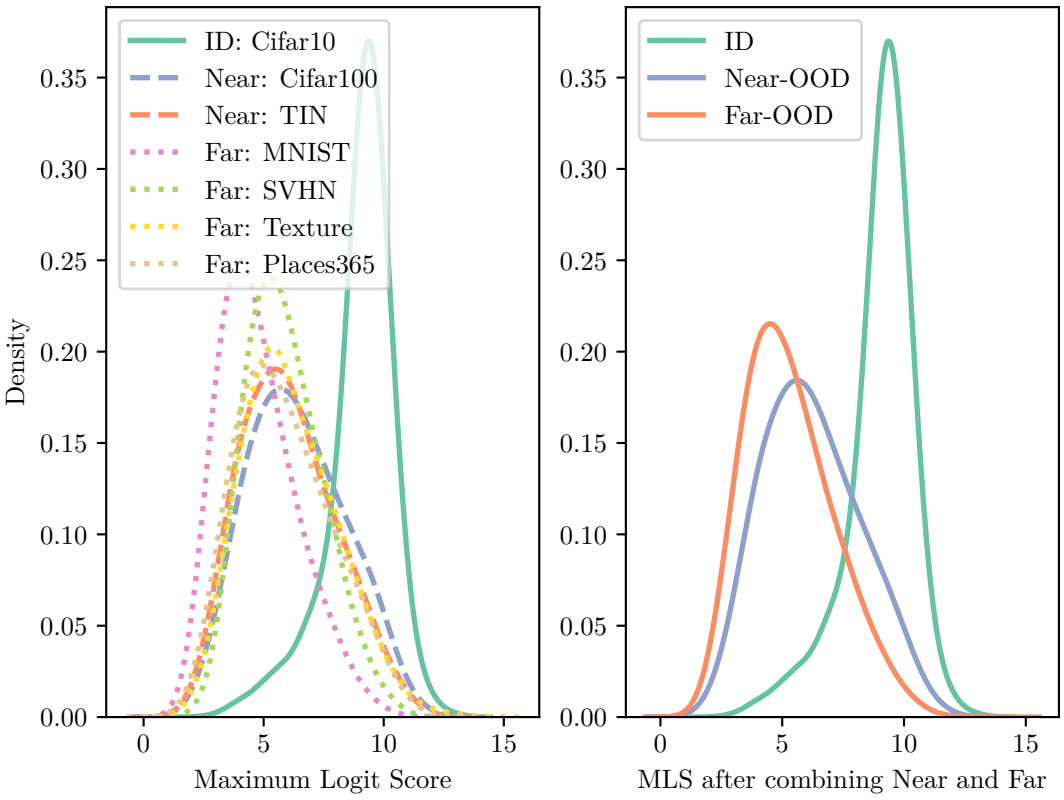

Figure 3.6: Density plots of MLS on CIFAR10 for all datasets individually and after combining Near- and Far-OOD

3.6.2 Statistical Analysis of Results

When evaluating a new method, it is not enough to simply report the results from a single experiment. Instead one should run the same experiment multiple times and perform statistical analysis to ensure that the results are robust. Therefore, I have bootstrapped each testing benchmark ten times during the calculation of the final test results. This makes it possible to report means and standard deviations of the results as opposed to a single point estimate. With these repeated experiments, it is also possible to perform statistical tests comparing the new methods against baseline OOD detection methods. If we compute the scores for the baseline methods and the new methods on the same bootstraps, we get paired samples. In this case, we should conduct a paired difference test. The t-test is a natural choice, however, we do not have any reason to assume that the differences in performance between paired samples is normally distributed, nor is the number of bootstraps high enough to assume approximate normality. The Wilcoxon signed-rank test is thus a better choice, and is the one I have used.

Given that I have designed my methods to be general frameworks for integrating XAI saliencies, it is possible to test the same method with many different XAI algorithms. Doing this, however, will introduce the *multiple comparison problem*, where a p-value of 0.05 will no longer be sufficient for each test, as the chance of randomly receiving results that seem statistically significant is increased by the presence of multiple tests. Thus,

I use Bonferroni corrected p-values whenever multiple tests are conducted on the same method. I set the threshold for statistical significance at 0.05, leading to a Bonferroni corrected p-value of $0.05/n$ for a given method, where n is the number of experiments conducted for a single method.

3.7 Implementation

This section goes over the implementation details of my thesis.

3.7.1 Basic hardware and software

Python is the most popular programming language for data science and ML research [69], and as such it is my language of choice as well. For this thesis, `Python 3.9` has been used. Below is a table of the key libraries that have been utilized.

Library	Version number	Short description
OpenOOD	1.5	Comprehensive OOD detection framework
PyTorch	2.4.1	GPU-accelerated ML library
Captum	0.7.1	XAI methods integrated with PyTorch
Scikit-Learn	1.5.2	Various ML methods
NumPy	1.26.4	Efficient matrix multiplication and scientific computing
Matplotlib	3.9.2	Visualization library
Seaborn	0.13.2	Visualization library built on top of Matplotlib
Pandas	2.2.3	Data visualization and manipulation library
SciPy	1.13.1	Scientific computing library

All development and computation was done on a single computer with an `Intel i7-8700K` CPU and an `Nvidia GeForce RTX 3090` GPU.

3.7.2 Method Evaluation: OpenOOD

As explained previously, OpenOOD [7] represents the most comprehensive benchmark for OOD detection methods. It is also a framework which easily allows for development and benchmarking of new methods, and is thus the ideal framework for the purposes of this thesis.

In particular, OpenOOD includes a "unified, easy to use evaluator" [6] that makes evaluating new methods very simple. All that is required is that new methods inherit from a base class (`BasePostprocessor`[4]), and override the calculation of OOD scores. Code listing 3.1 shows all the code required to create the Aggregate of Saliency OOD detector.

```
class SaliencyAggregatorPostprocessor(BasePostprocessor):
    def __init__(self, config, saliency_generator, aggregator):
        super().__init__(config)

        self.saliency_generator = saliency_generator
        self.aggregator = aggregator

    def postprocess(self, net: nn.Module, data: Any):
```

[4]In OpenOOD, `Postprocessors` are the OOD detection algorithms that generate an OOD score during inference

```
        predictions = torch.argmax(net(data), dim=-1)

        saliencies = self.saliency_generator(net, data)
        score_ood = self.aggregator(saliencies, dim=-1)

        return predictions, score_ood
```

Listing 3.1: Source code listing for the Aggregate of Saliency OOD detector

With this `postprocessor` defined, evaluating it on a specific benchmark is similarly simple (listing 3.2):

```
resnet18_pretrained = get_network('cifar10')

ood_detector = SaliencyAggregatorPostprocessor(None, GradCAM, torch.mean)

evaluator = Evaluator(
    net=resnet18_pretrained,
    id_name='cifar10',
    postprocessor=ood_detector,
)

metrics = evaluator.eval_ood()

print(metrics)
```

Listing 3.2: Source code listing for evaluating methods within the OpenOOD framework

This code will calculate the OOD scores for all data samples in both the ID and OOD datasets, and subsequently calculate the AUROC and FPR95 for all the OOD datasets when comparing their OOD values to the values of the ID datasets. Code listing 3.3 shows this output when using the baseline MSP method.

```
           FPR@95   AUROC   AUPR_IN   AUPR_OUT    ACC
cifar100    61.36   86.51     84.20      85.05  95.56
tin         42.02   88.88     88.81      85.57  95.56
nearood     51.69   87.69     86.51      85.31  95.56 # average of two above
mnist       19.38   93.86     79.72      98.89  95.56
svhn        24.78   91.38     84.26      95.49  95.56
texture     43.31   88.68     91.01      80.97  95.56
places365   41.62   89.21     68.49      96.28  95.56
farood      32.27   90.78     80.87      92.91  95.56 # average of four above
```

Listing 3.3: Output of calling `evaluator.eval_ood` with CIFAR10 as the benchmark and MSP as the detector

As we can see, OpenOOD allows for very easy evaluation of new methods. Furthermore, it allows for easy comparisons between methods, one of the stated goals of the framework [7]. This makes it an ideal framework for this thesis.

Modifications to OpenOOD

As explained previously, continually testing new methods on the same benchmarks will bias the final results. As of the writing of this thesis, there are no functionalities in OpenOOD which allow for creating development or testing benchmarks; all evaluations are done with entire datasets. For my purposes, which involve continuous exploration of different methods and careful inspection of the datasets, this is inadequate. Thus, I have made modifications to the `Evaluator` class such that it takes a `data_split` parameter

during initialization, and have also modified the function `get_id_ood_dataloader` to accept this parameter and return the correct split accordingly.

Furthermore, there is no functionality for sampling from the datasets as opposed to using them as they are, which is necessary to perform bootstrapping and calculate the statistical significance of the results. This has been done by passing a seed to the `Evaluator` class, which, if defined, will be used to seed a random sampling operation on the datasets. By instantiating several `Evaluator` classes with different seeds, we can bootstrap the evaluation and perform statistical analysis on the results.

3.7.3 Implementation of Saliency Methods

There are many libraries which implement XAI methods, such as [25, 70, 71]. When these implementations were suitable for my purposes, I have used them. However, given that I am not using these explanations for their original purpose (elucidating why a model came to a specific decision), there are many cases where the current implementations are inadequate. The two main problems are lack of access to the raw saliency values and slow speeds.

LIME

LIME has an implementation for Python, written by the original authors [25]. However, this implementation is not suitable for my purposes. The main issue is that it is far too slow, being implemented with `NumPy` which restricts the computation to the CPU. Furthermore, [25] also returns full size heatmaps, although the actual number of saliency values used to create this heatmap is far smaller than the number of pixels in the image. Thus I have implemented LIME myself using PyTorch.

Occlusion

Captum [70] is a library of XAI methods implemented in PyTorch, and this library contains a suitable implementation of occlusion. As explained previously, occlusion occludes parts of the image and compares the prediction scores before and after the occlusion. By occluding all parts of the image, we can get a saliency value for all positions. Occlusion is usually done using a sliding window, similar to a convolutional kernel, which is slid over all parts of the image. Such a window is rarely a single pixel, because it is often not interesting to see how a single pixel contributes to a prediction, but rather a larger region. What this means is that the final heatmap, although it is the same size as the input image, actually contains far fewer unique values. To avoid performing computations on thousands of repeated values, I reduce the size of the heatmap by sampling one pixel from each of the positions the sliding window has been applied, which can be efficiently done using a 1×1 MaxPooling kernel with a stride equal to the stride used during the occlusion.

GradCAM

GradCAM has been implemented from scratch in `PyTorch`. There are several libraries which implement GradCAM [70, 71]. However, given that these libraries are concerned with simply producing heatmaps that users can inspect, they do not output the raw saliency values, but upscale the saliencies to match the input image dimensions. As explained in 2.4.4, GradCAM uses the final feature map to generate an explanation, which usually has a spatial dimension which is far smaller than the input image (e.g. 7×7

versus 224×224). When overlaying these values on the input image, it is common to use bilinear interpolation [71], which interpolates all 224×224 positions based on the original 7×7 saliency map. For visualizations, this is reasonable. When attempting to use this data to separate ID and OOD data however, this is undesirable. Bilinear interpolation introduces new values, which changes many statistical qualities of the saliency values. This may reduce the separability of different samples. Furthermore, given that these new values do not add any new information, it is inefficient to involve these upscaled saliency maps in any computational operation.

Although it is relatively simple to modify the source code of these libraries to remove the interpolation, GradCAM is not very difficult to implement, and as such I have simply used my own implementation.

Integrated Gradients and Guided Backpropagation

Captum also contains a suitable implementation for integrated gradients and guided backpropagation, which I have utilized in essentially unmodified forms. By definition (see section 2.4.4 and 2.4.4), these methods return saliency maps over the entire input image dimensions. This separates them from the other methods, which return a far lower number of distinct saliencies, which are upscaled or transformed to the entire image (LIME and Occlusion output the same values for all pixels within a segment, GradCAM outputs an amount of saliencies corresponding to the final feature map, which is then upscaled). Because of this, the saliencies returned are exactly equivalent to the raw values I require, and no modifications are necessary.

However, this lack of dimensionality reduction also poses a technical challenge when we wish to store these saliency maps for later data analysis. For example, storing all saliencies, without dimensionality reduction, for ImageNet200 and its associated OOD datasets would require 28 GB. Thus, instead of storing the saliencies themselves, I calculate the aggregate values during saliency generation and store them instead. This has the downside that if a new form of aggregation is to be tested, the whole dataset of saliencies has to be generated again. However, the decrease in storage requirements is immense, being between a 1000 and 10 000 times decrease depending on the number of aggregates stored. This is strictly an implementation detail and has no effect on any analysis or evaluation of methods.

3.8 Summary

In this chapter, I have introduced the methodology that has been used in this thesis. I have described the three proposed XAI OOD detection frameworks that will be evaluated in chapter 4. In addition, I have described the four OpenOOD benchmarks and the datasets which constitute them, as well as the testing environment and the choice of evaluation metrics. Finally, I have described the implementation details of my work.

Chapter 4

Experiments and Results

This chapter contains the results of experiments as described in chapter 3. The chapter is split into two main sections: In section 4.1, I show how different forms of aggregation over saliency maps separate ID and OOD data points. This step is important, because it informs the choice of aggregation and XAI saliency mapping methods used in the Saliency Aggregation and Saliency Aggregation plus Logit OOD detection frameworks. I report AUROC for different forms of aggregation over CIFAR10 and ImageNet200. This is done on the validation sets, to avoid biasing the final results.

Section 4.2 contains the final results of applying methods from the three frameworks introduced in section 3.1 on the testing benchmarks. For Saliency Aggregation and Saliency Aggregation plus Logit, the choice of aggregation and XAI method is informed by the findings from section 4.1. For SaliencyVIM, the testing is done using the three XAI methods which are compatible with the methodology as described in section 3.1.2. These are LIME, occlusion and GradCAM. The AUROC for each method is calculated over ten bootstraps for each benchmark, enabling statistical analyses which investigate whether XAI OOD detection methods outperform baseline methods. For each framework, the results are first presented for all four benchmarks in an objective manner, after which the performance of the framework is analyzed and discussed.

4.1 Data Analysis of Saliency Maps

This section will detail how various XAI methods generate explanations which differ between ID and OOD. The section considers each validation benchmark individually. For each benchmark, I first present the level of separation achieved by the two baseline methods MSP and MLS, to give a general intuition about how easily the ID and OOD datasets are to separate. Following this, I go through the different XAI methods and their different statistical qualities.

4.1.1 ImageNet200

The ImageNet200 benchmark is a suitable benchmark to begin with, given ImageNet's ubiquity in AI research. Figure 4.1 shows the distribution of the maximum softmax score and the maximum logit. Here, we see that there is a decent amount of separation between ID and OOD data points, even with the simple baseline methods introduced by [16] and [42]. Separating the distributions using MSP, we get an AUROC score of 0.834 for Near-OOD and 0.915 for Far-OOD. Using MLS, we get an AUROC score of 0.833 for Near-OOD and 0.903 for Far-OOD.

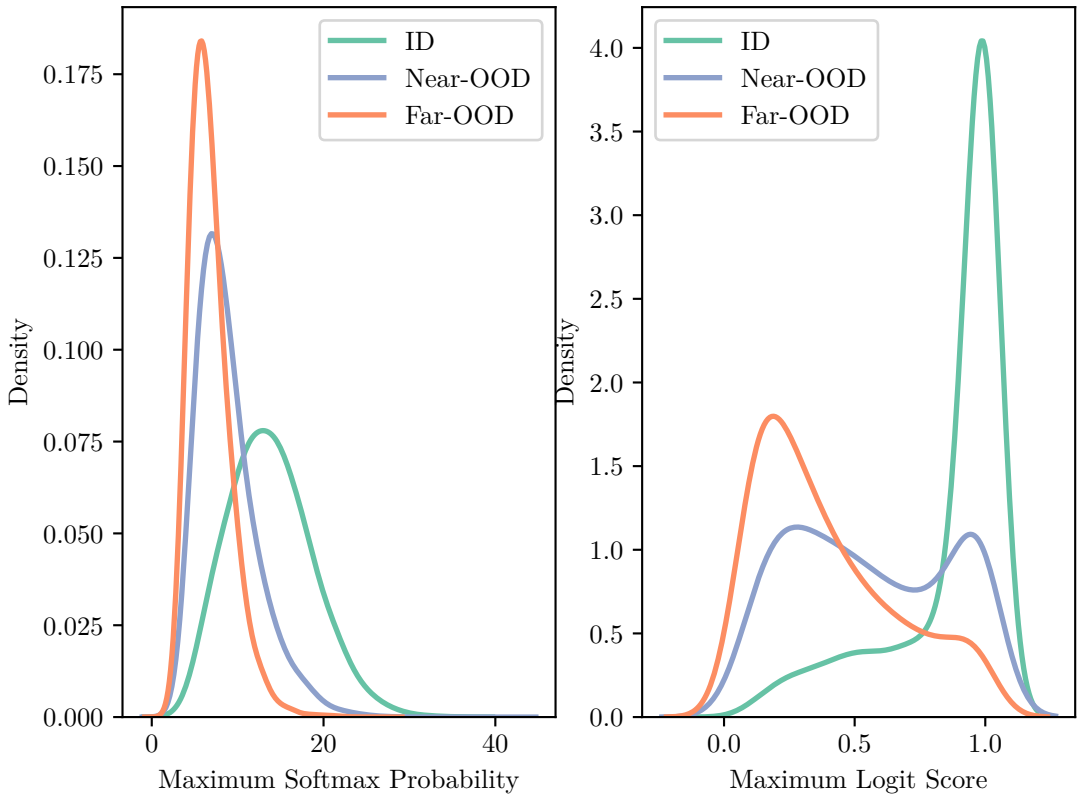

Figure 4.1: Density plot of the maximum softmax probability and maximum logit score on ImageNet200

LIME

With the baselines reported, we turn our attention to the first XAI saliency method, LIME. Figure 4.2 presents graphs representing the two main forms of aggregation that has been applied; those which consider the magnitude of the saliencies and those which consider the statistical spread. These two forms are here represented by the vector norm and the RMD.

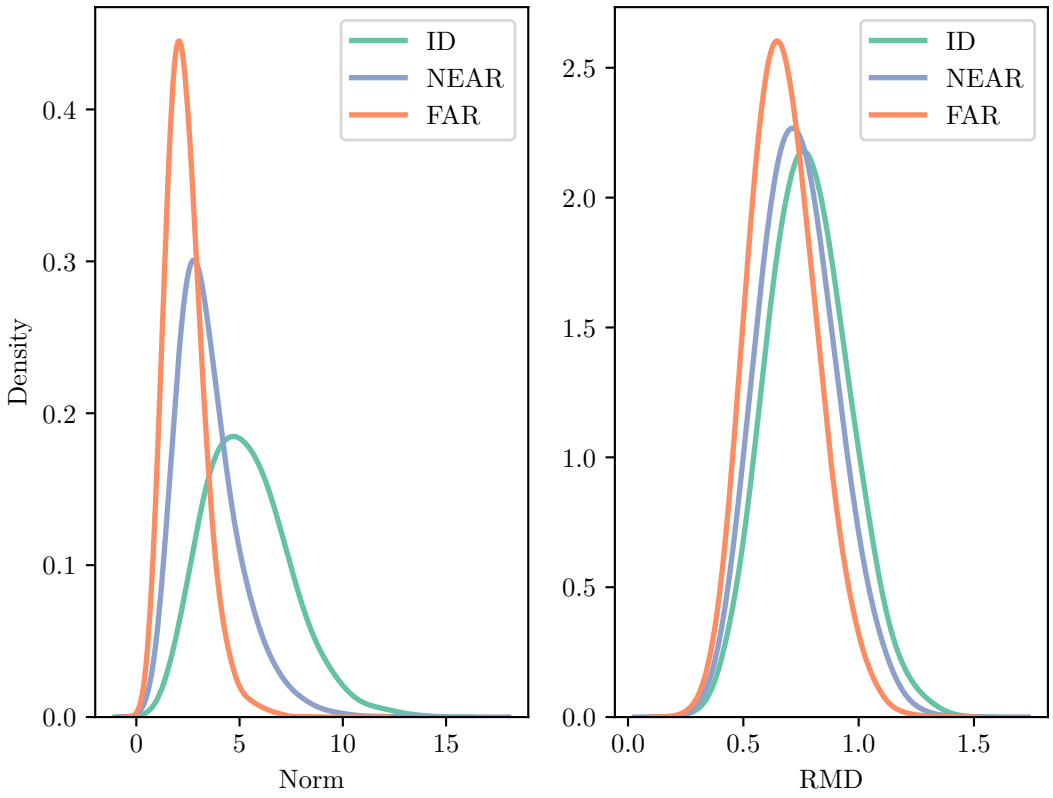

Figure 4.2: Vector norm and RMD density plots for LIME on ImageNet200

From the figure we can see that for LIME on ImageNet200, it is indeed the case that saliencies are higher for ID data points than for OOD data points. Using the vector norm to distinguish ID and OOD, we get an AUROC of 0.814 on Near-OOD, which is slightly lower than the baseline methods, and 0.925 on Far-OOD, which is actually higher than the baselines. These results are far better than those attained by [9], the only other related work which has attempted to use saliency maps to separate ID and OOD data. This work achieved an AUROC score of only 0.52 when used on benchmark which did not just contain small toy datasets, which is for all practical purposes equivalent to guessing. This large improvement suggests that the usage of raw saliency values, rather than normalized heatmaps (as was used by [9]) could be highly consequential for the performance of XAI OOD detection algorithms.

