DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Information Systems

Out-of-Distribution detection using One-vs-All Classifiers

Daniel Korth

DEPARTMENT OF INFORMATICS

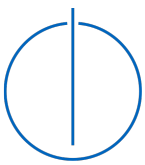
TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Information Systems

Out-of-Distribution detection using One-vs-All Classifiers

Erkennung von Daten außerhalb der Verteilung mit Eins-gegen-Alle Klassifikatoren

Author:	Daniel Korth
Supervisor:	PD Dr. Tobias Lasser
Advisor:	Alessandro Wollek, M.Sc.
Submission Date:	15.08.2022



I confirm that this bachelor's thesis in information systems is my own work and I have documented all sources and material used.

Munich, 15.08.2022 Daniel Korth

Acknowledgments

First and foremost, I want to thank my advisor Alessando Wollek for his continuous support throughout my thesis. His guidance and feedback was very valuable and helped me a lot.

I'd also like to express my gratitude to PD Dr. Tobias Lasser for giving me the opportunity to do research at his group and providing me with all the necessary hardware resources.

Finally I want to thank my family, friends and everyone else who supported me throughout this thesis.

Abstract

Detecting whether an input is In-Distribution (ID) or Out-of-Distribution (OOD) is essential for any machine learning algorithm deployed to the open world. Yet many state-of-the-art deep neural networks struggle to identify OOD inputs. The consequences can range from misclassification in autonomous vehicles to misdiagnosis and mistreatment in a medical setting. This thesis investigates how One-vs-All (OVA) Classifiers can help detect OOD samples and proposes a novel OOD detection approach. The key idea is to train separate modules that can extract class-specific features and are responsible for filtering out OOD data. For effective training, outlier data is generated and used. The approach is termed One-vs-All Filtering (OVAF) and is applicable to most neural network classifiers. It reduces the False Positive Rate by up to 47.2% compared to the baseline approach. This thesis bases its analysis on the setting of image classification, but adaptation to other domains is straightforward.

Contents

1 Introduction

In recent years, machine learning dramatically improved in computer vision [1], natural language processing [2] and many other areas. This has led to the deployment of neural networks in various environments, for example, autonomous driving or medical imaging. Despite their incredible performance on tasks that they were trained for, they still struggle when receiving and evaluating abnormal inputs [3], [4]. Ideally, the model should detect Out-of-Distribution (OOD) inputs, not classify them, and raise a flag indicating that it saw an abnormal input.
Consider a made-up example case of autonomous driving where a vehicle is heading towards a snowman - something the model has not seen in the training data and does not know how to classify. Instead of detecting the snowman as OOD and, for example, alerting the driver to take control of the car, it mistakenly classifies it as a human, hits the brakes and causes a rear-end collision.
The main challenge for OOD detection is that it is impossible to explicitly teach a model what kind of OOD inputs it will receive and how to handle them, especially when it operates in an uncontrolled environment, where the types of images it might receive are unlimited. At the same time, it is still an essential requirement for any safety-critical application deployed in the open world.

1.1 Closed-World Assumption and Softmax

In image classification, neural networks are trained on pairs of images and labels and must find a mapping from image to label. Since the number of labels is fixed, the model operates in a world where only In-Distribution (ID) labels exist. This leads to the model being trained on a closed-world assumption [5], meaning that the model's world is limited to these labels. It believes everything it sees belongs to one of the labels it learned, and no other labels exist. This assumption works well when an input corresponds to one of the known labels; however, the model is still forced to make a decision even when none of the labels apply - for example, in the case of OOD inputs.
To get a better intuition, the picture in Figure 1.1 depicts a small example of a neural network classifier trained on a 2-D dataset under closed-world assumption with softmax normalization. The ten colored clusters of points represent the different classes to predict, and the background coloring shows the network's confidence. Higher

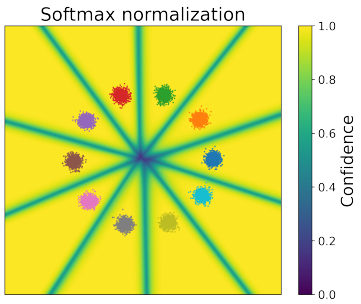


Figure 1.1: Visualization of the confidence landscape of a neural network. Clusters of colored points represent the different classes. 3-Layer Multi-layer Perceptron (MLP) trained with cross-entropy loss and softmax normalization. The model achieves a accuracy of 100%. High confidences remain even far away from the clusters.

probabilities (yellow) mean that the model is more confident that it knows to what class the point belongs. One can see that the model is not only confident in and around the area of the 10 clusters it was trained on but instead separates the entire 2-D plane into ten sections, one for each cluster. That means that the model is still highly confident that a point from the bottom left of the image belongs to class Pink, even though it is far away from the cluster of Pink points.
On the other side, the closed-world assumption simplifies the task of classification because the model must only separate the input plane into regions for the different classes instead of being able to fit perfect confidence regions around the clusters.

1.2 One-vs-All Classifiers

A method that has been around for a very long time is One-vs-All (OVA) classifiers. Instead of creating a single model that maps an input to one of the classes, OVA classifiers split the problem into several binary tasks. Each task is responsible for only one class and has to decide whether the input belongs to that class. In principle, this avoids the closed-world assumption [5], as it does not force that an input belongs to one of the known classes. Every classifier independently evaluates whether an input belongs to its class or not. It is ultimately possible that no classifier predicts an input to belong to its class, which is a natural way of saying that the input is OOD. Despite these properties and as the right image in Figure 1.2 reveals, the classifier's confidence is still not very focused on the red cluster, for which it has been trained. The high-confidence

region extends far into the top of the image, where OOD inputs could be located. Yet the conceptual idea of OVA is promising; it is not confident that any of the other clusters belongs to class red.

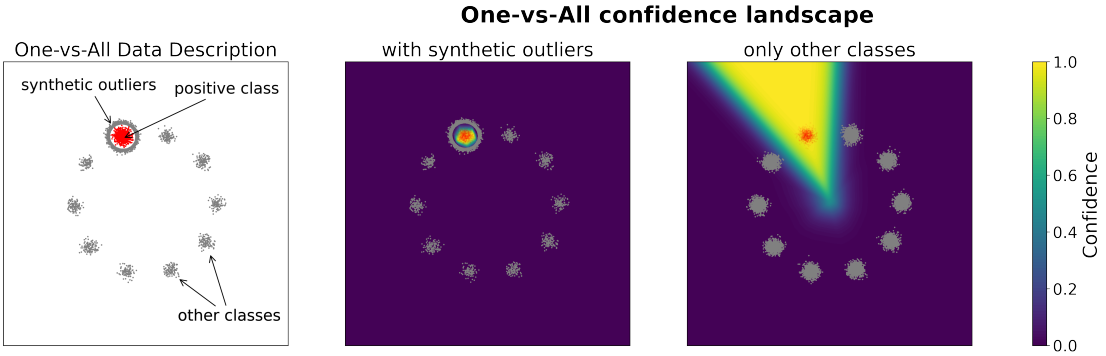


Figure 1.2: OVA technique and confidence landscape. The **left** image describes the different data types for OVA classifiers. Red denotes the positive class, grey the negative class. 3-Layer MLP trained to predict class Red; in the **middle** both with other classes and synthetic outliers as negative samples. On the **right** only with other classes. The addition of synthetic outliers narrows the high-confidence region of the classifier to only focus on the positive class and not extend far away from the distribution of red points.

1.3 Training with Out-of-Distribution Data

One of the best ways to make a model aware of OOD data is to provide such during training [3]. This is, however, challenging as a dataset can not represent the whole OOD landscape and can be costly to obtain. To solve the latter issue, researchers have tried other method to create outliers artificially. Approaches for example include CLIP [6] which can convert textual inputs into images or Autoencoders [7], which learn a latent representation of each class and then decode the representation into an image again.
This thesis focuses on generating additional data by sampling from a gaussian distribution. In particular, it samples from the low-density region of a distribution that resembles ID images. Such samples are referred to as synthetic outliers. Figure 1.2 describes the synthetic outliers. The left picture shows how synthetic outliers might look like in a simple 2-d case. The middle and right images depict small neural network classifiers trained to detect class red; the right one was trained in the conventional OVA way, and the middle used synthetic outliers additionally. One can observe that the synthetic outliers narrow the confidence band of the classifier, which is beneficial for

OOD detection. The main challenge is to replicate this behavior in high-dimensional space, where perfectly surrounding the class is almost impossible due to the curse of dimensionality. One of the central questions of this thesis, therefore, becomes:
How can one create a diverse set of outliers that effectively narrows the high-confidence regions of a classifier?