Measuring the statistical spread using RMD, we find that there is indeed also a higher spread in ID data when compared to OOD data. However, in this case the overlap is substantial, which is reflected in the AUROC scores attained when discriminating using RMD: The Near-OOD AUROC score was 0.594, and the Far-OOD score was 0.695. In both cases, the scores attained are far lower than the both of the baselines.

Table 4.1 shows the AUROC scores for all forms of aggregation on LIME, as well as the previously mentioned AUROC of the baseline methods. The AUROC scores are reported as percentages, to avoid having a redundant leading zero in all cells. In addition to the AUROC, the correlation between the aggregates and the maximum logit

and maximum softmax score is reported, which gives insight into how XAI saliency maps are related to the prediction confidence of the model.

Aggregation type	Baselines		Magnitude of saliencies						Statistical dispersion		
Aggregate	MLS	MSP	Mean	Median	Norm	Range	Max	Q3	CV↓	RMD	QCD↓
Near-OOD AUROC	83.3	83.4	78.8	69.3	**81.4**	77.1	78.1	78.0	53.7	59.4	53.2
Far-OOD AUROC	91.5	90.3	88.1	75.2	**92.5**	90.2	91.4	87.4	49.3	69.5	49.8
Correlation with MLS	-	-	0.75	0.60	0.71	0.45	0.54	0.72	-0.01	0.08	-0.01
Correlation with MSP	-	-	0.61	0.48	0.61	0.41	0.48	0.60	-0.01	0.09	-0.01

Table 4.1: AUROC scores for LIME on ImageNet200. The highest non-baseline value for Near- and Far-OOD is highlighted in bold. ↓ denotes that ID data points more often have a lower score with this aggregation, and thus the output values have been negated (as described in section 2.3.2)

From this table, we can note several interesting observations. Firstly, we see that the magnitude based aggregations in general perform quite well, with vector norm in this case being the best. In contrast, the methods based on statistical dispersion perform poorly, putting into question the hypothesis that the saliency maps of ID data is more concentrated and less spread out than those on OOD data. Indeed, we see that while the RMD is higher on average for ID data, the QCD and CV is lower on average when generating saliencies using LIME. This further puts doubt on the idea that the spread of saliency maps is a reliable indicator of OOD-ness.

Secondly, the aggregates which capture information about the magnitude of the saliencies are highly correlated with both the maximum logit and the maximum softmax score of the predicted class. This is not unexpected, as LIME generates saliencies using differences in prediction values as different parts of the image is masked. If the predicted value is higher on average for ID data, then it is likely that the drop in predicted value when masking parts of the image is larger as well, leading to higher saliencies. Regardless, it seems clear that it is not only the correlation to the model output which explains the discriminative power of these aggregates, as we see that vector norm aggregation has lower correlation with MLS and equal correlation with MSP when compared to the mean, but has higher scores.

In general, the results from these aggregations are promising, and show that there is definite potential for OOD detection algorithms based on XAI outputs. However, the reader should note that these results are done on the validation set, and that no statistical tests have been done at this point. The statistical significance of using XAI saliency maps for OOD detection will be revealed in section 4.2, when the final testing is done on the testing benchmarks and bootstrapping is performed.

Occlusion

Now, we turn our attention to occlusion saliency mapping. Looking at figure 4.3, we see that there is far more overlap between the ID and OOD densities than with LIME.

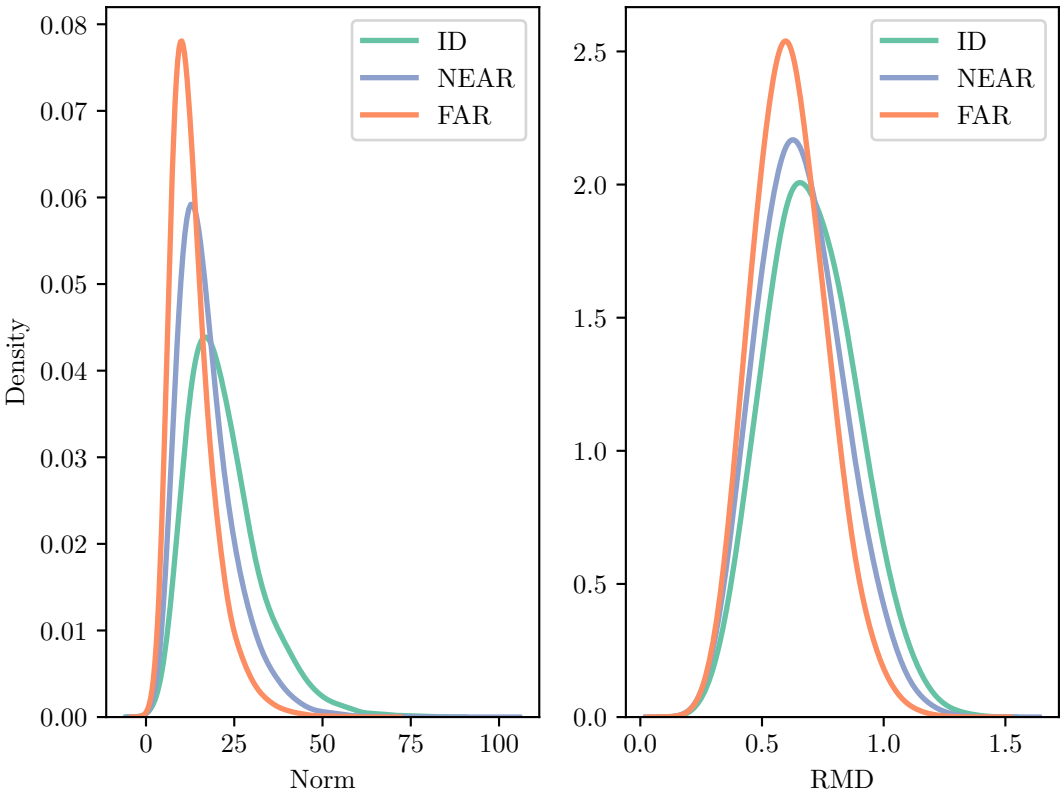

Figure 4.3: Density plots of Norm and RMD for occlusion on ImageNet200

From table 4.8 we see that this trend is apparent over all of the different forms of aggregation, not just the vector norm and RMD. Regardless, there is still a clear trend ID saliency magnitudes being higher than OOD magnitudes, as all AUROC scores are above 0.50 without negation. With this saliency method, aggregating using range performs the best, with max relatively close behind. Interestingly, we see that for occlusion, all the statistical dispersion aggregates are higher for ID data, as was the original hypothesis. Regardless, the AUROC scores are very poor, and thus these metrics do not seem suitable to discriminate between ID and OOD.

Aggregation type	Baselines		Magnitude of saliencies						Statistical dispersion		
Aggregate	MLS	MSP	Mean	Median	Norm	Range	Max	Q3	CV	RMD	QCD
Near-OOD AUROC	83.3	83.4	58.6	54.2	65.7	**69.1**	66.8	60.4	55.0	57.7	55.4
Far-OOD AUROC	91.5	90.3	69.7	62.9	77.8	**84.6**	83.5	72.3	63.8	64.7	62.7
Correlation with MLS	-	-	0.37	0.31	0.42	0.44	0.47	0.41	0.01	0.14	0.01
Correlation with MSP	-	-	0.29	0.24	0.37	0.41	0.41	0.34	0.01	0.13	0.01

Table 4.2: AUROC scores for Occlusion on ImageNet200. The highest non-baseline value for Near- and Far-OOD is highlighted in bold. ↓ denotes that ID data points more often have a lower score with this aggregation, and thus the output values have been negated (as described in section 2.3.2)

GradCAM

After having inspected two completely model independent XAI methods, we turn our attention to the first of the three gradient based methods, GradCAM. As we have seen density plots for both LIME and occlusion, I will omit these here, as the table contains the necessary information. From table 4.9, we see that vector norm aggregation of GradCAM saliencies actually beats the baselines on both Near- and Far-OOD detection. The increase on Far-OOD is particularly impressive, with an increase of 1.4 percentage points. However, we should keep in mind that these results are done on the validation set and that the final results and their statistical significance will only be explored in section 4.2.

Aggregation type	Baselines		Magnitude of saliencies						Statistical dispersion		
Aggregate	MLS	MSP	Mean	Median	Norm	Range	Max	Q3	CV	RMD	QCD
Near-OOD AUROC	83.3	83.4	83.3	80.3	**83.8**	80.5	81.9	83.5	50.8	51.8	51.7
Far-OOD AUROC	91.5	90.3	91.5	87.6	**92.9**	92.2	92.8	92.7	63.6	64.9	64.8
Correlation with MLS	-	-	1.00	0.94	0.97	0.69	0.81	0.96	-0.16	-0.12	-0.12
Correlation with MSP	-	-	0.83	0.78	0.82	0.62	0.70	0.81	-0.11	-0.07	-0.07

Table 4.3: AUROC scores for GradCAM on ImageNet200. The highest non-baseline value for Near- and Far-OOD is highlighted in bold. ↓ denotes that ID data points more often have a lower score with this aggregation, and thus the output values have been negated (as described in section 2.3.2)

As was expected, the mean value of the GradCAM saliencies is completely correlated with the maximum logit. As proven in section 3.2, performing mean aggregation on GradCAM saliency maps performed on the final convolutional layer of a ResNet network

is indeed equivalent to MLS OOD detection, hence the total correlation. In general, the correlations with the baseline scores are far higher here than with the previous two XAI methods.

Integrated Gradients

Looking at table 4.4, we see that the trend of larger saliency magnitudes on ID data continues to hold for integrated gradients as well. With integrated gradients the mean and the vector norm seem to be the most discriminative, with the mean saliency having scores which are around 1 percentage point below the baselines.

Aggregation type	Baselines		Magnitude of saliencies						Statistical dispersion		
Aggregate	MLS	MSP	Mean	Median	Norm	Range	Max	Q3	CV↓	RMD	QCD
Near-OOD AUROC	83.3	83.4	**82.1**	55.3	67.3	63.9	63.5	64.1	66.1	51.3	50.5
Far-OOD AUROC	91.5	90.3	**90.5**	56.0	87.8	86.7	85.9	79.6	49.8	39.1	53.4
Correlation with MLS	-	-	0.94	0.14	0.40	0.31	0.30	0.35	-0.16	0.01	0.01
Correlation with MSP	-	-	0.79	0.10	0.36	0.29	0.28	0.31	-0.15	0.00	0.00

Table 4.4: AUROC scores for IntegratedGradients on ImageNet200. The highest non-baseline value for Near- and Far-OOD is highlighted in bold. ↓ denotes that ID data points more often have a lower score with this aggregation, and thus the output values have been negated (as described in section 2.3.2)

GBP

We now turn our attention to the final XAI saliency method, GBP. Here we see something interesting; as the only method on ImageNet200, GBP actually has two magnitude metrics where OOD data points have higher values than ID data points. Looking at table 4.5, we see that both the mean and median saliency is lower on ID data. Moreover, the mean is considerably lower, with an AUROC of 0.737 on Near-OOD and 0.817 on Far-OOD when choosing lower values as ID as opposed to higher. The vector norm, range and maximum aggregations are still higher for ID data. These aggregations are the ones which are not affected by large negative values. The conclusion to draw from these results is as follows: For GBP, ID XAI saliency maps exhibit higher magnitudes, but they are not restricted to positive attributions.

Aggregation type	Baselines		Magnitude of saliencies						Statistical dispersion		
Aggregate	MLS	MSP	Mean↓	Median↓	Norm	Range	Max	Q3	CV	RMD↓	QCD↓
Near-OOD AUROC	83.3	83.4	73.7	55.7	**77.4**	71.3	71.8	71.0	50.2	52.8	51.5
Far-OOD AUROC	91.5	90.3	81.7	51.1	**92.3**	90.7	90.6	68.7	43.5	72.3	50.7
Correlation with MLS	-	-	-0.33	-0.08	0.45	0.25	0.25	0.36	0.00	-0.01	0.01
Correlation with MSP	-	-	-0.30	-0.08	0.42	0.25	0.24	0.35	0.00	-0.00	0.01

Table 4.5: AUROC scores for GBP on ImageNet200. The highest non-baseline value for Near- and Far-OOD is highlighted in bold. ↓ denotes that ID data points more often have a lower score with this aggregation, and thus the output values have been negated (as described in section 2.3.2)

Looking at figure 4.4, we see that the mean saliency value is indeed lower for ID data when compared to all the OOD data sets.

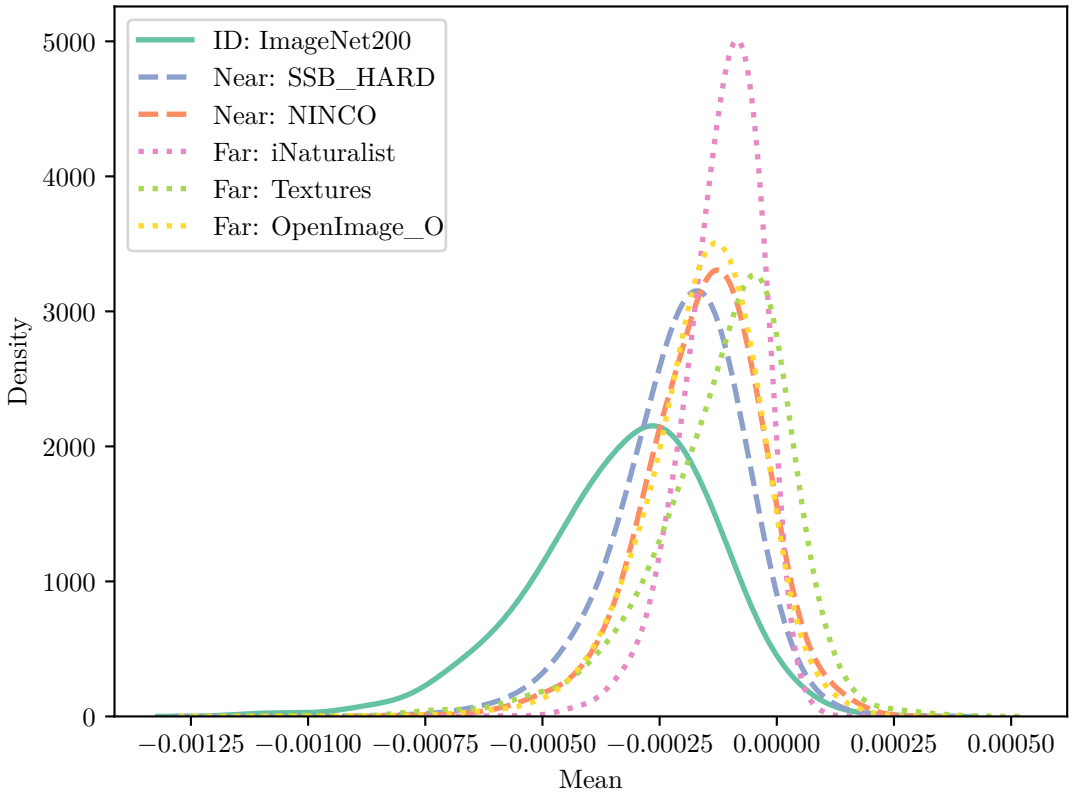

Figure 4.4: Density plot of the mean saliency for GBP for all datasets in the ImageNet200 benchmark

Overall results on ImageNet200

Next, we consider the overall performance of the different aggregations and XAI saliency methods on ImageNet200. Figure 4.5 shows the average Near- and Far-AUROC for magnitude aggregations. From this, we see that the vector norm, range and maximum aggregations performed the best over all XAI saliency methods. All statistical dispersion methods performed poorly, and as such I do not plot them. In general, saliency aggregation methods seem to perform far above the heatmap clustering performed by [9], at least on these validation benchmarks.

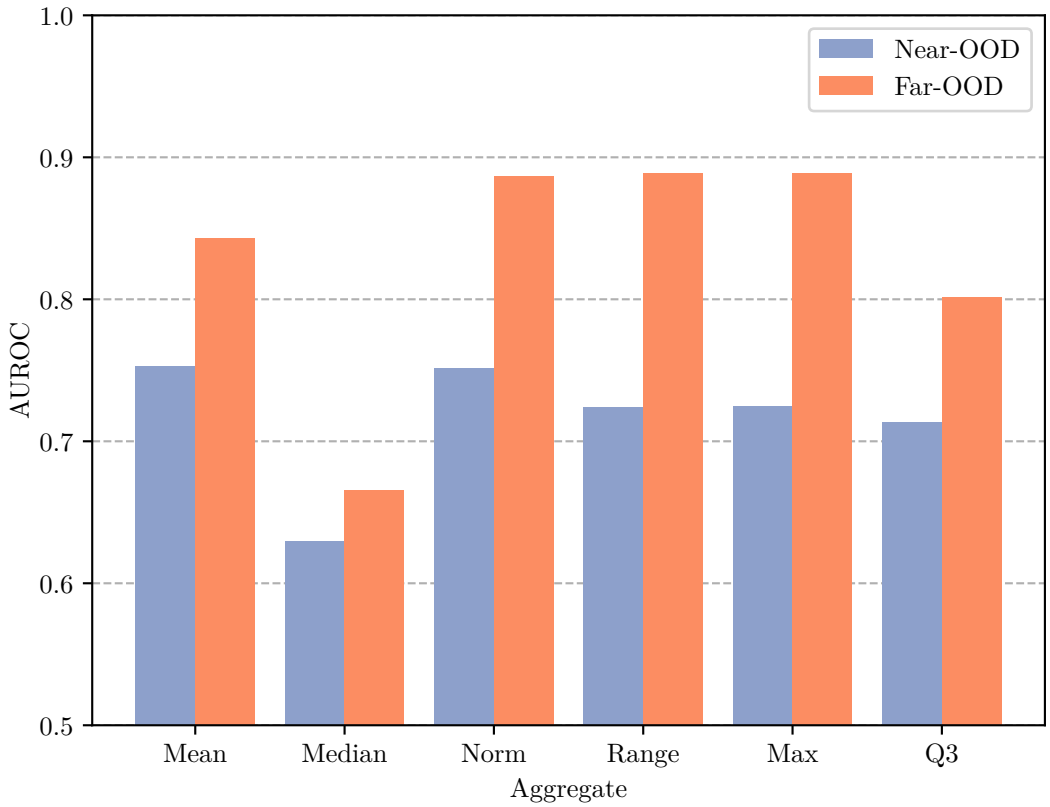

Figure 4.5: Barplot of average AUROC scores for each metric on ImageNet200. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

Table 4.6 shows the underlying data for the figure above, as well as the mean AUROC for both Near and Far. Here, we can see that the mean performed the best on Near-OOD while the range performed the best on Far-OOD. However, as we know, the mean was lower for GBP and lower scores had to be considered ID, which makes it less desirable than the other methods, which were consistently larger for ID data. The vector norm was only slightly worse than the mean on Near-OOD and only slightly worse than the range on Far-OOD, making it the best aggregation overall. As we can see, all the statistical dispersion methods performed poorly.

Aggregation type	Baselines		Magnitude of saliencies						Statistical dispersion		
Aggregate	MLS	MSP	Mean	Median	Norm	Range	Max	Q3	CV	RMD	QCD
Near-OOD AUROC	83.3	83.4	**75.3**	63.0	75.1	72.4	72.4	71.4	55.2	54.6	52.5
Far-OOD AUROC	91.5	90.3	84.3	66.6	88.6	**88.9**	88.8	80.1	54.0	62.1	56.3
Mean AUROC	87.4	86.9	79.8	64.8	**81.9**	80.6	80.6	75.7	54.6	58.4	54.4

Table 4.6: Average AUROC scores for all XAI saliency methods on ImageNet200. The highest non-baseline value for Near- and Far-OOD is highlighted in bold. ↓ denotes that ID data points more often have a lower score with this aggregation, and thus the output values have been negated (as described in section 2.3.2)

If we instead look at the best aggregation for each XAI saliency method (figure 4.6), and compare these to the baselines, we find that LIME, GradCAM and GBP achieve higher degrees of separation than the baselines on Far-OOD, while GradCAM is the only method who has a higher degree of separation than the baselines on Near-OOD. Except for occlusion, all XAI saliency methods are able to discriminate between ID and OOD reasonably well, given a correct choice of aggregation.

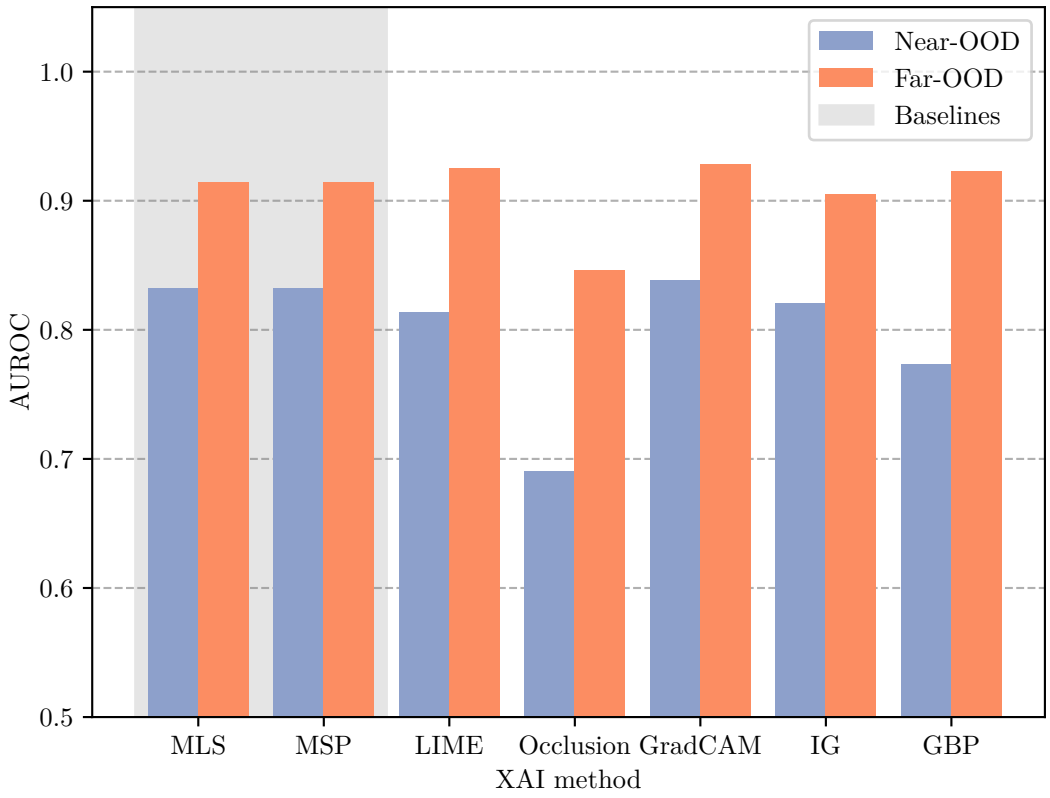

Figure 4.6: Barplot of the highest AUROC scores per XAI saliency method on ImageNet200. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

Finally, figure 4.7 shows a heatmap of the average performance over both Near-OOD and Far-OOD for all combinations of aggregation and XAI saliency method. From this we can get a visual overview over what combinations achieve the highest degree of separation between ID and OOD.

Figure 4.7: Heatmap of overall performance AUROC performance for all combinations of XAI methods and magnitude aggregations

From this, we can see that GradCAM separates ID and OOD well overall (likely due to its high correlation with MLS) while occlusion performs poorly, as we saw previously. Apart from this, we see that some promising combinations are LIME+Norm, GBP+Norm and IG+Mean.

The results above demonstrate that saliency maps themselves, without any other information from the network, seem to be able to adequately separate ID from OOD data points, contrary to what was found by [9]. In the next section, I will explore whether these results hold on CIFAR10 as well. After these tests, we will have an informed opinion about what XAI saliency methods and aggregations best separate the data on the validation set, and we can choose a selection of combinations and gather final results on the testing benchmarks.

4.1.2 CIFAR10

CIFAR10 differs from ImageNet200 in some important respects, making it an ideal second benchmark to investigate. While ImageNet200 has 200 classes, CIFAR10 has only 10. This means that ImageNet200 is a much broader classification task, with networks that are trained to recognize a wide variety of different objects. With a more narrow ID dataset of only 10 classes, we may see different behaviour when a network is exposed to OOD data. Another important change is the size of the images. ImageNet200

images are 224×224 pixels, while CIFAR10 images are only 32×32. This reduction in resolution may also affect how saliency maps are generated, as each pixel now covers a much larger area of the total image and is thus more important compared to each pixel in ImageNet200 images. First, let us inspect the baseline performances.

From figure 4.8, we can see that there is a decent degree of separation when using the maximum logit and the maximum softmax score. The maximum softmax score is highly concentrated around 0.95-1.00 for ID data, to a much higher degree than with ImageNet200. This is not surprising, given the much smaller number of classes in CIFAR10. With fewer and more easily separated classes, it is much easier for the network to saturate the maximum softmax score.

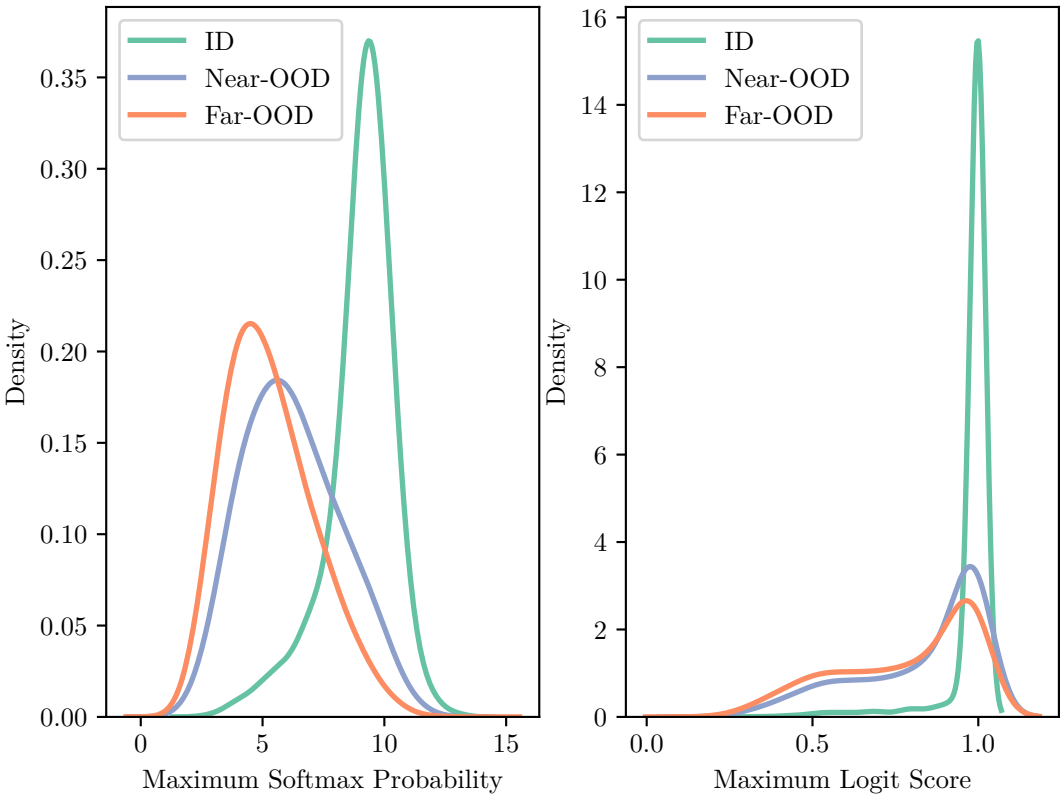

Figure 4.8: Density plot of the maximum logit and softmax score on CIFAR10

Separating the distributions using MSP, we get an AUROC score of 0.877 for Near-OOD and 0.908 for Far-OOD. Using MLS, we get an AUROC score of 0.867 for Near-OOD and 0.914 for Far-OOD. With the baselines reported, we turn our attention to the first XAI saliency method, LIME.

LIME

Table 4.7 shows the results for the aggregations on the saliency maps generated by LIME. From this table, we see a similar trend as when we applied LIME to ImageNet200: the magnitude of saliency methods all show a clear trend of higher values on ID data, while

the statistical dispersion methods are poor and uninformative. However, the degree of separation compared to the baselines is worse on CIFAR10, with over 5 percentage points lower scores on both Near- and Far-OOD. This is far worse when considering that LIME actually achieved a better AUROC than the baselines on ImageNet200 when using vector norm aggregation. It may be that because of the lower resolution of the images, each occluded region used to generate LIME explanations carries less information, which introduces some instability in when generating explanations.

Aggregation type	Baselines		Magnitude of saliencies						Statistical dispersion		
Aggregate	MLS	MSP	Mean	Median	Norm	Range	Max	Q3	CV↓	RMD	QCD↓
Near-OOD AUROC	86.7	87.7	**81.2**	73.0	77.7	67.1	76.1	76.6	63.6	61.0	59.0
Far-OOD AUROC	91.4	90.8	**86.1**	77.9	84.0	74.5	81.9	83.9	59.0	61.2	53.2
Correlation with MLS	-	-	0.40	0.28	0.34	0.22	0.31	0.37	-0.00	0.11	-0.00
Correlation with MSP	-	-	0.29	0.20	0.23	0.15	0.22	0.26	0.01	0.08	-0.00

Table 4.7: AUROC scores for LIME on CIFAR10. The highest non-baseline value for Near- and Far-OOD is highlighted in bold. ↓ denotes that ID data points more often have a lower score with this aggregation, and thus the output values have been negated (as described in section 2.3.2)

Regardless of the lower scores when compared to ImageNet200, the scores still show that the magnitudes of XAI saliency maps differ between ID and OOD data when using LIME to generate them.

Occlusion

Table 4.8 shows the results of using occlusion to generate saliency maps. Here we see something quite surprising: all magnitude metrics are lower for ID data. This result forces us to reconsider the theory that the magnitude of saliencies is higher on average for ID data, as we see a complete inversion of the results we got when using occlusion on ImageNet200.

Aggregation type	Baselines		Magnitude of saliencies						Statistical dispersion		
Aggregate	MLS	MSP	Mean↓	Median↓	Norm↓	Range↓	Max↓	Q3↓	CV	RMD	QCD↓
Near-OOD AUROC	86.7	87.7	63.4	63.1	**76.6**	74.2	67.3	75.5	52.4	54.7	50.3
Far-OOD AUROC	91.4	90.8	61.4	63.1	**74.6**	71.6	63.8	74.6	51.8	57.2	51.4
Correlation with MLS	-	-	0.22	0.11	0.13	0.05	0.27	0.13	0.00	0.34	0.00
Correlation with MSP	-	-	0.22	0.12	0.14	0.07	0.25	0.15	0.00	0.26	0.00

Table 4.8: AUROC scores for Occlusion on CIFAR10. The highest non-baseline value for Near- and Far-OOD is highlighted in bold. ↓ denotes that ID data points more often have a lower score with this aggregation, and thus the output values have been negated (as described in section 2.3.2)

Looking at the distribution for the vector norm (figure 4.9), we find a consistently lower ID value when compared to all OOD datasets. Essentially the same plot can be seen with the other magnitude metrics as well.

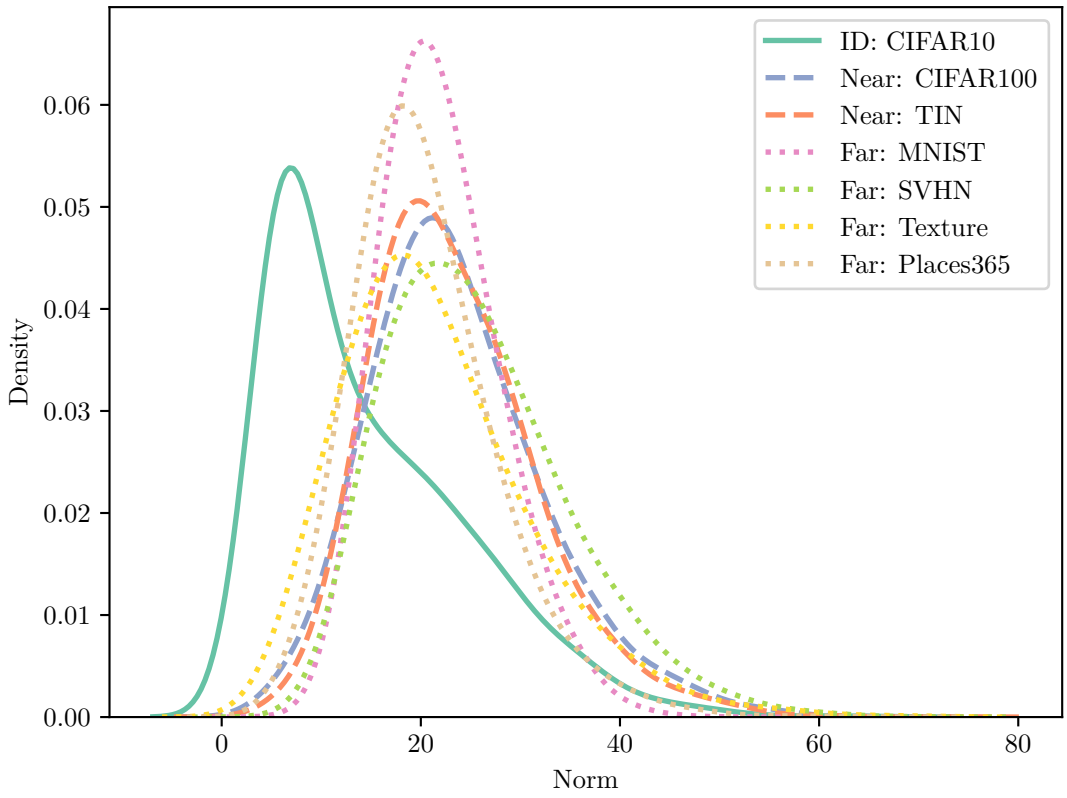

Figure 4.9: Density plot of the vector norm of occlusion saliencies for all datasets in the CIFAR10 benchmark

These results are very interesting, as they show that there is no guarantee that XAI saliency maps will be of higher magnitude on ID data. Even more interesting is the fact that when they are lower, they are lower across all OOD datasets and still allow for separation between ID and OOD by considering low saliency magnitudes as ID as opposed to high saliency magnitudes. These results concur with the second hypothetical scenario given in section 3.1.1.

GradCAM

Saliency maps made from GradCAM, with their mathematical equivalence to the maximum logit when applying the mean, unsurprisingly do not suffer from the same problems as occlusion. As we can see from table 4.9, the results for mean are equivalent to the maximum logit.[1]

[1]The 0.1 difference between MLS and mean aggregation on Near-OOD is most likely due to floating point imprecision

Aggregation type	Baselines		Magnitude of saliencies						Statistical dispersion		
Aggregate	MLS	MSP	Mean	Median	Norm	Range	Max	Q3	CV↓	RMD↓	QCD↓
Near-OOD AUROC	86.7	87.7	**86.8**	85.9	86.7	71.8	86.5	85.2	75.3	76.7	74.2
Far-OOD AUROC	91.4	90.8	**91.4**	91.2	91.4	80.7	91.2	90.6	78.8	80.1	74.5
Correlation with MLS	-	-	1.00	0.99	1.00	0.68	0.96	0.98	-0.47	-0.48	-0.45
Correlation with MSP	-	-	0.79	0.78	0.79	0.57	0.77	0.78	-0.42	-0.42	-0.41

Table 4.9: AUROC scores for GradCAM on CIFAR10. The highest non-baseline value for Near- and Far-OOD is highlighted in bold. ↓ denotes that ID data points more often have a lower score with this aggregation, and thus the output values have been negated (as described in section 2.3.2)

Integrated Gradients

With integrated gradients we see another surprising result. Here, the mean and median are higher for ID data, while the vector norm, range, maximum and third quartile are lower.

Aggregation type	Baselines		Magnitude of saliencies						Statistical dispersion		
Aggregate	MLS	MSP	Mean	Median	Norm↓	Range↓	Max↓	Q3↓	CV↓	RMD↓	QCD↓
Near-OOD AUROC	86.7	87.7	83.1	59.2	78.6	76.0	74.8	78.4	**84.6**	51.0	52.7
Far-OOD AUROC	91.4	90.8	**88.4**	57.9	68.6	65.1	64.0	70.7	80.0	57.4	51.0
Correlation with MLS	-	-	0.65	0.08	-0.10	-0.07	-0.06	-0.11	-0.03	-0.01	0.00
Correlation with MSP	-	-	0.50	0.06	-0.07	-0.05	-0.05	-0.06	-0.02	0.01	-0.00

Table 4.10: AUROC scores for IntegratedGradients on CIFAR10. The highest non-baseline value for Near- and Far-OOD is highlighted in bold. ↓ denotes that ID data points more often have a lower score with this aggregation, and thus the output values have been negated (as described in section 2.3.2)

This implies that ID data has higher positive saliencies on average than OOD data, but that the magnitude of both positive and negative saliencies is higher on OOD data. Looking at figure 4.10, we see this difference clearly.

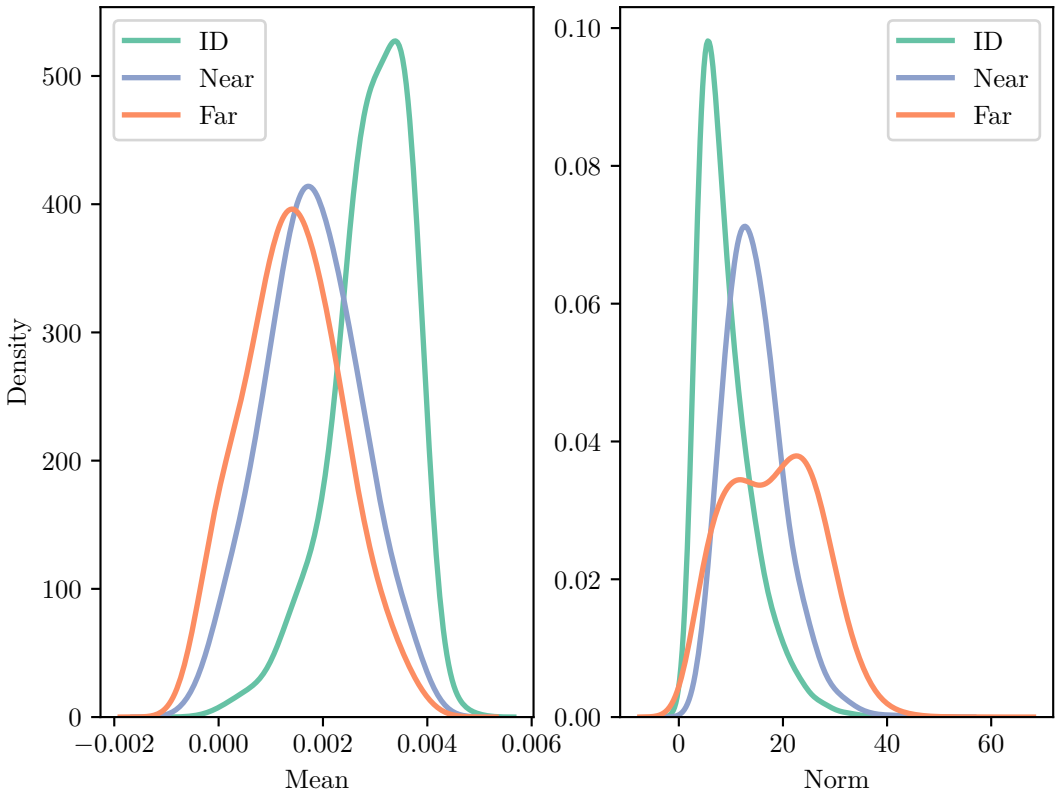

Figure 4.10: Mean and vector norm density plots for Integrated Gradients saliencies.

GBP

GBP saliency maps separate ID and OOD data decently, with the best saliency aggregation method (vector norm) achieving scores which are a few percentage points below the baselines. The only surprise here is that the mean is lower on average for ID data, which it also was when applying GBP to ImageNet200.

Aggregation type	Baselines		Magnitude of saliencies						Statistical dispersion		
Aggregate	MLS	MSP	Mean↓	Median	Norm	Range	Max	Q3	CV↓	RMD↓	QCD
Near-OOD AUROC	86.7	87.7	56.0	58.6	**83.4**	81.2	80.6	65.5	53.1	64.6	51.8
Far-OOD AUROC	91.4	90.8	50.0	70.6	**89.6**	88.7	87.5	79.1	61.6	69.2	60.9
Correlation with MLS	-	-	-0.11	0.03	0.30	0.22	0.22	0.18	-0.04	-0.03	-0.02
Correlation with MSP	-	-	-0.08	0.01	0.21	0.15	0.15	0.11	-0.02	-0.02	-0.01

Table 4.11: AUROC scores for GBP on CIFAR10. The highest non-baseline value for Near- and Far-OOD is highlighted in bold. ↓ denotes that ID data points more often have a lower score with this aggregation, and thus the output values have been negated (as described in section 2.3.2)

Overall results on CIFAR10

Figure 4.11 shows the average scores for each aggregation over the 5 different XAI saliency methods. From this we see that again, vector norm has the highest degree of separation overall, and occlusion the worst.