1.4 OVAF - One-vs-All Filtering

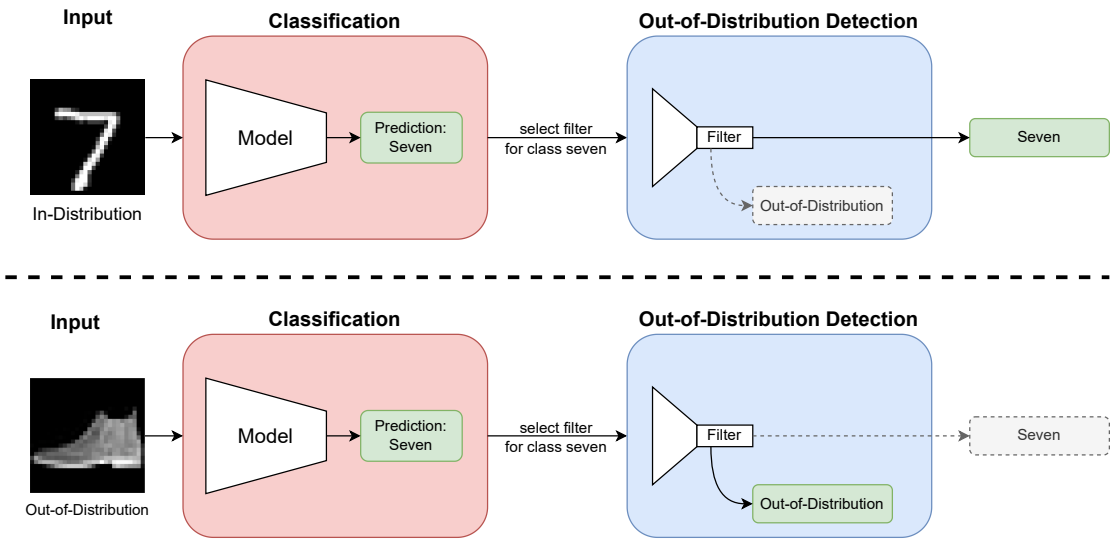


Figure 1.3: Conceptual description of OVAF. Example Task: classify handwritten digits. The algorithm receives input and predicts its class. Then the class-specific Filter is chosen to validate the decision (**top**) or detect and filter out an outlier input (**bottom**)

This thesis proposes a method for detecting OOD images termed **O**ne-**V**s-**A**ll **F**iltering (OVAF). It combines all observations made above. It acknowledges that training under the closed-world assumption is effective and performant for classification. Additionally, it uses OVA components that do not suffer from the closed-world training to detect OOD data. These components are called Filtering Heads; one exists per ID class, and its goal is to filter out the OOD inputs. OVAF uses synthetic outliers to teach the heads what actual outliers in the open world might look like and narrow the classifier's high-confidence region.
The conceptual idea of OVAF is depicted in Figure 1.3. The model receives input and

classifies it. Then a class-specific Filtering Head is chosen to either validate the class or detect an OOD input that is then filtered away.

1.5 Research Objective and Main Contributions

The main research objective of this thesis is to investigate the underexplored area of OVA classifiers for OOD detection. Based on the findings, a new method for OOD detection should be developed that uses the concept of OVA.
The key contributions of this thesis include:

- proposing a novel and flexible method to detect OOD inputs that is applicable to almost any pre-trained neural network classifier
- investigating strengths and weaknesses of using OVA classifiers for OOD detection
- studying and comparing the effectiveness of synthetic and real-world OOD data for fine-tuning networks to detect outliers

1.6 Outline

The rest of the thesis is organized as follows:

- chapter 2 introduces notation and necessary concepts as well as providing information on OOD detection and OVA classifiers
- chapter 3 describes a few relevant OOD detection methods in depth that are also used later chapters
- chapter 4 explains the proposed OVAF method both at training and inference time
- chapter 5 sets up the experiments used to evaluate performance of OVAF
- chapter 6 shows the empirical results from the different experiments
- chapter 7 discusses the experiments and suggests areas of future work
- chapter 8 concludes the thesis and summarizes key findings

2 Background

This section first introduces terminology and defines the problem of OOD detection. It presents three widespread activation functions and competitive neural network architectures used repeatedly in the thesis. Afterward, it provides additional background information on OOD detection and OVA classifiers.

2.1 Problem Setup and Terminology

Consider an image classification problem. Given an ID dataset $\mathcal{D}_{\text{in}}$ consisting of images x and labels y, one tries to classify each image $x_i \in \mathbb{R}^n$ with the corresponding label $y_i \in \{1, \ldots, C\}$. Each image belongs to exactly 1 out of the C different classes. A classifier $f(x) : \mathbb{R}^n \to \mathbb{R}^C$ is trained on the ID dataset to solve the image classification task. At inference, the model $f(x)$ is evaluated on both $\mathcal{D}_{in}$ and $\mathcal{D}_{out} = \mathcal{D} \setminus \mathcal{D}_{in}$. The latter represents the OOD data, everything that does not belong to one of the ID classes. The general goal of OOD detection is to have a high ID accuracy while detecting and filtering out the OOD data that is not classifiable for the model. In general $g(x)$ will denote the OOD detector, and $h(x) : \mathbb{R}^n \to \mathbb{R}^m$ with $m \ll n$ a mapping from the input space to a reduced hidden space. The hidden space, sometimes also referred to as feature space, is considered as any layer that is neither the input space $\mathbb{R}^n$ nor the output space $\mathbb{R}^C$. Generally, those hidden space features are a dense representation of the input; in this thesis mostly the penultimate layer activations. In many cases, the neural network $f(x)$ serves as a basis for fine-tuning the methods to detect OOD inputs. In such cases, $f(x)$ is referred to as the base or backbone model. OOD samples can come from many different sources. This thesis mainly distinguishes synthetic outliers, which are generated using mathematical techniques, and real, naturally occurring outliers such as images of clothes when the training dataset consists of digits.

2.2 Activation Functions

Activation functions are essential to neural networks due to their non-linearity. Without them, one could summarize a neural network with just a single matrix multiplication.

2.2.1 Softmax

The softmax is the most widespread activation function for multi-class classification. It normalizes and transforms an input vector into a vector that can be interpreted from a probabilistic point of view. Let $z \in \mathbb{R}^C$ with $z = (z_1, z_2, \ldots, z_C)$. The softmax $S(z) : \mathbb{R}^C \to \mathbb{R}^C$ transforms each point individually, using the following operation

$$S_i(z) = \frac{\exp(z_i)}{\sum_{j=1}^{C} \exp(z_j)} \tag{2.1}$$

$S_i(z)$ can be interpreted as the probability of input z belonging to class i.

2.2.2 Sigmoid

The sigmoid activation function is used for binary decisions. Binary decisions have a positive and negative class, often represented with O and 1. The input to the sigmoid function $\sigma(z) : \mathbb{R} \to \mathbb{R}$ is a single scalar, and it is transformed and mapped to a probability in the following way:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

$\sigma(z)$ can be used to represent the probability of z belonging to the positive class, so class 1.

In general, the sigmoid, and especially the softmax, is used in the ultimate layer of a neural network $f(x)$ to transform the outputs into interpretable probabilities. The outputs of $f(x)$ is referred to as logits, and $f_i(x)$ represents the i-th logit in the model. The softmax and sigmoid activation functions therefore receive the logits of the neural network to make a decision on which class to choose.

2.2.3 ReLU

The rectified linear unit (ReLU) [8] is a popular activation function in neural networks, that is mostly used in the hidden layers. It sets all negative inputs to 0 and has an identity mapping for positive inputs, mathematically:

$$\text{ReLU}(z) = \begin{cases} z & \text{if } z > 0 \\ 0 & \text{if } z <= 0 \end{cases}$$

Here, z represents a single scalar value; extension to vectors works by applying the function element-wise. One of the main advantages of ReLU is that the gradient is 1 if the input was greater than 0, mitigating the issue of vanishing gradients during backpropagation [8].