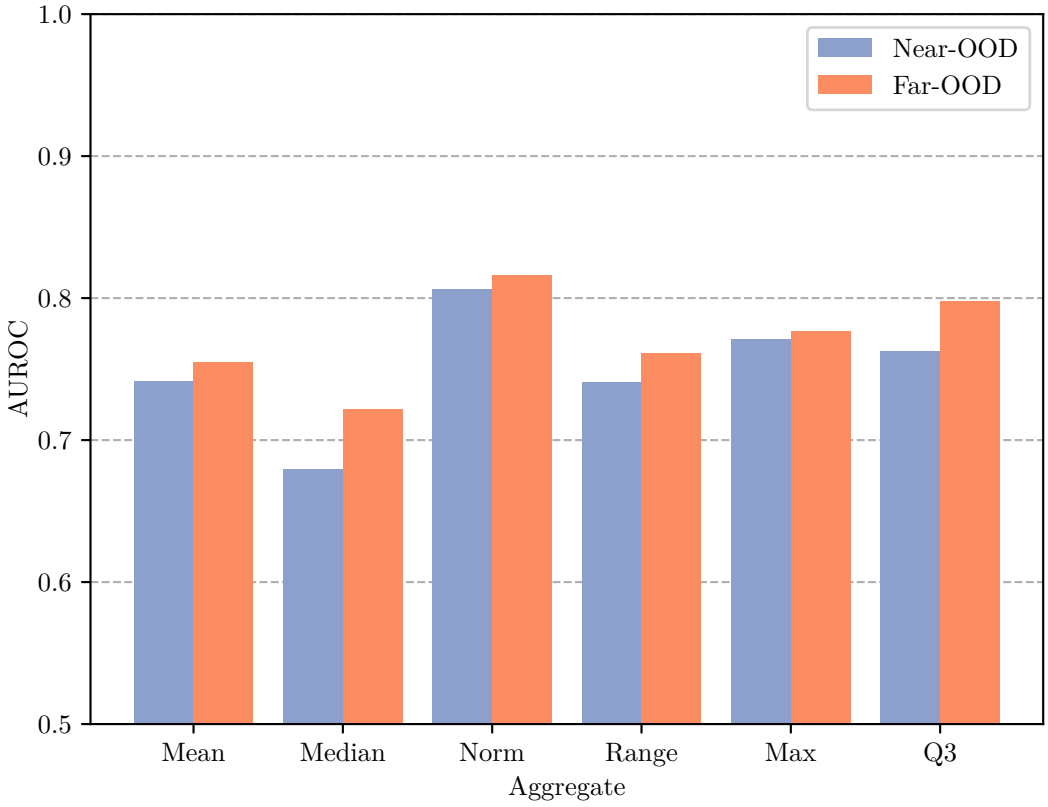

Figure 4.11: Barplot of average AUROC scores for each metric on CIFAR10. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

Table 4.12 shows the results in more detail. Here, we see that not only is the vector norm the most discriminative when averaging across Near and Far, but also individually on each of the categories as well.

Aggregation type	Magnitude of saliencies						Statistical dispersion		
Aggregate	Mean	Median	Norm	Range	Max	Q3	CV	RMD	QCD
Near-OOD AUROC	74.1	68.0	**80.6**	74.1	77.1	76.2	65.8	61.6	57.6
Far-OOD AUROC	75.4	72.1	**81.6**	76.1	77.7	79.8	66.2	65.0	58.2
Mean AUROC	74.8	70.1	**81.1**	75.1	77.4	78.0	66.0	63.3	57.9

Table 4.12: Average AUROC scores for all XAI saliency methods on CIFAR10. The highest non-baseline value for Near- and Far-OOD is highlighted in bold.

Figure 4.12 shows the results when we instead look at the best aggregation for each XAI saliency method. Here we see that, as with ImageNet200, GradCAM, integrated

gradients and GBP separate ID and OOD reasonably well, while occlusion does not. Unlike ImageNet200, LIME does not achieve high degrees of separation on CIFAR10, and lags behind the gradient based methods.

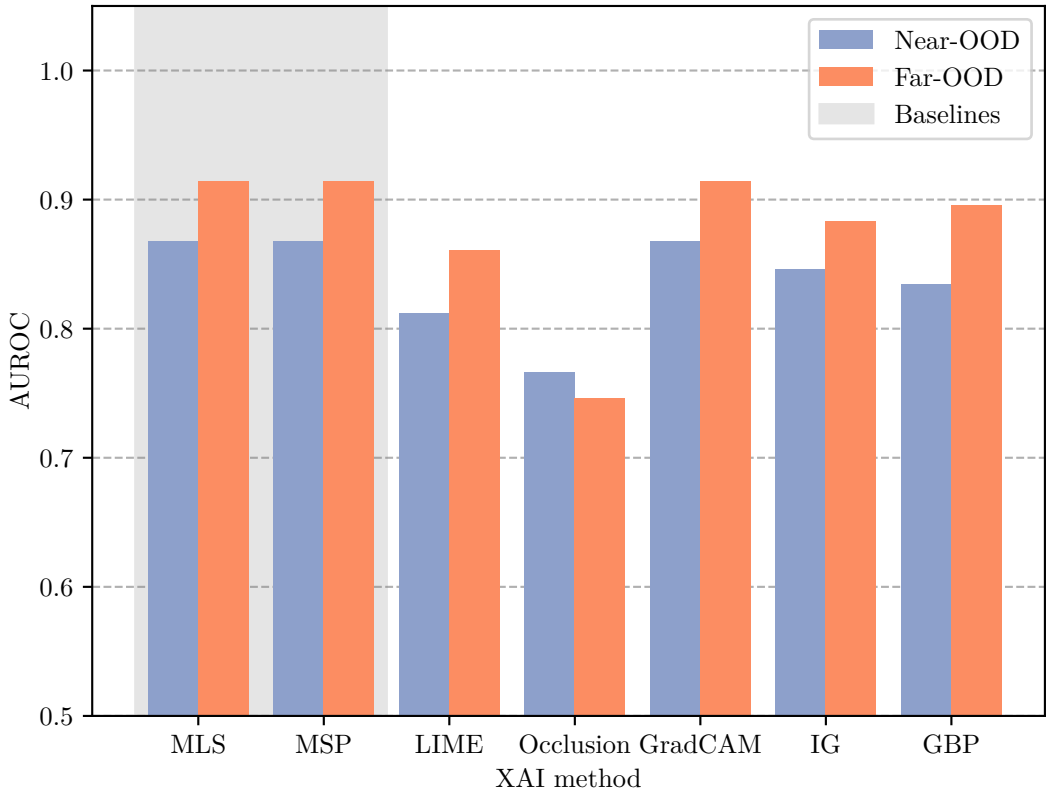

Figure 4.12: Barplot of the highest AUROC scores per XAI saliency method on CIFAR10. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

Finally, we can look at all combinations of aggregation and XAI method when we combine both Near- and Far-OOD separation. From figure 4.13, we see that many of the combinations which separated ID and OOD well on ImageNet200 also do so on CIFAR10: the vector norm with GBP and GradCAM and the mean with integrated gradients all perform very well. Contrasting with ImageNet200, we find that on CIFAR10, using mean aggregation is actually more discriminative than the vector norm on LIME saliencies. In addition, we also see that the mean leads to an every slightly higher score for GradCAM.

Figure 4.13: Heatmap of overall performance AUROC performance for all combinations of XAI methods and magnitude aggregations on CIFAR10

4.1.3 Overall results on both validation benchmarks

Finally, let us consider both validation benchmarks together. Figure 4.14 shows the average degree of separation over both Near-OOD and Far-OOD for both ImageNet200 and CIFAR10. Using this heatmap, we can select the most promising combinations to use for the final OOD detectors on the testing benchmarks.

Figure 4.14: Heatmap of overall performance AUROC performance for all combinations of XAI methods and magnitude aggregations on ImageNet200 and CIFAR10

As we can see, occlusion has consistently underperformed when using saliency aggregation on both benchmarks, and as such I will not use this XAI method in the Saliency Aggregation or Saliency Aggregation plus Logit frameworks. For the remaining four XAI methods, I select the most discriminative aggregation on both of the validation benchmarks for use on the testing benchmarks. From the heatmap, we can see that this results in using vector norm for LIME, GradCAM and GBP, and mean for integrated gradients. In the next section, I will denote these combinations as LIMENorm, GradCAMNorm, IGMean and GBPNorm.

4.2 Evaluation of XAI OOD detectors

This section contains the final tests conducted on the testing benchmarks. This section will be divided into three parts, corresponding to the three proposed OOD detection frameworks introduced in the methodology: Saliency Aggregation, Saliency Aggregation plus Logit and SaliencyVIM. Each of these sections will detail the performance of the corresponding framework on the four testing benchmarks, and statistically compare the results to the baseline methods. For Saliency Aggregation and Saliency Aggregation plus Logit, the tests are conducted using the combinations of XAI methods and aggregations that performed the best on the validation benchmarks, as described in the preceding

section. The statistical analysis is based on ten bootstraps of each benchmarks, which has been performed on each method, as well as the baselines. As mentioned in section 3.6.2, the Bonferroni-corrected threshold for statistical significance is set at $0.05/n$, where n is the number of experiments conducted on each method.

When reporting the p-values, I follow the `R`-standard for appending "significance stars", which give a quick visual indication of the statistical significance of a result. In `R`, any p-value lower than 0.05 has an asterisk appended, any p-value lower than 0.01 has two asterisks appended, and any p-value lower than 0.001 has three asterisks appended. I also append these asterisks, but use Bonferroni corrected p-values when determining if a value should have an asterisk appended. This means that whenever a p-value has at least one asterisk appended, the reader knows that the corresponding Wilcoxon signed-rank test showed statistical significance according to a Bonferroni corrected level of significance of $0.05/n$. It should be noted that due to the discrete nature of the Wilcoxon signed-rank test, there is a minimal p-value that can be attained, based on the number of experiments. With ten bootstraps, the minimal p-value is 0.001, which is achieved when all baseline AUROC scores are lower than the method tested.

For each experiment, I first report the results for each of the four benchmarks in an objective manner, by simply describing the scores attained by the different methods and comparing them to the baseline methods. At the end of each experiment, I give an analysis of the results and attempt to connect the results to the problem statement.

4.2.1 Results for Saliency Aggregation

Saliency Aggregation, as described in 3.1.1, is the first OOD detection framework that will be tested. As we have seen from the validation benchmarks, saliency aggregation on its own can sometimes perform on par with the baselines on Near-OOD and can sometimes surpass the baselines on Far-OOD. The Saliency Aggregation framework requires a choice of XAI saliency mapping method s and aggregation function A. From the results in section 4.1, the combinations I will use are LIME, GradCAM and GBP with vector norm aggregation, and integrated gradients with mean aggregation.

In this section, the generation of saliency maps and aggregation will be done again, on ten bootstraps of the four testing benchmarks, and AUROC scores will be calculated for each bootstrap for each method.

ImageNet200

As in the preceding section, we first investigate the results on the ImageNet200 testing benchmark. As explained in section 3.3.3, this benchmark consists of held out samples from the ImageNet200 dataset and all corresponding OOD datasets, and thus allows us to calculate unbiased results.

Figure 4.15 shows the mean AUROC for the baselines and the three combinations of XAI methods and aggregation functions mentioned above. In addition, the confidence intervals for each mean value is plotted as whiskers.

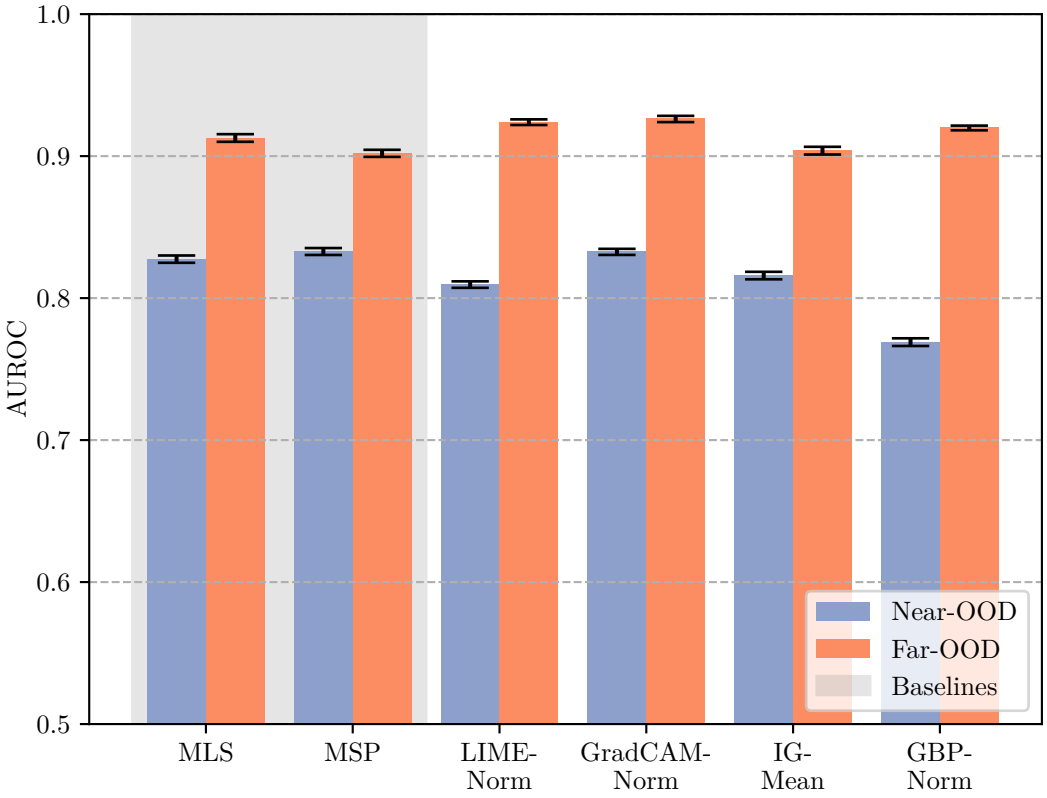

Figure 4.15: Barplot of average AUROC scores for Saliency Aggregation on the bootstrapped ImageNet200 testing benchmark. 95% confidence intervals are plotted as whiskers. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

As we can see, the vector norm of LIME, GradCAM and GBP actually seems to outperform the baselines on Far-OOD, while the mean of the integrated gradients is about on par with the Far-OOD performance of MSP. When it comes to Near-OOD, only the norm of GradCAM seems to compete with the baselines.

To get a more precise understanding of the performance of the different models, we turn to table 4.13, which shows the results of the Wilcoxon signed-rank tests done against the baseline methods. From this table we see that the Far-OOD results for LIMENorm, GradCAMNorm and GBPNorm were indeed statistically significantly higher than both baselines.

Dataset	AUROC	ΔAUROC MLS	P-value MLS	ΔAUROC MSP	P-value MSP
LIMENorm					
Near-OOD	80.95	-1.792	1.000	-2.329	1.000
Far-OOD	92.39	+1.113	0.001 **	+2.195	0.001 **
GradCAMNorm					
Near-OOD	83.26	+0.511	0.001 **	-0.026	0.839
Far-OOD	92.62	+1.339	0.001 **	+2.421	0.001 **
IGMean					
Near-OOD	81.59	-1.156	1.000	-1.694	1.000
Far-OOD	90.38	-0.894	1.000	+0.188	0.002 **
GBPNorm					
Near-OOD	76.90	-5.845	1.000	-6.382	1.000
Far-OOD	91.98	+0.706	0.001 **	+1.788	0.001 **

Table 4.13: Results of performing a Wilcoxon signed-rank test on the AUROC means of against MLS and MSP, showing the mean AUROC over 10 runs on ImageNet200, the difference in means compared to the baselines, and the corresponding p-values. Each p-value is appended a significance code which follows the `R`-standard.

CIFAR10

Next, we turn our attention to CIFAR10. Figure 4.16 shows the bootstrapped means and the confidence intervals. Here, we can see that the results are in general worse than the baselines, mirroring the initial findings from the validation benchmark (section 4.1.2). In this case, only GradCAMNorm can compete with the baseline methods.

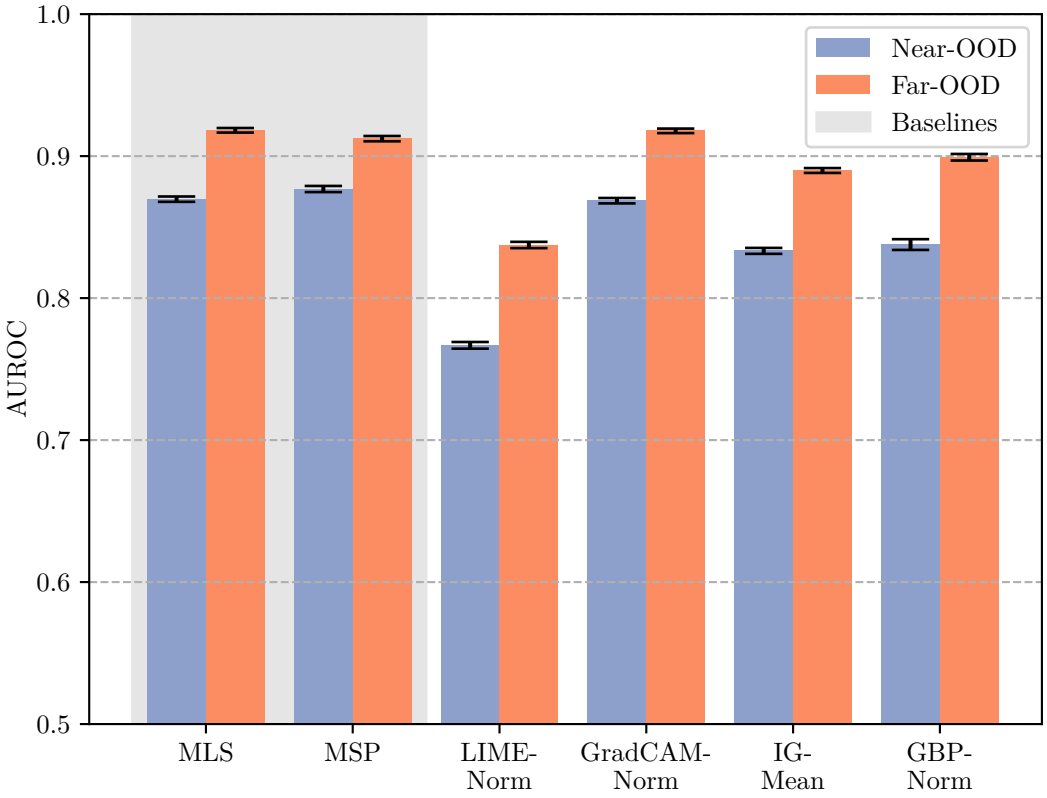

Figure 4.16: Barplot of average AUROC scores for Saliency Aggregation on the bootstrapped CIFAR10 testing benchmark. 95% confidence intervals are plotted as whiskers. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

From this plot, we do not expect to see many statistically significant improvements. Indeed, table 4.14 shows that no method outperforms both baselines on either Near- or Far-OOD, and in general all methods have a mean AUROC which is far lower than the baseline methods.

Dataset	AUROC	ΔAUROC MLS	P-value MLS	ΔAUROC MSP	P-value MSP
LIMENorm					
Near-OOD	76.67	-10.294	1.000	-11.008	1.000
Far-OOD	83.74	-8.078	1.000	-7.490	1.000
GradCAMNorm					
Near-OOD	86.86	-0.108	1.000	-0.822	1.000
Far-OOD	91.78	-0.039	1.000	+0.549	0.001 **
IGMean					
Near-OOD	83.33	-3.642	1.000	-4.357	1.000
Far-OOD	88.99	-2.832	1.000	-2.244	1.000
GBPNorm					
Near-OOD	83.77	-3.197	1.000	-3.912	1.000
Far-OOD	89.92	-1.898	1.000	-1.310	1.000

Table 4.14: Results of performing a Wilcoxon signed-rank test on the AUROC means of against MLS and MSP, showing the mean AUROC over 10 runs on CIFAR10, the difference in means compared to the baselines, and the corresponding p-values. Each p-value is appended a significance code which follows the R-standard.

ImageNet1K

Next, we look at the first benchmark which was not used during the data analysis, ImageNet1K.

From figure 4.17, we see that three of the XAI saliency aggregation methods comfortably outperform the baselines on Far-OOD; LIMENorm, GradCAMNorm and GBPNorm. On Near-OOD, as with ImageNet200, their performance is not competitive with the baselines.

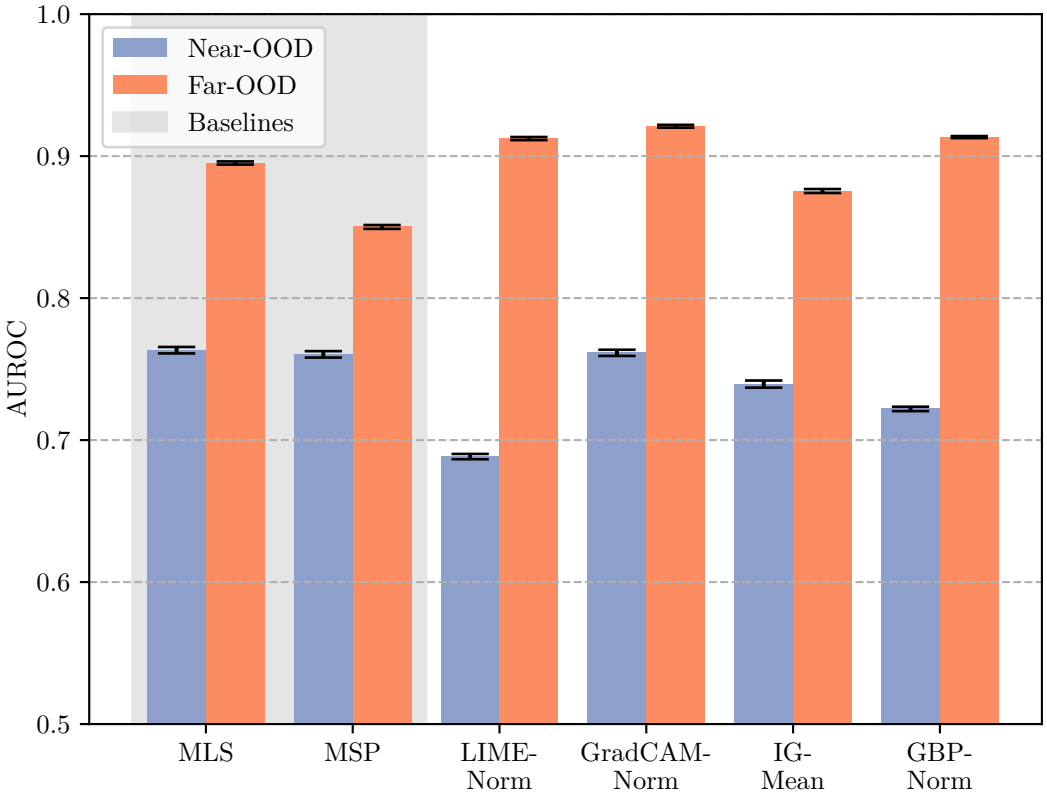

Figure 4.17: Barplot of average AUROC scores for Saliency Aggregation on the bootstrapped ImageNet testing benchmark. 95% confidence intervals are plotted as whiskers. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

Looking at table 4.15, the three methods mentioned above all report considerable AUROC improvements over both baselines on Far-OOD

Dataset	AUROC	ΔAUROC MLS	P-value MLS	ΔAUROC MSP	P-value MSP
LIMENorm					
Near-OOD	68.84	-7.492	1.000	-7.199	1.000
Far-OOD	91.24	+1.721	0.001 **	+6.225	0.001 **
GradCAMNorm					
Near-OOD	76.15	-0.185	0.999	+0.107	0.065
Far-OOD	92.10	+2.587	0.001 **	+7.091	0.001 **
IGMean					
Near-OOD	73.95	-2.386	1.000	-2.093	1.000
Far-OOD	87.54	-1.973	1.000	+2.531	0.001 **
GBPNorm					
Near-OOD	72.20	-4.136	1.000	-3.843	1.000
Far-OOD	91.34	+1.825	0.001 **	+6.329	0.001 **

Table 4.15: Results of performing a Wilcoxon signed-rank test on the AUROC means of against MLS and MSP, showing the mean AUROC over 10 runs on Imagenet, the difference in means compared to the baselines, and the corresponding p-values. Each p-value is appended a significance code which follows the R-standard.