2.3 Model Architectures

The literature uses different model architectures to evaluate the performance of OOD detection. Two of the most common architectures are DenseNet [9] and WideResNet [10], both of which extend and build upon the ResNet [11] architecture. They are described below.

2.3.1 ResNet

Deep neural networks are a sequence of linear connections and non-linear activation functions. A layer can be described as

$$y = N(Wx + b)$$

where y is the output of the layer, W and b are the weights and bias of the layer, N is a non-linear activation function, and x is the input to the layer. Several of these layers can be stacked together to form a residual block, denoted by $\mathcal{F}$. The regular ResNet [11] architecture adds a skip-connection between two blocks that by-passes the non-linearity in the following way:

$$x_\ell = \mathcal{F}_\ell(x_{\ell-1}) + x_{\ell-1}$$

where ℓ describes the layer of the block in the network. Figure 2.1 visualizes a residual block. The skip-connection allows the gradient to flow directly through the network during backpropagation and mitigates the vanishing gradient problem for deeper networks, where early layers are not updated anymore since the gradient is almost zero [11]. ResNets consist of two different building blocks that use the skip-connection:

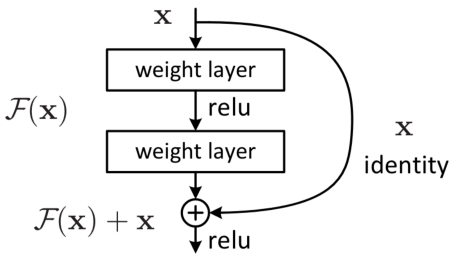


Figure 2.1: Residual block [11]

- **Basic** blocks for feature extraction that consist of two 3×3 convolutions, batch normalization [12] and the ReLU activation function [8]
- **Bottleneck** block for dimensionality reduction that consist of 1×1, 3×3 and again 1×1 convolutions

2.3.2 DenseNet

DenseNets [9] extends the original idea of skip-connections by connecting the individual layers not only to one but multiple preceding layers:

$$x_\ell = H_\ell\left([x_0, x_1, \ldots, x_{\ell-1}]\right)$$

The layers $x_0, x_1, \ldots, x_{\ell-1}$ are concatenated into a single tensor. The authors define $H_\ell(\cdot)$ as a composition of batch normalization [12], ReLU [8] and a 3×3 convolution and refer to it as a dense block (see also Figure 2.2).

In between the different dense blocks, they put transition layers responsible for down-sampling and reducing the size of the so-called feature map, which basically describes how many layers of features are currently available. The transition layer consists of batch normalization, a 1×1 convolution, and a 2×2 average pooling layer. Another important concept is the growth rate k, which controls the size of the feature maps. As each convolutional layer produces k feature maps, the number of maps for the l-th layer generalizes to:

$$k_\ell = k_0 + k * (l - 1)$$

G. Huang *et al.* [9] propose two additional architectural changes, bottlenecks and compressions, which they denote with **DenseNet-B** and **DenseNet-C**.
DenseNet-B leverages 1×1 convolutions to reduce the size of the feature map before the 3×3 convolutions in the dense blocks.
DenseNet-C further reduces the model size by reducing the feature maps in the transition layers. Assuming the dense block contains m feature maps, then the transition layers generate θm feature maps in the output. θ refers to the compression rate and is between 0 and 1.
The two concepts mentioned above can be combined to create the **DenseNet-BC**.

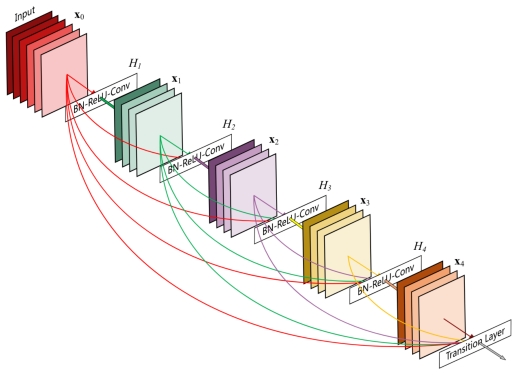


Figure 2.2: Dense block [9]

2.3.3 WideResNet

The main contribution of WideResNet [10] is the idea of making the residual network wider instead of deeper. That works by increasing the number of feature maps per convolution. The following notation describes the architecture choices: WRN-*l*-*k*, where l is the deepening factor, and k the widening factor. When k is equal to one, then the network is a regular ResNet, and $k > 1$ describes a wide ResNet. l represents the number of convolutional layers in the network.

2.4 Out-of-Distribution Detection

The general idea behind and motivation for OOD detection is that images that do not belong to the training distribution are not suited for prediction and should be filtered out.
The first significant work that introduced the term of OOD detection in the area of deep learning was by Hendrycks and Gimpel in 2017 [3]. It introduced a baseline by using softmax and determining a threshold value. Since then, work and research conducted regarding OOD detection can, for the most part, be categorized into three main approaches. Firstly, there is the approach that uses a pre-trained model out of the box without needing to adjust or retrain the model. It leverages the properties of the existing neural network. Then there is the approach that fine-tunes an existing model to detect OOD inputs better. Lastly, there are the methods that need to train a new model from scratch to discriminate between ID and OOD.

2.4.1 Pre-trained model

The baseline approach from Hendrycks and Gimpel falls into the category of pre-trained models. It uses the maximum softmax probability and sets a threshold accordingly. The softmax has been identified as an issue for OOD detection by several authors. ODIN [13] and Generalized ODIN [14] smoothes the softmax activations via temperature scaling $S_i(x;T) = \frac{\exp(f_i(x)/T)}{\sum_{j=1}^{C}\exp(f_j(x)/T)}$, the Energy score [15] uses the logits instead of softmax probabilities to detect outliers. Wang *et al.* [16] extend the Energy score by using all energies instead of a single one.
Other pre-trained approaches use the Mahalanobis distance [17] over hidden space representations or ensemble multiple models [18] to strenghten the prediction and confidence of the network.
The proposed OVAF method does not fall into the category of pre-trained models, as such models are usually trained under a closed-world assumption [5], which makes them inherently not very well suited for OOD detection.

2.4.2 Model fine-tuning

The main approach for model fine-tuning is to take a pre-trained model and extend or adjust the functionality by training it for a few epochs to make them better suited for OOD detection. Outlier Expose [19] fine-tunes existing models to output a uniform distribution in the case of OOD inputs. MixOE [20] extends this by mixing ID and OOD samples, for example, via interpolation. The Energy score [15] also proposes a method that relies on additional outlier data to adjust the pre-trained model's energies. There also exist self-supervised [21] and even unsupervised approaches [22].
The proposed OVAF method can be categorized as a fine-tuning procedure. The advantages are that one can leverage strong existing models while still changing the network structure to the desired needs.

2.4.3 Model trained from scratch

A few methods also rely on training an entire neural network from scratch. VOS [23] trains a regular classifier as well as an OOD detector at the same time, where the OOD detector receives both ID and generated OOD data. The IsoMax [24] is an in-place substitution for the softmax, which incorporates a notion of distance into the model and performs on par for image classification while improving the OOD detection.
OVAF does not need a complete model to be trained from scratch, as this is very time intensive and does not provide significant performance boosts.

2.5 One-vs-All Classifiers

The main scheme for OVA Classifiers is to train C different binary classifiers that distinguish the examples from a single class to all remaining classes. They were popularized due to the possibility to extend Support Vector Machines [25] from the binary to multi-class setting [26], [27]. But they were quickly overtaken and replaced by more sophisticated methods, despite its simplicity and potential [27]. OVA Classifiers can still perform on par or even better than the now-standard softmax-based classifiers [5], but are not very scalable. Suppose the training dataset consists of 100 different classes, then an OVA approach would need to train 100 individual classifiers, whereas a traditional process with softmax would only require one.

2.6 Combining One-vs-All Classifiers and Out-of-Distribution Detection

Due to the rather low popularity of OVA Classifiers, there is little research on the intersection of OVA and OOD. Most approaches combine OVA classifiers with a softmax based classifier. OVANet [28] uses softmax predictions to find the closest class and OVA Classifiers to verify the class or detect an OOD input. Franchi *et al.* combine the probabilities of the sigmoid and softmax output to get more reliable results [29].

3 Related Work

This section introduces related work in detail. The methods presented here will be compared with OVAF in future experiments.