CIFAR100

Finally, we turn to CIFAR100, the second benchmark not used during the analysis. Figure 4.18 shows the average AUROC scores on this benchmark. Like with CIFAR10, the results here are worse than on ImageNet, and LIMENorm again performs poorly.

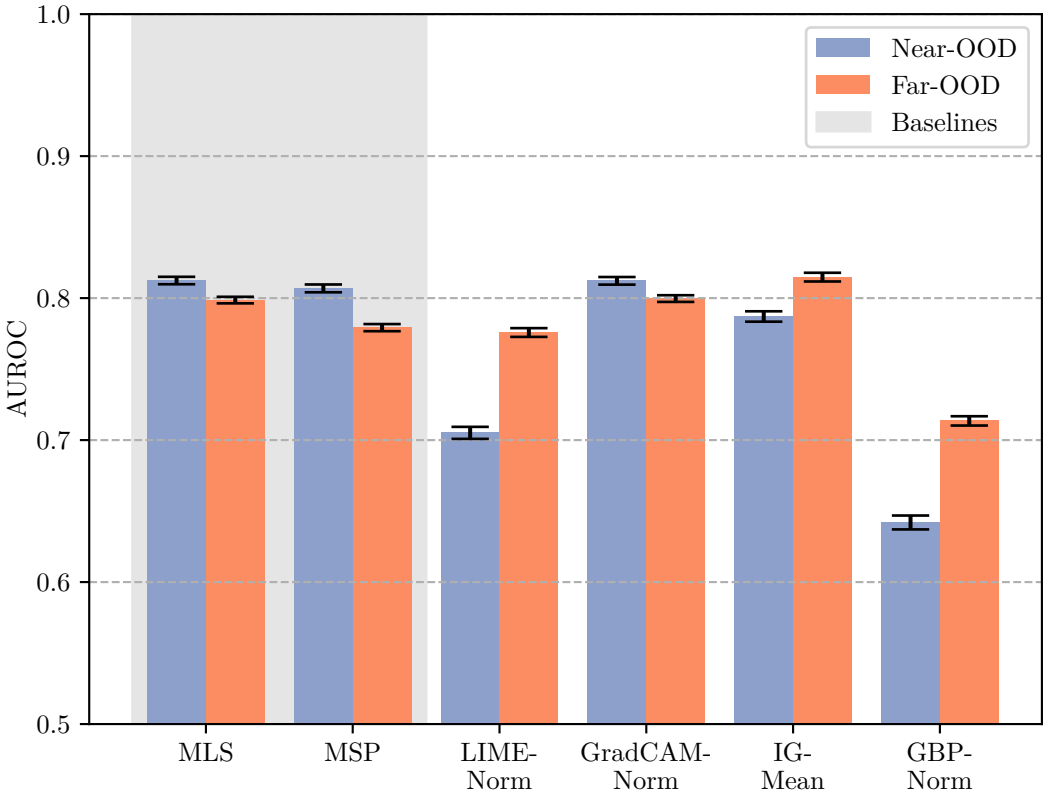

Figure 4.18: Barplot of average AUROC scores for Saliency Aggregation on the bootstrapped CIFAR100 testing benchmark. 95% confidence intervals are plotted as whiskers. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

From table 4.16, we see that both GradCAMNorm and IGMean perform well on Far-OOD, while LIMENorm and GBPNorm fall far behind, with double digit percentage point losses compared to the baselines.

Dataset	AUROC	ΔAUROC MLS	P-value MLS	ΔAUROC MSP	P-value MSP
LIMENorm					
Near-OOD	70.51	-10.728	1.000	-10.175	1.000
Far-OOD	77.58	-2.285	1.000	-0.344	0.998
GradCAMNorm					
Near-OOD	81.22	-0.020	0.958	+0.533	0.001 **
Far-OOD	79.97	+0.106	0.001 **	+2.047	0.001 **
IGMean					
Near-OOD	78.71	-2.532	1.000	-1.979	1.000
Far-OOD	81.48	+1.616	0.001 **	+3.557	0.001 **
GBPNorm					
Near-OOD	64.20	-17.039	1.000	-16.486	1.000
Far-OOD	71.35	-8.509	1.000	-6.568	1.000

Table 4.16: Results of performing a Wilcoxon signed-rank test on the AUROC means of against MLS and MSP, showing the mean AUROC over 10 runs on CIFAR100, the difference in means compared to the baselines, and the corresponding p-values. Each p-value is appended a significance code which follows the `R`-standard.

Overall analysis of Saliency Aggregation

Overall, the results from simply aggregating the saliency maps of different XAI methods are very promising, especially considering that the only other work which has attempted to use XAI saliency maps for OOD detection barely achieved AUROC scores above 0.50. These results show that XAI methods extract valuable information from the network, which can be used to effectively discriminate between ID and OOD data samples. The results on ImageNet200 and ImageNet1K are particularly interesting, as LIMENorm, GradCAMNorm and GBPNorm all outperform the baselines on Far-OOD. The results on CIFAR10 and CIFAR100 are less impressive, and show that these methods may not achieve consistently good results on all benchmarks. However, it should be noted that this inconsistency is not unusual amongst OOD detectors, as one of the takeaways from from [6] was that there is "no single winner that always outperforms others across multiple data sets".

4.2.2 Results for Saliency Aggregation plus Logit

Given the fact that Saliency Aggregation has shown itself to be capable of differentiating ID and OOD samples in many cases, especially for Far-OOD, we might expect good results from Saliency Aggregation plus Logit. As mentioned in section 3.1.2, `COMBOOD` [10] achieved SoTA results on ImageNet200 and ImageNet1K when combining two complementary distance metrics. As we have seen from section 4.2.1, most XAI methods (except for GradCAM) have relatively low correlation with either MSP or MLS, which could give similar benefits as those found by `COMBOOD`.

ImageNet200

Again, we start with the ImageNet200 testing benchmark. Figure 4.19 shows the results of the 10 bootstraps. We can see that the poor performances on Near-OOD

have mostly disappeared when combining saliency aggregation with MLS: Instead, all methods perform quite closely to the baselines on Near-OOD.

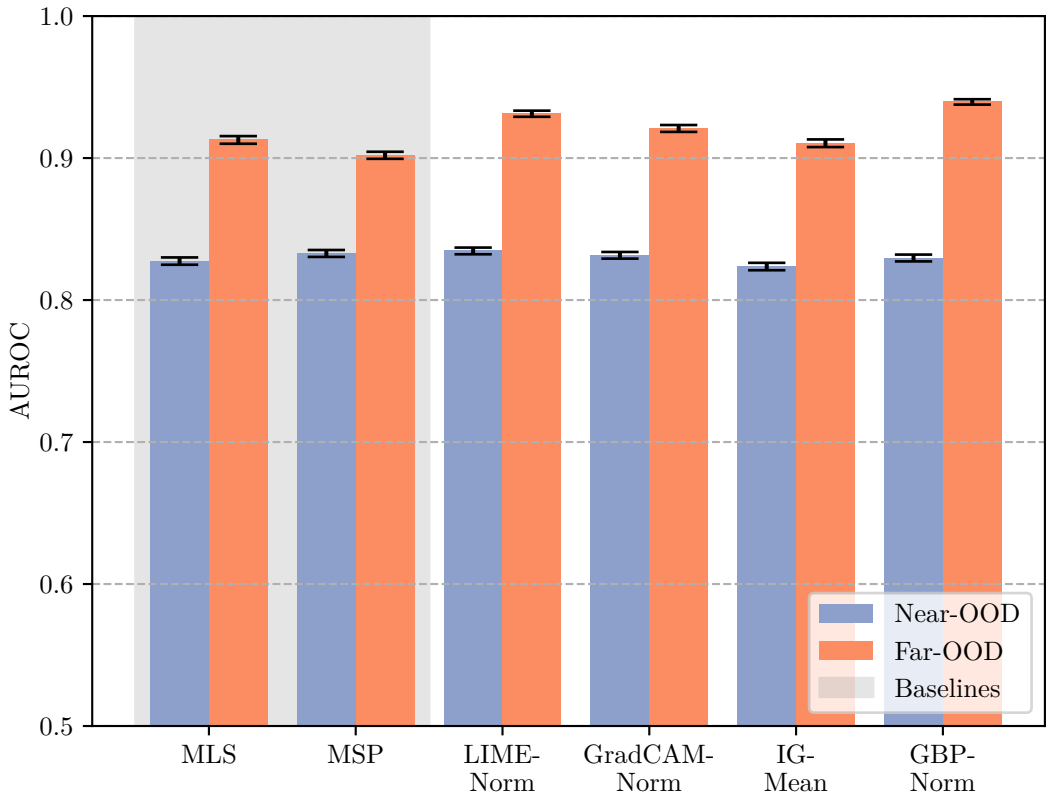

Figure 4.19: Barplot of average AUROC scores for Saliency Aggregation plus Logit on the bootstrapped ImageNet200 testing benchmark. 95% confidence intervals are plotted as whiskers. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

In addition to the Near-OOD improvements, GBPNorm+Logit achieves a very high Far-OOD performance on ImageNet200. LIMENorm+Logit improves considerably over LIMENorm, and also beats the baselines on Far-OOD. GradCAMNorm+Logit also outperforms the baselines, but we should remember that GradCAMNorm (without the addition of logits) also performed well, so these results may not be better than simply using GradCAMNorm alone. Let us look closer at the performance of each method, considering the results of the Wilcoxon signed-rank tests performed against the baseline methods.

As we see from table 4.17, we see far more improvements when combining our saliency aggregation method with the MLS. Where we previously saw multiple percentage points lower Near-OOD on several methods with Saliency Aggregation, we now have no method which performs worse than a single percentage point than either baseline.

Dataset	AUROC	ΔAUROC MLS	P-value MLS	ΔAUROC MSP	P-value MSP
LIMENorm+Logit					
Near-OOD	83.46	+0.715	0.001 **	+0.178	0.007 *
Far-OOD	93.12	+1.845	0.001 **	+2.927	0.001 **
GradCAMNorm+Logit					
Near-OOD	83.15	+0.407	0.001 **	-0.130	0.968
Far-OOD	92.09	+0.808	0.001 **	+1.890	0.001 **
IGMean+Logit					
Near-OOD	82.36	-0.384	1.000	-0.922	1.000
Far-OOD	91.04	-0.233	1.000	+0.848	0.001 **
GBPNorm+Logit					
Near-OOD	82.96	+0.217	0.010 *	-0.320	0.990
Far-OOD	93.95	+2.677	0.001 **	+3.759	0.001 **

Table 4.17: Results of performing a Wilcoxon signed-rank test on the AUROC means of against MLS and MSP, showing the mean AUROC over 10 runs on ImageNet200, the difference in means compared to the baselines, and the corresponding p-values. Each p-value is appended a significance code which follows the `R`-standard.

CIFAR10

Next, we turn to CIFAR10. As we can see from figure 4.20, GBPNorm+Logit not only beats the baselines in the Far-OOD category, but also on Near-OOD, where the performance is considerably higher. In addition, LIMENorm+Logit and IGMean+Logit outperform the baselines on Far-OOD.

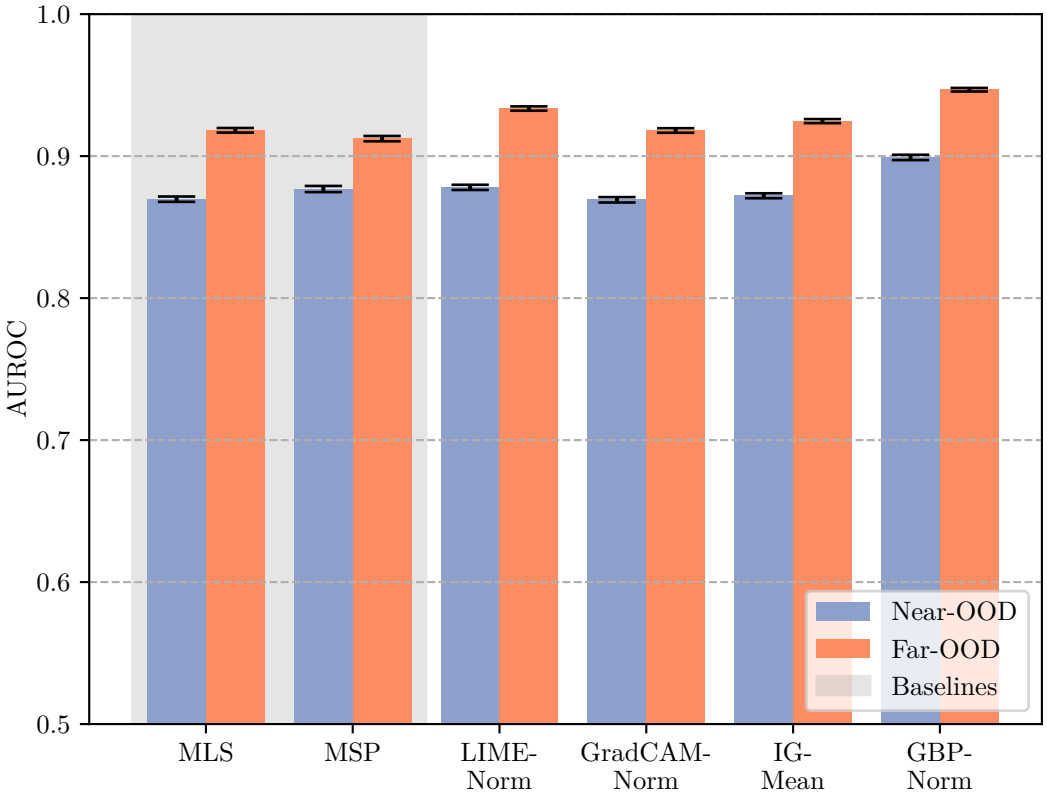

Figure 4.20: Barplot of average AUROC scores for Saliency Aggregation plus Logit on the bootstrapped CIFAR10 testing benchmark. 95% confidence intervals are plotted as whiskers. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

Like in the previous sections, let look at the results of the Wilcoxon signed-rank tests to gain a more robust understanding of the performance of the different methods. As table 4.18 shows, we now see statistically significant results on Far-OOD for all methods except for GradCAMNorm+Logit. In addition, GBPNorm+Logit sees a large improvement on both Near- and Far-OOD datasets.

Dataset	AUROC	ΔAUROC MLS	P-value MLS	ΔAUROC MSP	P-value MSP
LIMENorm+Logit					
Near-OOD	87.80	+0.831	0.001 **	+0.116	0.053
Far-OOD	93.35	+1.532	0.001 **	+2.120	0.001 **
GradCAMNorm+Logit					
Near-OOD	86.92	-0.046	1.000	-0.760	1.000
Far-OOD	91.81	-0.013	1.000	+0.575	0.001 **
IGMean+Logit					
Near-OOD	87.21	+0.240	0.001 **	-0.475	1.000
Far-OOD	92.47	+0.647	0.001 **	+1.236	0.001 **
GBPNorm+Logit					
Near-OOD	89.91	+2.940	0.001 **	+2.225	0.001 **
Far-OOD	94.68	+2.858	0.001 **	+3.446	0.001 **

Table 4.18: Results of performing a t-test on the AUROC means of against MLS and MSP, showing the mean AUROC over 10 runs on CIFAR10, the difference in means compared to the baselines, and the corresponding p-values. Each p-value is appended a significance code which follows the R-standard.

ImageNet1K

Next, we turn to ImageNet1K. Again, we see high AUROC values from combining saliency aggregation and the maximum logit, with several methods outperforming the baselines by several percentage points.

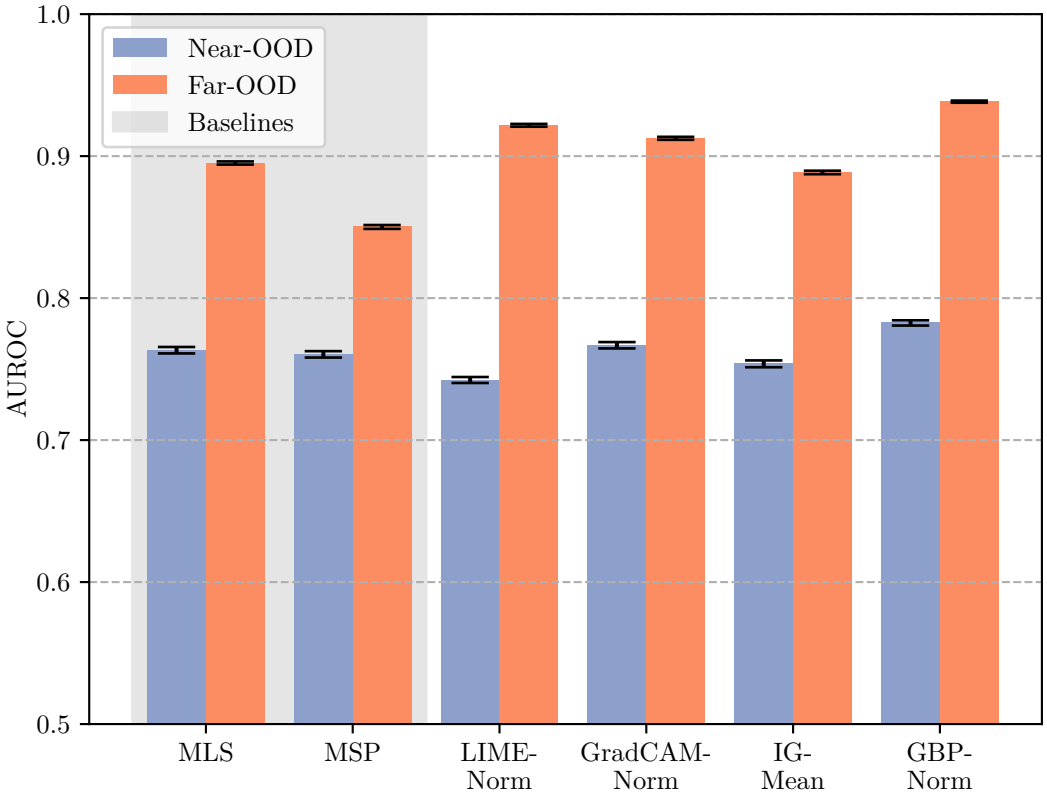

Figure 4.21: Barplot of average AUROC scores for Saliency Aggregation plus Logit on the bootstrapped ImageNet200 testing benchmark. 95% confidence intervals are plotted as whiskers. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

Figure 4.22: Barplot of average AUROC scores on ImageNet1K. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

Looking at table 4.19, we see that the performance on Far-OOD is high across all methods except for IGMean, with GBPNorm+Logit especially reporting a substantial improvement on both Near-OOD and Far-OOD.

Dataset	AUROC	ΔAUROC MLS	P-value MLS	ΔAUROC MSP	P-value MSP
LIMENorm+Logit					
Near-OOD	74.23	-2.098	1.000	-1.806	1.000
Far-OOD	92.17	+2.656	0.001 **	+7.160	0.001 **
GradCAMNorm+Logit					
Near-OOD	76.68	+0.348	0.001 **	+0.641	0.001 **
Far-OOD	91.25	+1.739	0.001 **	+6.243	0.001 **
IGMean+Logit					
Near-OOD	75.37	-0.960	1.000	-0.668	1.000
Far-OOD	88.85	-0.665	1.000	+3.839	0.001 **
GBPNorm+Logit					
Near-OOD	78.25	+1.920	0.001 **	+2.212	0.001 **
Far-OOD	93.83	+4.320	0.001 **	+8.824	0.001 **

Table 4.19: Results of performing a Wilcoxon signed-rank test on the AUROC means of against MLS and MSP, showing the mean AUROC over 10 runs on Imagenet, the difference in means compared to the baselines, and the corresponding p-values. Each p-value is appended a significance code which follows the R-standard.

CIFAR100

Finally, we turn to CIFAR100. Again we find that several of the XAI based methods outperform the baselines on Far-OOD, while the performance on Near-OOD is competitive with baseline methods.

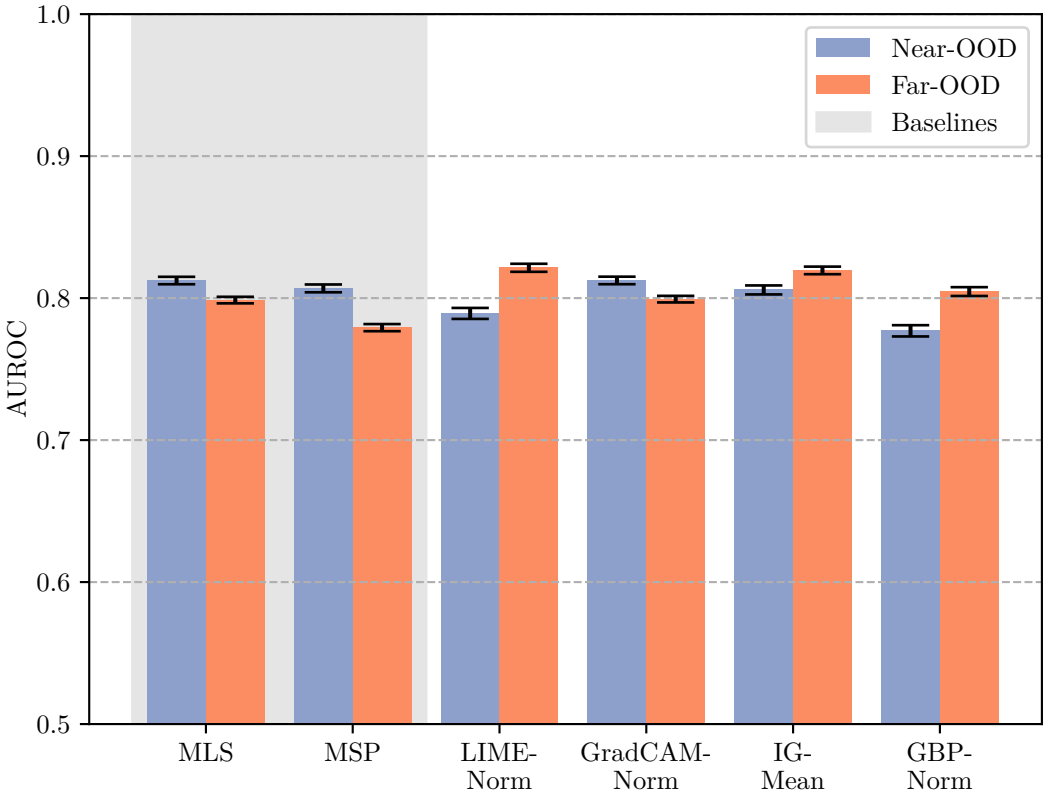

Figure 4.23: Barplot of average AUROC scores for Saliency Aggregation plus Logit on the bootstrapped CIFAR100 testing benchmark. 95% confidence intervals are plotted as whiskers. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

Looking at table 4.20, we see that all methods have statistically significantly higher AUROC scores than both baseline methods on Far-OOD, except for GradCAMNorm, which is only statistically significantly better than MLS. For Near-OOD, none of the methods surpass the baselines.

Dataset	AUROC	ΔAUROC MLS	P-value MLS	ΔAUROC MSP	P-value MSP
LIMENorm+Logit					
Near-OOD	78.92	-2.316	1.000	-1.763	1.000
Far-OOD	82.13	+2.273	0.001 **	+4.214	0.001 **
GradCAMNorm+Logit					
Near-OOD	81.24	+0.006	0.116	+0.559	0.001 **
Far-OOD	79.93	+0.066	0.001 **	+2.007	0.001 **
IGMean+Logit					
Near-OOD	80.58	-0.662	1.000	-0.108	0.981
Far-OOD	81.95	+2.089	0.001 **	+4.030	0.001 **
GBPNorm+Logit					
Near-OOD	77.70	-3.542	1.000	-2.989	1.000
Far-OOD	80.46	+0.599	0.001 **	+2.540	0.001 **

Table 4.20: Results of performing a Wilcoxon signed-rank test on the AUROC means of against MLS and MSP, showing the mean AUROC over 10 runs on CIFAR100, the difference in means compared to the baselines, and the corresponding p-values. Each p-value is appended a significance code which follows the R-standard.

Overall analysis of Saliency Aggregation plus Logit

Overall, the results build on the Saliency Aggregation framework and improve the performance considerably. While the results of Saliency Aggregation showed a clear deficit on Near-OOD, the addition of the MLS into the OOD detector consistently reduced this deficit and equalized the performance when compared to the baselines. These results concur with those found by [10], which also found that the subpar Near-OOD performance of their Mahalanobis OOD detector was improved when combined with nearest neighbour OOD detection.

The performance of GBPNorm+Logit is a particularly interesting example, beating both baselines by several percentage points on all four benchmarks on Far-OOD. In fact, the performance is actually quite close to SoTA scores on all benchmarks except for CIFAR100: On ImageNet200, the Far-OOD AUROC was 93.83%, only slightly more than a percentage point behind ASH+PixMix (95.01%), the best performing Far-OOD method. On CIFAR10, the performance of GBPNorm+Logit is quite close to the best performing method which does not require retraining, `COMBOOD`. The Far-OOD scores are essentially equal (94.68% for GBPNorm+Logit vs 94.65% for `COMBOOD`) while the Near-OOD score is only two percentage points behind (89.91% vs 91.13%). On ImageNet1K, the Far-OOD performance of GBPNorm+Logit was also quite impressive: The AUROC achieved was 93.83%, only four percentage points below the best performing SoTA model, which is AdaSCALE [47] at 97.85%.

Considering the results from the other methods, LIMENorm+Logit and IGMean also showcased the clear benefit of combining saliency aggregation with MLS, with scores that were much higher than in the Saliency Aggregation framework. The results for GradCAMNorm+Logit were no better than the equivalent results without the addition of the logit. This is not unexpected, given the extremely high correlation between GradCAM saliency aggregations and the MLS. There is little to be gained from combining two metrics which are almost entirely the same, as we do not gain any supplementary information over just using one of them. [10] found best results when

using different feature extraction strategies for the two distance metrics, which allows for complementary information to be extracted by each metric. With such a high correlation, it is not surprising that no performance was gained when combining MLS and GradCAM vector norms.

In general, these results clearly show that XAI based OOD detectors are not just adequate, but also competitive with the SoTA in the OOD detection field.

4.2.3 Results for SaliencyVIM

For the final tests, we consider SaliencyVIM, the framework described in section 3.1.2. As described in section 3.1.2, SaliencyVIM requires saliency methods which output lower dimensional saliency maps, as opposed to methods such as GBP and integrated gradients, which output a value for each pixel. Thus, the applicable XAI methods included in this thesis are LIME, occlusion and GradCAM. Although occlusion performed very poorly on the validation set when aggregating saliencies, SaliencyVIM uses every saliency value and could thus capture information which has not been captured by aggregation methods. Thus, occlusion should be included in the testing. The three methods from the SaliencyVIM framework are then OcclusionVIM, LIMEVIM and GradCAMVIM.

In this case, it does not make sense to use MLS or MSP as the baselines to compare against, since the method is based on VIM. Instead, we compare our results with VIM, to see whether the addition of saliencies can improve this method.

ImageNet200

We start with the ImageNet200 benchmark. From figure 4.24, we see that the differences are extremely small between the baseline and the XAI methods. However, the confidence intervals are also very small, which might mean that there are statistically significant performance increases. Thus, we look to the results of the Wilcoxon signed-rank tests.

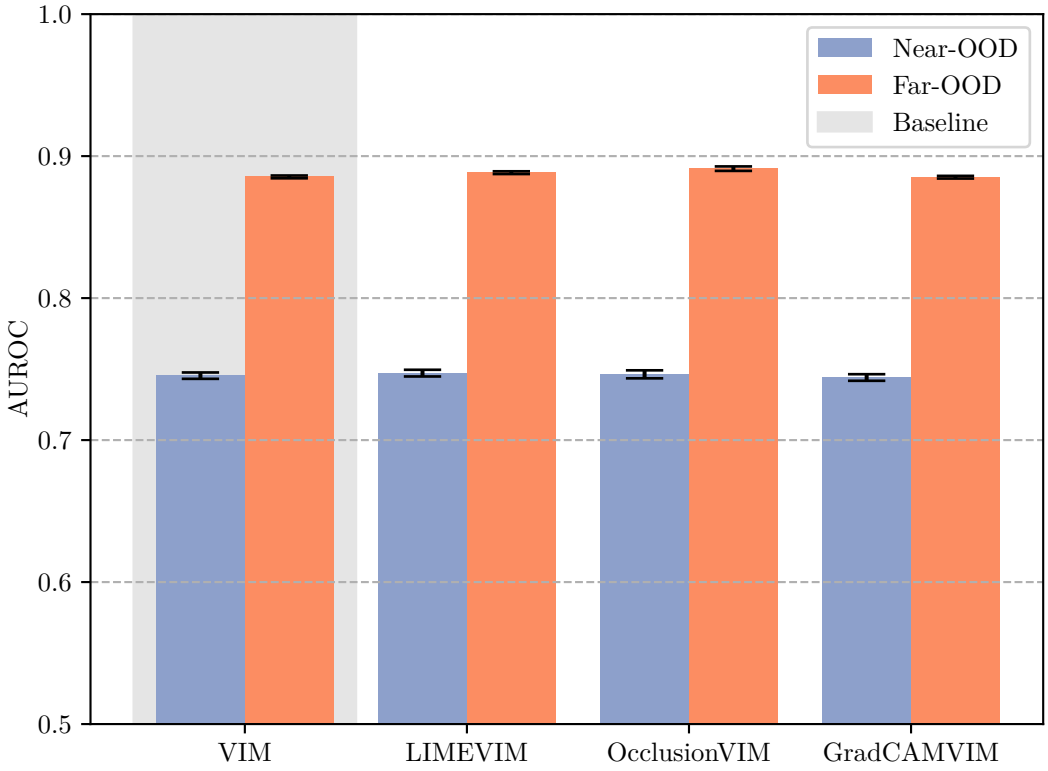

Figure 4.24: Barplot of average AUROC scores for SaliencyVIM on the bootstrapped ImageNet200 testing benchmark. 95% confidence intervals are plotted as whiskers. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

From table 4.21, we see that both OcclusionVIM and LIMEVIM outperform VIM on Far-OOD. In addition, LIMEVIM outperforms the baseline on Near-OOD. GradCAMVIM sees no increases in performance over the baseline on either Near- or Far-OOD.

Dataset	AUROC	ΔAUROC VIM	P-value VIM
LIMEVIM			
Near-OOD	74.72	+0.178	0.001 **
Far-OOD	88.83	+0.298	0.001 **
OcclusionVIM			
Near-OOD	74.63	+0.094	0.312
Far-OOD	89.12	+0.580	0.001 **
GradCAMVIM			
Near-OOD	74.41	-0.130	1.000
Far-OOD	88.51	-0.022	0.993

Table 4.21: Results of performing a Wilcoxon signed-rank test on the AUROC means of against VIM, showing the mean AUROC over 10 runs on ImageNet200, the difference in means compared to the baselines, and the corresponding p-values. Each p-value is appended a significance code which follows the R-standard.

CIFAR10

Next, we perform OOD detection using VIM and SaliencyVIM on CIFAR10. From figure 4.25, SaliencyVIM with GradCAM as the saliency generator seems to be the best performer here, above the baseline. Occlusion and LIME on the other hand, seem to perform worse.

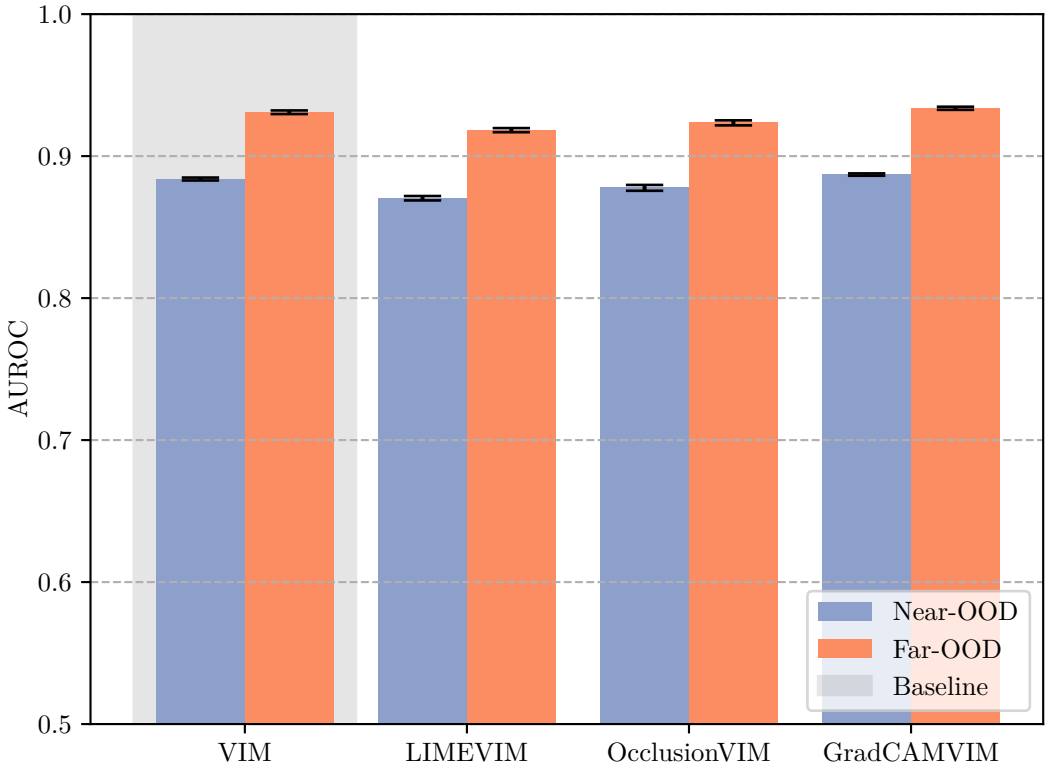

Figure 4.25: Barplot of average AUROC scores for SaliencyVIM on the bootstrapped CIFAR10 testing benchmark. 95% confidence intervals are plotted as whiskers. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

As we see from table 4.22, GradCAMVIM outperforms the baseline on both Near- and Far-OOD. Both occlusion and LIME see drops in performance across the board, contrary to the results on ImageNet200.

Dataset	AUROC	ΔAUROC VIM	P-value VIM
LIMEVIM			
Near-OOD	87.03	-1.352	1.000
Far-OOD	91.83	-1.260	1.000
OcclusionVIM			
Near-OOD	87.76	-0.621	1.000
Far-OOD	92.34	-0.747	1.000
GradCAMVIM			
Near-OOD	88.70	+0.315	0.001 **
Far-OOD	93.37	+0.280	0.001 **

Table 4.22: Results of performing a Wilcoxon signed-rank test on the AUROC means of against VIM, showing the mean AUROC over 10 runs on CIFAR10, the difference in means compared to the baselines, and the corresponding p-values. Each p-value is appended a significance code which follows the R-standard.

ImageNet1K

Next, we look at the results on the ImageNet1K testing benchmark. From figure 4.26, we see that the confidence intervals are extremely small, and we must clearly look at the results from the statistical tests to glean any information from this benchmark.

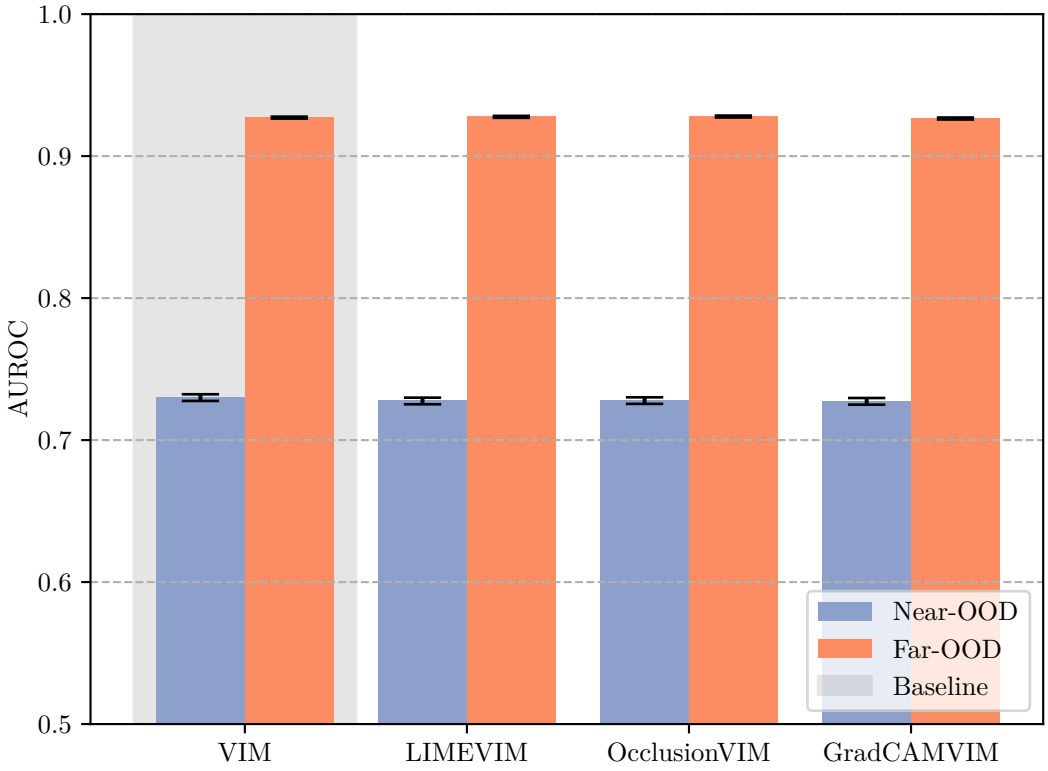

Figure 4.26: Barplot of average AUROC scores for SaliencyVIM on the bootstrapped ImageNet1K testing benchmark. 95% confidence intervals are plotted as whiskers. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

From table 4.23, we see that both LIMEVIM and OcclusionVIM are statistically significantly better than VIM on Far-OOD, while GradCAMVIM is worse on both Near- and Far-OOD.

Dataset	AUROC	ΔAUROC VIM	P-value VIM
LIMEVIM			
Near-OOD	72.76	-0.237	1.000
Far-OOD	92.77	+0.053	0.001 **
OcclusionVIM			
Near-OOD	72.78	-0.212	1.000
Far-OOD	92.79	+0.077	0.001 **
GradCAMVIM			
Near-OOD	72.73	-0.263	1.000
Far-OOD	92.65	-0.063	1.000

Table 4.23: Results of performing a Wilcoxon signed-rank test on the AUROC means of against VIM, showing the mean AUROC over 10 runs on Imagenet, the difference in means compared to the baselines, and the corresponding p-values. Each p-value is appended a significance code which follows the R-standard.

CIFAR100

Finally, we consider the CIFAR100 benchmark. Again, the margins are very close, and we should look to the results of the Wilcoxon signed-rank tests.

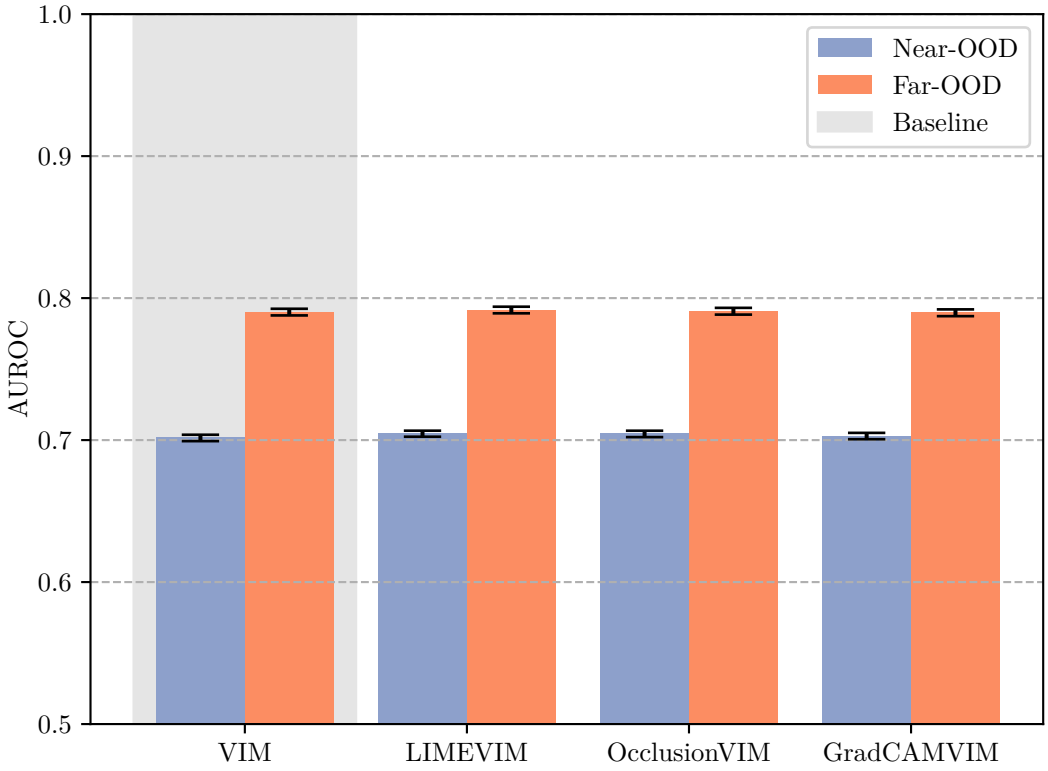

Figure 4.27: Barplot of average AUROC scores for SaliencyVIM on the bootstrapped CIFAR100 testing benchmark. 95% confidence intervals are plotted as whiskers. Note that the y-axis starts at 0.50, the practical floor for an AUROC score.