3.1 MSP - Maximum Softmax Probability

A very popular baseline for OOD detection is the Maximum Softmax Probability (MSP) proposed by Hendrycks and Gimpel in 2017 [3]. The usual softmax classification procedure is selecting the class with the highest probability for each image. In the case of OOD detection, the above procedure gets extended by determining a threshold δ that decides whether an input image is OOD. Intuitively, this means that if the softmax confidences are too low, the image is regarded as OOD. Mathematically, the detector can be formulated as

$$g(x) = \begin{cases} \text{ID} & \text{if } \max_i(S_i(f(x))) > \delta \\ \text{OOD} & \text{if } \max_i(S_i(f(x))) \leq \delta \end{cases}$$

where $S_i(f(x))$ is the softmax function applied element-wise to the logits of the neural network $f(x)$ (see Equation 2.1).

3.2 Mahalanobis

In 2018, Lee *et al.* [17] have proposed a distance-aware method that uses the Mahalanobis distance. The Mahalanobis distance allows to calculate the distance between a vector and a distribution. In the proposed method, the vector is a feature space representation of the input image, and the distribution is a pre-computed distribution of each class in the feature space. Recall that $h(x)$ describes a feature space representation of the input image x. For every class, one needs to compute the empirical mean and covariance

based on all the training samples $(x_i, y_i) \in \mathcal{D}_{\text{in}}^{\text{train}}$ in the following way:

$$\hat{\mu}_c = \frac{1}{N_c} \sum_{i:y_i=c} h(x_i)$$
$$\hat{\mathbf{\Sigma}} = \frac{1}{N} \sum_{c} \sum_{i:y_i=c} (h(x_i) - \hat{\mu}_i)(h(x_i) - \hat{\mu}_i)^\top$$

where N_c denotes the number of training samples for class c. The closest class-conditional gaussian distribution to an input vector is calculated by

$$M(x) = \min_c \left((h(x) - \hat{\mu}_c)^\top \hat{\Sigma}^{-1} (h(x) - \hat{\mu}_c) \right)$$

At inference, the method calculates the distance to the closest class-conditional gaussian distribution and, as before, sets a threshold based on it:

$$g(x) = \begin{cases} \text{ID} & \text{if } M(x) <= \delta \\ \text{OOD} & \text{if } M(x) > \delta \end{cases} \tag{3.1}$$

The authors propose the two following extensions to improve the OOD detection performance further. Firstly, one can use input pre-processing to make ID and OOD samples more separable:

$$\hat{x} = x + \epsilon \times \text{sign}(\nabla_x M(x)) = x - \epsilon \times \text{sign}(\nabla_x (h(x) - \hat{\mu}_{\hat{c}})^\top \hat{\Sigma}^{-1} (h(x) - \hat{\mu}_{\hat{c}})) \tag{3.2}$$

The magnitude of the noise is given by ϵ, and $\hat{c}$ denotes the closest class in feature space. Secondly, Lee *et al.* propose to use feature ensembling over different layers in the neural network. They combine the distances via a weighted average over the layers and train a logistic regression detector using the validation samples.

3.3 Energy Score

Another approach is based on energy-based models. Instead of using the softmax probabilities, W. Liu *et al.* propose to use and fine-tune energies to find a good threshold value for OOD input [15]. They show the connection between energies and neural classifiers and define the energy as $E(x) = -f(x)$. For OOD detection, energies are used to determine a threshold:

$$g(x) = \begin{cases} \text{ID} & \text{if } \max_i(-E_i(x)) > \delta \\ \text{OOD} & \text{if } \max_i(-E_i(x)) \leq \delta \end{cases} \tag{3.3}$$

Note that the authors use the negative Energy score, which is equivalent to the logits of the neural network. In addition to the inference-time detection, W. Liu *et al.* [15] propose a method to fine-tune the energies on other OOD data to train a good energy gap between ID and OOD intentionally. The training objective for the fine-tuning is

$$\begin{aligned} L_{\text{energy}} = & E_{(x_{\text{in}},y)\sim\mathcal{D}_{\text{in}}^{\text{train}}}(\max(0, E(x_{\text{in}}) - m_{\text{in}}))^2 \\ & + E_{(x_{\text{out}})\sim\mathcal{D}_{\text{out}}^{\text{train}}}(\max(0, m_{\text{out}} - E(x_{\text{out}})))^2 \end{aligned}$$

The values m_{in} and m_{out} are hyperparameters used to penalize when an ID input generates too high energies, and an OOD input generates too low ones. When the model is fine-tuned, inference OOD detection works just as in Equation 3.3.