Looking at table 4.24, we see that both LIMEVIM and OcclusionVIM are statistically significantly better than VIM on both Near- and Far-OOD, while GradCAMVIM is better on Near-OOD.

Dataset	AUROC	ΔAUROC VIM	P-value VIM
LIMEVIM			
Near-OOD	70.45	+0.298	0.001 **
Far-OOD	79.16	+0.147	0.001 **
OcclusionVIM			
Near-OOD	70.44	+0.283	0.001 **
Far-OOD	79.08	+0.060	0.001 **
GradCAMVIM			
Near-OOD	70.29	+0.134	0.001 **
Far-OOD	78.97	-0.048	1.000

Table 4.24: Results of performing a Wilcoxon signed-rank test on the AUROC means of against VIM, showing the mean AUROC over 10 runs on CIFAR100, the difference in means compared to the baselines, and the corresponding p-values. Each p-value is appended a significance code which follows the R-standard.

Overall analysis of SaliencyVIM

Overall, the results of integrating XAI saliencies directly into the already developed VIM OOD detector also shows potential, with multiple instances of several methods under this framework beating the baseline method. However, compared to Saliency Aggregation plus Logit, the increase in AUROC scores is much lower, with the highest increase over the baseline across all benchmarks being only 0.580 percentage points. Regardless, the results are statistically significant, and show that integrating saliencies directly into already developed OOD detection methods has potential. It is also interesting to see that the inclusion of occlusion saliencies lead to statistically significant improvements under this framework, when the discriminative power of aggregating occlusion was so low, as reported in section 4.1.

4.3 Summary

In this chapter, I have presented the results from inspecting saliency aggregation over the validation benchmarks, as well results from the three different XAI based OOD detection algorithms I introduced in chapter 3.

In section 4.1, I have analyzed the discriminative power of different saliency aggregation methods on the validation sets of ImageNet200 and CIFAR10. From this, I have found that methods based on the magnitude of saliencies can effectively separate ID and OOD samples. On the contrary, aggregations which measure the statistical spread of saliencies independent of the magnitude, such as CV, RMD or QCD, do not have sufficient discriminative power and are not suitable for OOD detection. In addition, I have found that generating saliencies using occlusion does not lead to aggregations which are suitable for OOD detection. At the end of this section, I analyze the average performance of the different combinations of XAI saliency methods and aggregation functions. Based on this analysis, I found that LIMENorm, GradCAMNorm, IntegratedGradientsMean and GBPNorm showed potential.

In section 4.2 I tested the selected methods on bootstrapped testing benchmarks, and compared the methods to baseline OOD detectors. For Saliency Aggregation and Saliency Aggregation plus Logit, I used the combinations of XAI method and

aggregation which were found to be sufficiently discriminative on the validation sets. For SaliencyVIM, I used LIME, occlusion and GradCAM, as these XAI methods generate saliencies which are suited for integration into the method developed by [11], while GBP and integrated gradients do not.

The tests were conducted on four benchmarks, ImageNet200, CIFAR10, ImageNet1K and CIFAR100. From these tests, I have found that there is definite potential for using XAI methods for OOD detection. Across the first two frameworks (Saliency Aggregation and Saliency Aggregation plus Logit), XAI methods perform well on Far-OOD, surpassing the baselines on many occasions. On Near-OOD, the results are more mixed. Under the SaliencyVIM framework, the results also show potential, with significant results on multiple datasets. In general, XAI methods show better performance on Far-OOD than Near-OOD, but impressive performance is attained in both scenarios on several occasions. GBPNorm+Logit is one such example, performing almost at the level of SoTA methods on CIFAR10, ImageNet200 and CIFAR100.

Chapter 5

Discussion

This chapter will discuss the findings of chapter 4 in more depth and in relation to the problem statement. In addition, I will analyse the results further, attempting to explain some of the tendencies that have been revealed during the experiments.

5.1 Answering the problem statement

As we have now seen, XAI methods can indeed perform OOD detection at a level competitive with OOD detection baselines, and can in some cases even perform close to SoTA models amongst methods which do not retrain the underlying classifying network. The experiments have been conducted on all four OpenOOD OOD detection benchmarks, making the results solid and generalizable. Considering the problem statement, we can conclude that XAI based OOD detection algorithms can perform at a high level, and that the inclusion of XAI saliency maps during OOD detection can lead to higher performance than without them. These results are especially apparent on Far-OOD, where XAI methods beat the baselines by several percentage points on several benchmarks. These results contradict previous research [9], which found that XAI based OOD detection methods had little to offer. Although the results are not SoTA, the frameworks introduced in this thesis are a valuable addition to the field of OOD detection, and could be built upon in many ways, as will be discussed in section 6.3.

5.2 Limitations of the Results

Because the methods have been benchmarked on all four OOD benchmarks included in OpenOOD, the results are relatively solid and generalizable. However, the limitations of OpenOOD itself transfer to the results of this thesis. The experiments have only been done on image classification problems, which limits the number applications the results apply to. Tasks such as segmentation, object detection, regression or natural language processing are not considered in this thesis. In addition, the results only consider semantic OOD shift, as opposed to covariate shift.

5.3 Deeper analysis of the results

In the preceding section, we have answered the problem statement by concluding that XAI based OOD detectors can perform very competitively in several instances. This

chapter will further discuss some of the findings in more detail, and attempt to explain some of the results.

5.3.1 Analysing the reasons for the effectiveness of XAI based OOD detection

The first and most obvious question to ask after having shown that XAI based OOD detectors are effective, is to ask why this is the case. Based on the findings of chapter 4, especially section 4.1, I make the claim that it is primarily the magnitude information collected by XAI methods which make them effective OOD detectors. These results are interesting, because magnitude information is often disregarded in XAI saliency methods. As described in 2.4.3, XAI methods applied to images are usually normalized and displayed visually, often without conveying any information about the magnitude of the saliencies. This may be reasonable when attempting to illuminate the areas which a model is focused on, but is insufficient to discriminate between ID and OOD samples. The results from section 4.1 show this clearly: Here, the statistical dispersion aggregations, which are essentially invariant to the magnitude, perform far worse than the majority of magnitude based aggregations. Furthermore, [9], which used normalized GradCAM heatmaps, attained far worse results than my GradCAM aggregations. The comparatively weak improvements of methods under the SaliencyVIM framework, which uses all saliency values, compared to the saliency aggregation methods further strengthen the idea that positional information in explanations is secondary to magnitude information.

The next question to ask is why XAI saliencies carry such discriminative magnitudal information. From the performance three gradient based methods, it is clear that the gradient information from a given prediction is highly discriminative in an OOD detection context. The value of gradients for OOD detection has already been shown by works such as GradNorm ([58]) and ODIN ([17]), which both use gradients as part of their OOD detection methodology. However, my methods actually outperform GradNorm on every OpenOOD benchmark, and methods under the Saliency Aggregation plus Logit framework sometimes outperform ODIN, showing that the way gradients are used for detection has a large effect on the final results.

Clearly, in the search for explanations which fulfill desirable qualities such as sensitivity, fidelity and implementation invariance, gradient based XAI methods have inadvertently transformed the gradient information of a network in ways which also offer OOD discriminative power. With LIME, [25] also had the stated goals of creating a high fidelity implementation invariant (model independent) XAI method. We can posit that the training of a locally interpretable model on the differences in logits before and after masking extracts saliency magnitudes in a way that is similar to the gradient based XAI methods, although the implementation of the different methods is completely different.

5.3.2 Analysing the difference in performance between magnitude and dispersion aggregation

In section 3.1.1, two intuitions were put explained: The first intuition was that the magnitudes of XAI saliencies could be distributed differently on ID datasets than on OOD datasets. The second intuition was that the statistical dispersion of saliencies were distributed differently. As we have seen from section 4.1, the first intuition seems

to be correct, and in the previous section I attribute most of the performance of the XAI frameworks to the difference in magnitude.

However, statistical dispersion aggregates do not seem to be distributed sufficiently differently between ID and OOD. This is an interesting result. As described in 3, one might intuitively expect that ID samples have high saliency values around regions of interest, while OOD samples have more evenly dispersed values because there are no regions in the image which correspond to the training classes. This would lead ID samples to have a higher dispersion of values. Contrarily, one could imagine that OOD samples lead to XAI saliency maps which are noisy and spread out because XAI methods are not designed to handle OOD data and lack robustness. This would lead to OOD data samples having higher dispersion than ID. However, as we have seen in section 4.1, neither hypothesis is correct, and there is only rarely enough difference in statistical dispersion between ID and OOD to sufficiently discriminate between them. From the results on the validation data sets, the AUROC scores of statistical dispersion are often between 0.50 and 0.60, barely better than pure guessing. In some cases, they may be higher, however there is no clear pattern of either ID or OOD being larger across datasets. Instead, the ID data points may have higher dispersion with a particular XAI method on a particular dataset, while they may be substantially lower than OOD data points on another dataset. This, combined with the low scores overall, show that there is no consistent difference in statistical dispersion across ID and OOD data sets.

To get a better understanding of why this is the case, we may look to some example images from the ImageNet1K benchmark and its associated OOD datasets. In figure 5.1, I show the normalized heatmaps of GradCAM on ImageNet1K and OpenImage_O, one of the Far-OOD datasets from this benchmark.

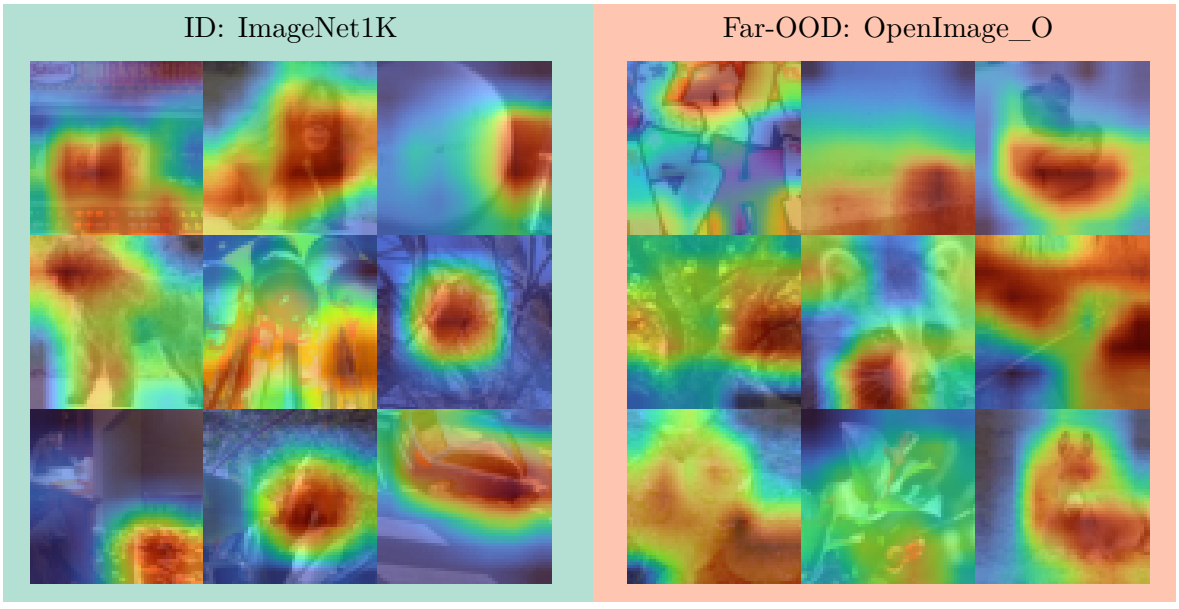

Figure 5.1: Figure showing normalized heatmaps from ImageNet1K and OpenImage_O

As we can see from the figure, there are some clear problems with using statistical dispersion to separate ID and OOD. Firstly, we see that there is a great amount of variation in the position and size of objects of interest in ID data, which in turn makes it less likely that ID data will have a consistent amount of spread. An object of interest may be relatively far away from the camera, leading to only a small region being salient

(with an associated high dispersion), but it may just as easily be very close to the camera, leading a large and spread out saliency. We can see this when comparing the bird in the middle right of the ID images with the dog in the middle left. These investigations concur with the findings of [9], who also pointed to the fact that objects "appear in arbitrary parts of the image and have a high degree of compositional variability" when explaining the poor performance of their heatmap clustering. In addition, the spread of OOD saliency maps does not seem to show any clear pattern, either of being evenly spread out because no objects of interest are present, or of being unstable and highly dispersed due to a lack of robustness in the XAI saliency method when faced with OOD data. Instead, the OOD saliency maps look about the same as the ID saliency maps, and it is hard to find any notable differences in the two.

To contrast, we may look at the same images but without normalizing the saliency values. From figure 5.2, we see a much clearer distinction between ID and OOD data; all of the Far-OOD have considerably lower saliency values. As we have seen from chapter 4, aggregations such as the mean and vector norm have a moderate to strong correlation with the outputs of baseline OOD detection methods such as MLS and MSP, which is reflected in these results.

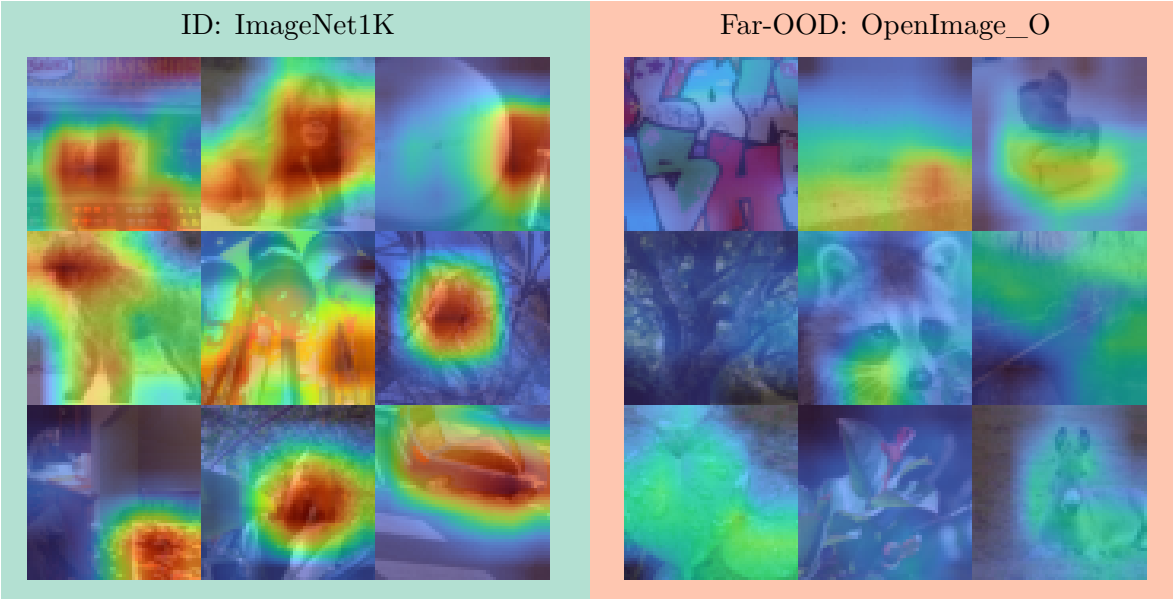

Figure 5.2: Figure showing unnormalized heatmaps from ImageNet1K and OpenImage_O

From these figures, we can see visually how statistical dispersion falls short compared to magnitude aggregation, giving a strong indication as to why these two types of aggregation differ so strongly in discriminative power.

5.3.3 Analysing the poor performance of occlusion saliency aggregation

From the results found in section 4.1, occlusion is an outlier amongst the five XAI methods utilized, and falls consistently behind the other methods. This is somewhat surprising, given the similarity between LIME and occlusion. As explained in section 2.4.4, both LIME and occlusion calculate saliency by occluding parts of the image and looking at the change in confidence of the model. The major difference is that occlusion simply occludes one region at a time and sets the saliency to the drop in confidence,

while LIME calculates saliencies in a more involved fashion where several regions are occluded at once and a new, less complex model is trained to approximate the network.

To understand why the performance of occlusion is poor, we may compare it to LIME, given their similarities. Figure 5.3 shows the unnormalized heatmaps of occlusion and LIME on the same images as in the previous figures.

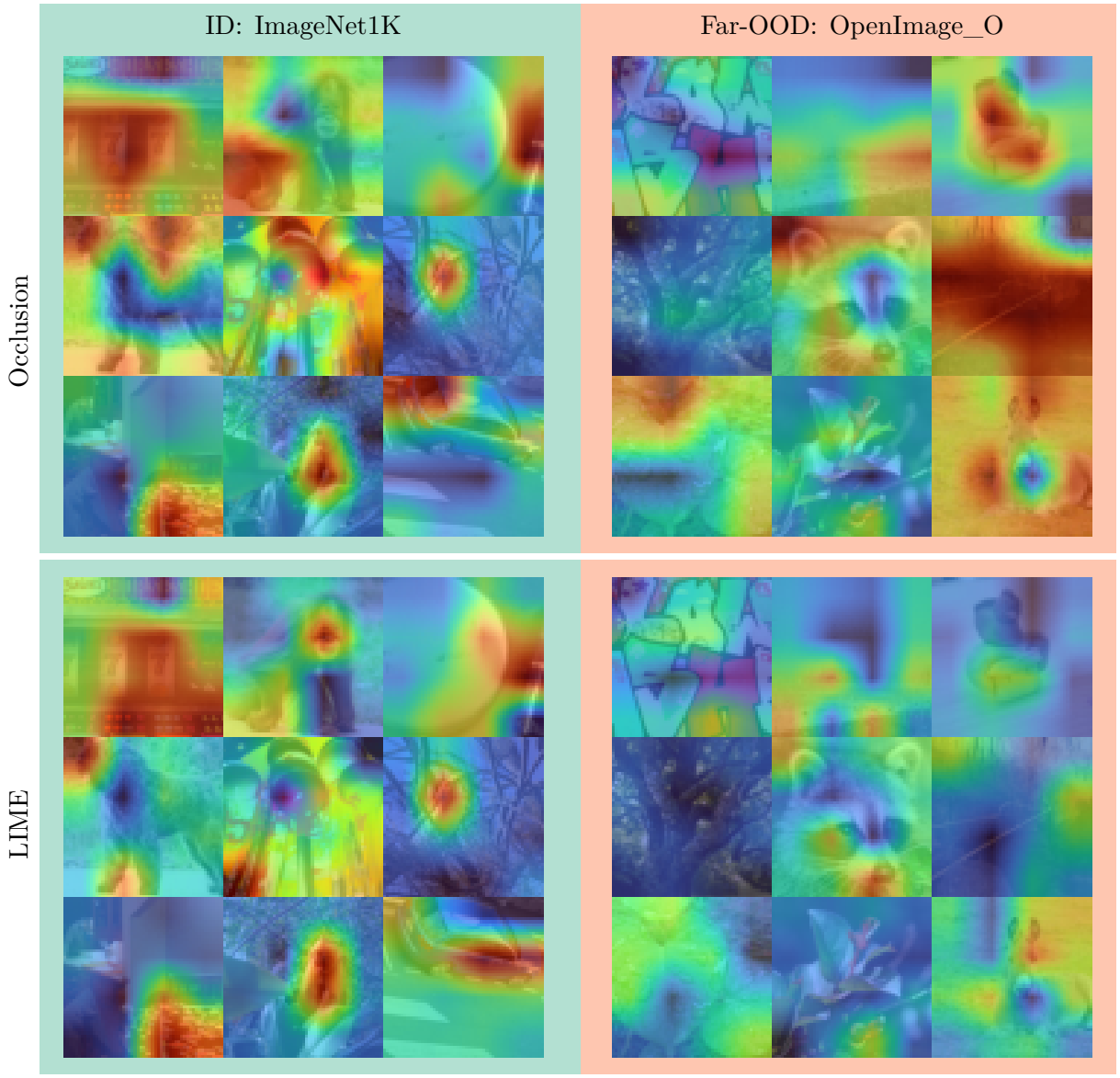

Figure 5.3: Figure showing unnormalized heatmaps from ImageNet1K and OpenImage_O for LIME and occlusion. The heatmaps are displayed using bilinear interpolation of sliding window centers, as opposed to how they are displayed in figure 2.7, because this makes them somewhat easier to interpret

From this figure, we see that on ID data, LIME and occlusion are pretty similar, highlighting about the same regions. However, on OOD data, the occlusion heatmaps are more unstable than LIME. We see that for some images, almost the entire heatmap is filled with high saliency values, and in general the saliencies for OOD data seem to be higher than for LIME. There may be many reasons for why this is case, however, I will

posit one preliminary theory:

Occlusion calculates the saliency for a specific region by calculating the drop in confidence of a single pass on an image where this region is masked. In our case, the mask is a simple rectangular region, and the masking is achieved by replacing all values with a single value (0, which is equivalent to a brownish gray after ResNet normalization). An implicit assumption we have when we use occlusion is that the drop in confidence from this masking represents the network's reaction to the absence of whichever part of the image is masked. However, as [72] has shown, removing information with a simple mask has drastic changes on the distribution of the image, and this assumption may not be correct. For example, an image of a dog in park, whose head has been replaced by a rectangular black square, is not equivalent to an image of a park with no dog present. This means that the drop in confidence does not exclusively represent the absence of the masked region, but also the unpredictable changes that a model may exhibit when large regions of an image are replaced by a constant value. Combining the destructive changes of masking with exposure to OOD data may have unpredictable effects which inhibit occlusion from capturing the differences between ID and OOD data. While LIME uses the same masking strategy, the saliencies returned by LIME are not based on a single forward pass, but based on many permutations of the input image where different regions are masked in tandem. This probably increases the stability of LIME when compared to occlusion, which could explain the difference in performance.

Chapter 6

Conclusion

For the final chapter of this thesis, I give short summary of the thesis, followed by a recapitulation of the main contributions, and finally a description of possible ways forward.

6.1 Thesis summary

This thesis has been an investigation into the potential benefits of using XAI for OOD detection. To investigate this potential, I have developed three different frameworks for utilizing XAI saliencies for OOD detection. These three frameworks are *Saliency Aggregation*, *Saliency Aggregation plus Logit* and *SaliencyVIM*.

Saliency Aggregation is a framework which takes a saliency mapping XAI method and an aggregation function and uses the aggregate of the saliencies as an OOD detection score. Saliency Aggregation plus Logit builds upon the previous framework by combining saliency aggregation with MLS, a baseline OOD detection method. The two metrics are combined by adding their respective Z-scores, inspired by the work of [10]. Finally, SaliencyVIM is a framework which integrated XAI saliencies directly into the OOD detection method VIM. The saliencies are integrated by concatenating them with the features of the penultimate layer, before a PCA is performed on the concatenated matrices. The OOD score is based on the PCA reconstruction error in comparison with the logits.

To perform a selection of methods under these frameworks, an analysis of the statistical qualities of different XAI saliencies has been performed. These analyses have shown that saliency magnitudes are sufficiently differently distributed between ID and OOD datasets to allow for effective OOD detection. The statistical dispersion, on the other hand, is not sufficiently discriminative of OOD data, and should not be used for OOD detection.

Based on the analysis described above, the three frameworks were rigorously tested on all four OOD detection benchmarks included in OpenOOD. Based on these tests, it was found that methods developed under all three frameworks could outperform baseline OOD detection methods. Out of the three frameworks, Saliency Aggregation plus Logit was the best performing, with several methods achieving results which were several percentage points over the baselines, on multiple occasions. In addition, the method GBPNorm+Logit achieved results which were somewhat close to the SoTA on both CIFAR10 and ImageNet1K, showing the definite potential of XAI OOD detection.

6.2 Main contributions

The main contribution of this thesis is the development of three different frameworks for using XAI saliencies for OOD detection. These three frameworks have been rigorously tested on all four OpenOOD OOD detection benchmarks, and all three showcase statistically significant improvements over baseline methods. All are highly general, opening the door for potential improvements when using different XAI methods or in combination with different OOD detectors.

Secondarily, comprehensive analysis has also been done on the statistical qualities of XAI saliency maps on ID and OOD data (chapter 4 section 4.1). These findings give deep insight into how XAI methods behave when faced with OOD data, and show that while the statistical dispersion of saliencies is not very different between ID and OOD data, the magnitude of saliencies is. These findings could be used to further explore the integration of XAI and OOD detection.

6.3 Future work

Although comprehensive investigations and tests have been conducted as part of this thesis, there are many avenues which could be explored further. As I have introduced general frameworks for integrating XAI into OOD detection, as opposed to specific methods, there is great potential for further studies into different combinations of XAI methods, aggregations and other OOD detection metrics. For example, one could perform saliency aggregation on other XAI methods such as GradCAMPlusPlus [35], Shapley [22] or DeepLift [73]. In addition, instead of combining saliency aggregation with logits, one could combine it with any other OOD detection metric, such as MSP, the energy score as described in [41] or the modified softmax score as described in [17]. To further explore the integration of saliencies directly into OOD detection methods, other methods than VIM can be used, for example SCALE [74] or other OOD detection methods which work on logits.

As described in section 1.3, the scope of this thesis has also restricted the types of OOD detection taxonomies which are used. Notably, the lack of methods which retrain the network excludes data augmentation methods such as PixMix [50] and RotPred [49]. Both of these methods are used in combination with OOD detectors to great effect, achieving SoTA results on CIFAR10 and ImageNet200. A possible further path of research could then be to expand the scope and explore how XAI methods could be used in pre-training as well.

Finally, the hyperparameters chosen for each XAI method have been fixed in this thesis, as opposed to being chosen through hyperparameter tuning. This has been done to reduce the computational burden, but also represents a restriction of the scope of the testing that has been performed. In future work, different sliding window dimensions or segmentation methods could be chosen for LIME and occlusion, different layers could be chosen with GradCAM and different baselines could be used with integrated gradients. Performing such experiments could further illuminate the robustness and potential of using these methods for OOD detection.

As we can see from the preceding paragraphs, this thesis is merely a first step towards integrating XAI into OOD detection. The three frameworks that have been developed and the analysis that has been conducted as part of this thesis lays the groundwork for further research in a field which is essentially unexplored.

Bibliography

[1] Shaveta Dargan et al. 'A Survey of Deep Learning and Its Applications: A New Paradigm to Machine Learning'. In: *Archives of Computational Methods in Engineering* 27.4 (Sept. 2020), pp. 1071–1092. ISSN: 1886-1784. DOI: 10.1007/s11831-019-09344-w. URL: https://doi.org/10.1007/s11831-019-09344-w.

[2] Sajid Nazir, Diane M. Dickson and Muhammad Usman Akram. 'Survey of explainable artificial intelligence techniques for biomedical imaging with deep neural networks'. In: *Computers in biology and medicine* 156 (2023), p. 106668. URL: https://api.semanticscholar.org/CorpusID:257067347.

[3] Alvin Rajkomar, Jeffrey Dean and Isaac Kohane. 'Machine Learning in Medicine'. In: *New England Journal of Medicine* 380.14 (2019), pp. 1347–1358. DOI: 10.1056/NEJMra1814259. eprint: https://www.nejm.org/doi/pdf/10.1056/NEJMra1814259. URL: https://www.nejm.org/doi/full/10.1056/NEJMra1814259.

[4] Jordan Zheng Ting Sim et al. 'Machine learning in medicine: what clinicians should know'. en. In: *Singapore Med J* 64.2 (May 2021), pp. 91–97.

[5] Christian Szegedy et al. *Intriguing properties of neural networks*. 2014. arXiv: 1312.6199 [cs.CV].

[6] Jingyang Zhang et al. 'OpenOOD v1.5: Enhanced Benchmark for Out-of-Distribution Detection'. In: *arXiv preprint arXiv:2306.09301* (2023).

[7] Jingyang Zhang et al. 'OpenOOD v1.5: Enhanced Benchmark for Out-of-Distribution Detection'. In: *arXiv preprint arXiv:2306.09301* (2023).

[8] Peter J. Denning et al. 'Computing as a discipline'. In: *Computer* 22.2 (1989), pp. 63–70.

[9] Aitor Martinez-Seras, Javier Del Ser and Pablo Garcia-Bringas. 'Can Post-hoc Explanations Effectively Detect Out-of-Distribution Samples?' In: *2022 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. 2022, pp. 1–9. DOI: 10.1109/FUZZ-IEEE55066.2022.9882726.

[10] Shashi Shekhar et al. *Proceedings of the 2024 SIAM International Conference on Data Mining (SDM)*. Ed. by Shashi Shekhar et al. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2024. DOI: 10.1137/1.9781611978032. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611978032. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611978032.

[11] Haoqi Wang et al. *ViM: Out-Of-Distribution with Virtual-logit Matching*. 2022. arXiv: 2203.10807 [cs.CV].

[12] Tom M Mitchell. *Machine learning*. Vol. 1. 9. McGraw-hill New York, 1997.

[13] F. Rosenblatt. 'The perceptron: A probabilistic model for information storage and organization in the brain.' In: *Psychological Review* 65.6 (1958), pp. 386–408. ISSN: 0033-295X. DOI: 10.1037/h0042519. URL: http://dx.doi.org/10.1037/h0042519.

[14] George V. Cybenko. 'Approximation by superpositions of a sigmoidal function'. In: *Mathematics of Control, Signals and Systems* 2 (1989), pp. 303–314. URL: https://api.semanticscholar.org/CorpusID:3958369.

[15] Yann Lecun et al. 'Gradient-Based Learning Applied to Document Recognition'. In: *Proceedings of the IEEE* 86 (Dec. 1998), pp. 2278–2324. DOI: 10.1109/5.726791.

[16] Dan Hendrycks and Kevin Gimpel. *A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks*. 2018. arXiv: 1610.02136 [cs.NE].

[17] Shiyu Liang, Yixuan Li and R. Srikant. *Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks*. 2020. arXiv: 1706.02690 [cs.LG].

[18] Jingkang Yang et al. *Generalized Out-of-Distribution Detection: A Survey*. 2024. arXiv: 2110.11334 [cs.CV].

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

[20] European Union. *Article 71: European Data Protection Board*. Accessed: February 13, 2024. 2016. URL: https://www.privacy-regulation.eu/en/r71.htm.

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

[22] Scott Lundberg and Su-In Lee. *A Unified Approach to Interpreting Model Predictions*. 2017. arXiv: 1705.07874 [cs.AI]. URL: https://arxiv.org/abs/1705.07874.

[23] Bas H.M. van der Velden et al. 'Explainable artificial intelligence (XAI) in deep learning-based medical image analysis'. In: *Medical Image Analysis* 79 (2022), p. 102470. ISSN: 1361-8415. DOI: https://doi.org/10.1016/j.media.2022.102470. URL: https://www.sciencedirect.com/science/article/pii/S1361841522001177.

[24] Radhakrishna Achanta et al. 'SLIC Superpixels Compared to State-of-the-Art Superpixel Methods'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.11 (2012), pp. 2274–2282. DOI: 10.1109/TPAMI.2012.120.

[25] Marco Tulio Ribeiro, Sameer Singh and Carlos Guestrin. *"Why Should I Trust You?": Explaining the Predictions of Any Classifier*. 2016. arXiv: 1602.04938 [cs.LG]. URL: https://arxiv.org/abs/1602.04938.

[26] Matthew D. Zeiler and Rob Fergus. 'Visualizing and Understanding Convolutional Networks'. In: *Computer Vision – ECCV 2014*. Ed. by David Fleet et al. Cham: Springer International Publishing, 2014, pp. 818–833. ISBN: 978-3-319-10590-1.

[27] Håvard Horgen Thunold et al. 'A Deep Diagnostic Framework Using Explainable Artificial Intelligence and Clustering'. In: *Diagnostics* 13.22 (2023). ISSN: 2075-4418. DOI: 10.3390/diagnostics13223413. URL: https://www.mdpi.com/2075-4418/13/22/3413.

[28] Bolei Zhou et al. *Learning Deep Features for Discriminative Localization*. 2015. arXiv: 1512.04150 [cs.CV].

[29] Ramprasaath R. Selvaraju et al. 'Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization'. In: *International Journal of Computer Vision* 128.2 (Oct. 2019), pp. 336–359. ISSN: 1573-1405. DOI: 10.1007/s11263-019-01228-7. URL: http://dx.doi.org/10.1007/s11263-019-01228-7.

[30] Jost Tobias Springenberg et al. *Striving for Simplicity: The All Convolutional Net*. 2015. arXiv: 1412.6806 [cs.LG]. URL: https://arxiv.org/abs/1412.6806.

[31] Leila Arras, Ahmed Osman and Wojciech Samek. 'CLEVR-XAI: A benchmark dataset for the ground truth evaluation of neural network explanations'. In: *Information Fusion* 81 (2022), pp. 14–40.

[32] Theerasarn Pianpanit et al. 'Parkinson's disease recognition using SPECT image and interpretable AI: A tutorial'. In: *IEEE Sensors Journal* 21.20 (2021), pp. 22304–22316.

[33] Mukund Sundararajan, Ankur Taly and Qiqi Yan. *Axiomatic Attribution for Deep Networks*. 2017. arXiv: 1703.01365 [cs.LG]. URL: https://arxiv.org/abs/1703.01365.

[34] Sebastian Bach et al. 'On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation'. In: *PLOS ONE* 10.7 (July 2015), pp. 1–46. DOI: 10.1371/journal.pone.0130140. URL: https://doi.org/10.1371/journal.pone.0130140.

[35] Aditya Chattopadhay et al. 'Grad-CAM++: Generalized Gradient-Based Visual Explanations for Deep Convolutional Networks'. In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, Mar. 2018. DOI: 10.1109/wacv.2018.00097. URL: http://dx.doi.org/10.1109/WACV.2018.00097.

[36] Rachel Lea Draelos and Lawrence Carin. *Use HiResCAM instead of Grad-CAM for faithful explanations of convolutional neural networks*. 2021. arXiv: 2011.08891 [eess.IV]. URL: https://arxiv.org/abs/2011.08891.

[37] Jerome H Friedman. 'Greedy function approximation: a gradient boosting machine'. In: *Annals of statistics* (2001), pp. 1189–1232.

[38] Daniel W Apley and Jingyu Zhu. 'Visualizing the effects of predictor variables in black box supervised learning models'. In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 82.4 (2020), pp. 1059–1086.

[39] Ian J. Goodfellow, Jonathon Shlens and Christian Szegedy. *Explaining and Harnessing Adversarial Examples*. 2015. arXiv: 1412.6572 [stat.ML].

[40] Peng Cui and Jinjia Wang. 'Out-of-Distribution (OOD) Detection Based on Deep Learning: A Review'. In: *Electronics* 11.21 (2022). ISSN: 2079-9292. DOI: 10.3390/electronics11213500. URL: https://www.mdpi.com/2079-9292/11/21/3500.

[41] Weitang Liu et al. *Energy-based Out-of-distribution Detection*. 2021. arXiv: 2010.03759 [cs.LG].

[42] Dan Hendrycks et al. *Scaling Out-of-Distribution Detection for Real-World Settings*. 2022. arXiv: 1911.11132 [cs.CV]. URL: https://arxiv.org/abs/1911.11132.

[43] Matthew Cook, Alina Zare and Paul Gader. *Outlier Detection through Null Space Analysis of Neural Networks*. 2020. arXiv: 2007.01263 [cs.LG].

[44] Ibrahima Ndiour, Nilesh Ahuja and Omesh Tickoo. *Out-Of-Distribution Detection With Subspace Techniques And Probabilistic Modeling Of Features*. 2020. arXiv: 2012.04250 [cs.LG].

[45] Yiyou Sun et al. *Out-of-Distribution Detection with Deep Nearest Neighbors*. 2022. arXiv: 2204.06507 [cs.LG]. URL: https://arxiv.org/abs/2204.06507.

[46] Kimin Lee et al. *A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks*. 2018. arXiv: 1807.03888 [stat.ML]. URL: https://arxiv.org/abs/1807.03888.

[47] Sudarshan Regmi. *AdaSCALE: Adaptive Scaling for OOD Detection*. 2025. arXiv: 2503.08023 [cs.CV]. URL: https://arxiv.org/abs/2503.08023.

[48] Andrija Djurisic et al. 'Extremely Simple Activation Shaping for Out-of-Distribution Detection'. In: *The Eleventh International Conference on Learning Representations*. 2023. URL: https://openreview.net/forum?id=ndYXTEL6cZz.

[49] Dan Hendrycks et al. 'Using Self-Supervised Learning Can Improve Model Robustness and Uncertainty'. In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/a2b15837edac15df90721968986f7f8e-Paper.pdf.

[50] Dan Hendrycks et al. 'PixMix: Dreamlike Pictures Comprehensively Improve Safety Measures'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 16783–16792.

[51] Eoin Delaney, Derek Greene and Mark T. Keane. *Uncertainty Estimation and Out-of-Distribution Detection for Counterfactual Explanations: Pitfalls and Solutions*. 2021. arXiv: 2107.09734 [cs.LG]. URL: https://arxiv.org/abs/2107.09734.

[52] John Sipple and Abdou Youssef. 'A General-Purpose Method for Applying Explainable AI for Anomaly Detection'. In: *Foundations of Intelligent Systems*. Ed. by Michelangelo Ceci et al. Cham: Springer International Publishing, 2022, pp. 162–174. ISBN: 978-3-031-16564-1.

[53] AJ Tallón-Ballesteros and C Chen. 'Explainable AI: Using Shapley Value to Explain Complex Anomaly Detection ML-Based Systems'. In: *Machine Learning and Artificial Intelligence: Proceedings of MLIS 2020* 332 (2020), p. 152.

[54] Osvaldo Arreche et al. 'E-XAI: Evaluating Black-Box Explainable AI Frameworks for Network Intrusion Detection'. In: *IEEE Access* 12 (2024), pp. 23954–23988. DOI: 10.1109/ACCESS.2024.3365140.

[55] Basim Mahbooba et al. 'Explainable Artificial Intelligence (XAI) to Enhance Trust Management in Intrusion Detection Systems Using Decision Tree Model'. In: *Complexity* 2021.1 (2021), p. 6634811. DOI: https://doi.org/10.1155/2021/6634811. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1155/2021/6634811. URL: https://onlinelibrary.wiley.com/doi/abs/10.1155/2021/6634811.

[56] Erzhena Tcydenova et al. 'Detection of adversarial attacks in AI-based intrusion detection systems using explainable AI'. In: *Human-Centric Comput Inform Sci* 11 (2021).

[57] Tahmina Zebin, Shahadate Rezvy and Yuan Luo. 'An Explainable AI-Based Intrusion Detection System for DNS Over HTTPS (DoH) Attacks'. In: *IEEE Transactions on Information Forensics and Security* 17 (2022), pp. 2339–2349. DOI: 10.1109/TIFS.2022.3183390.

[58] Rui Huang, Andrew Geng and Yixuan Li. *On the Importance of Gradients for Detecting Distributional Shifts in the Wild*. 2021. arXiv: 2110.00218 [cs.LG].

[59] Anh Nguyen, Jason Yosinski and Jeff Clune. *Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images*. 2015. arXiv: 1412.1897 [cs.CV]. URL: https://arxiv.org/abs/1412.1897.

[60] Robert Geirhos et al. 'Shortcut learning in deep neural networks'. In: *Nature Machine Intelligence* 2.11 (2020), pp. 665–673.

[61] Ian J. Goodfellow, Jonathon Shlens and Christian Szegedy. *Explaining and Harnessing Adversarial Examples*. 2015. arXiv: 1412.6572 [stat.ML]. URL: https://arxiv.org/abs/1412.6572.

[62] Alex Krizhevsky. 'Learning Multiple Layers of Features from Tiny Images'. In: (2009), pp. 32–33. URL: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[63] Samira Pouyanfar et al. 'A survey on deep learning: Algorithms, techniques, and applications'. In: *ACM computing surveys (CSUR)* 51.5 (2018), pp. 1–36.

[64] Julian Bitterwolf, Maximilian Mueller and Matthias Hein. 'In or Out? Fixing ImageNet Out-of-Distribution Detection Evaluation'. In: *ICML*. 2023. URL: https://proceedings.mlr.press/v202/bitterwolf23a.html.

[65] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV]. URL: https://arxiv.org/abs/1512.03385.

[66] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV]. URL: https://arxiv.org/abs/2010.11929.

[67] Walid Bousselham et al. *LeGrad: An Explainability Method for Vision Transformers via Feature Formation Sensitivity*. 2025. arXiv: 2404.03214 [cs.CV]. URL: https://arxiv.org/abs/2404.03214.

[68] Pascal Sturmfels, Scott Lundberg and Su-In Lee. 'Visualizing the Impact of Feature Attribution Baselines'. In: *Distill* (2020). https://distill.pub/2020/attribution-baselines. DOI: 10.23915/distill.00022.

[69] Giang Nguyen et al. 'Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey'. In: *Artificial Intelligence Review* 52 (2019), pp. 77–124.

[70] Narine Kokhlikyan et al. *Captum: A unified and generic model interpretability library for PyTorch*. 2020. arXiv: 2009.07896 [cs.LG].

[71] Jacob Gildenblat and contributors. *PyTorch library for CAM methods*. https://github.com/jacobgil/pytorch-grad-cam. 2021.

[72] Sara Hooker et al. *A Benchmark for Interpretability Methods in Deep Neural Networks*. 2019. arXiv: 1806.10758 [cs.LG]. URL: https://arxiv.org/abs/1806.10758.

[73] Avanti Shrikumar, Peyton Greenside and Anshul Kundaje. *Learning Important Features Through Propagating Activation Differences*. 2019. arXiv: 1704.02685 [cs.CV]. URL: https://arxiv.org/abs/1704.02685.

[74] Kai Xu et al. 'Scaling for Training Time and Post-hoc Out-of-distribution Detection Enhancement'. In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=RDSTjtnqCg.