3.4 VOS - Virtual Outlier Synthesis

It has been shown that OOD detectors can benefit from exposure to real OOD data during the training [19]. However, OOD data is often not readily available and hard to generate. Du *et al.* [23] try to fix this problem by synthesizing outliers during the training of the classifier. The outliers are generated in the feature space and are fed to a small logistic regression model. The logistic regression model aims to to find a good discriminator between ID and OOD data. To synthesize the outliers, Du *et al.* [23] calculate a class-wise multivariate gaussian distribution with shared covariance as described in 3.1 in an online fashion, where the features used for calculation are constantly updated as the training progresses. Then they sample virtual outliers $\mathcal{V}$ from the ϵ-likelihood region of the distribution via the following equation:

$$\mathcal{V}_c = \{v_c | \frac{1}{(2\pi)^{m/2}|\hat{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(v_c - \hat{\mu}_c)^\top \hat{\Sigma}^{-1}(v_c - \hat{\mu}_c)\right) < \epsilon\},$$

where $v_c \sim \mathcal{N}(\hat{\mu}_c, \hat{\Sigma})$ denotes the generated outliers for class c and ϵ is chosen such that it is close to the class boundaries but still in the low-density region. At inference, they use the output of the logistic regression head to find a good threshold value δ to discriminate between ID and OOD. The logistic regression is based on the energies of the model, which allows formulating $g(x)$ similar to the Energy score from above (see 3.3).

3.5 OVNNI

OVNNI [30] is an approach that combines traditional softmax classification and OVA classifiers. Here, the authors train a conventional softmax classifier using cross-entropy loss and C different OVA classifiers for each class. At inference time, they combine the output of both models to receive a better probability estimate for each class

$$p_i^{\text{OVNNI}}(x) = S_i(x) \times P_i^{\text{OVA}}(y_i = 1|x)$$

where $S_i(x)$ describes the softmax probability (see 2.1) for class i and P_i^{OVA} is the prediction of the OVA classifier corresponding to class i. As in many other methods as well, they then try to find a good threshold value δ to discriminate between ID and OOD:

$$g(x) = \begin{cases} \text{ID} & \text{if } \max_i(p_i^{\text{OVNNI}}) > \delta \\ \text{OOD} & \text{if } \max_i(p_i^{\text{OVNNI}}) \leq \delta \end{cases}$$

4 Methodology

This section describes the proposed method in detail. It explains how to train Filtering Heads and how OVAF works at inference time.

4.1 OVAF Architecture

Let $f(x)$ denote a pre-trained neural network classifier and $h(x)$ a forward pass to the penultimate layer representation, such that $f(x) = w \times h(x) + b$. On top of the pre-trained model, there exist different Filtering Heads; one for each ID class. The Filtering Heads receive the output of $h(x)$ and map it to a probability.
There are two main reasons why the Filtering Heads are trained on the hidden space representation of an image instead of the image itself. Firstly, the hidden space representation is lower in dimension and contains much more dense information, making it easier and faster to train as the Heads do not require the capacity to extract information from high-dimensional images. Secondly, synthesizing outliers in the hidden space is more traceable due to a lower dimensionality.

4.2 Training the Filtering Heads

For simplicity, the following will only describe the training process for a single Filtering Head responsible for a single class $1 \leq c \leq C$. The procedure applies to all of the C Filtering Heads. The main goal of the Filtering Head for class c is to verify whether an input belongs to the class c or not. A Filtering Head maps from a feature space representation of an image to a single scalar output, which denotes the probability of the representation belonging to the class c. Formally, a Filtering Head is a function $b_c(x) : \mathbb{R}^H \rightarrow \mathbb{R}$ where H denotes the size of the feature space. By default, the Head consists of two fully-connected layers with dropout [31] and ReLU [8] activation functions. The Head needs positive and negative samples to learn a reasonable decision boundary between class c and the rest. Figure 4.1 visualizes the training process for the Filtering Head of class red.

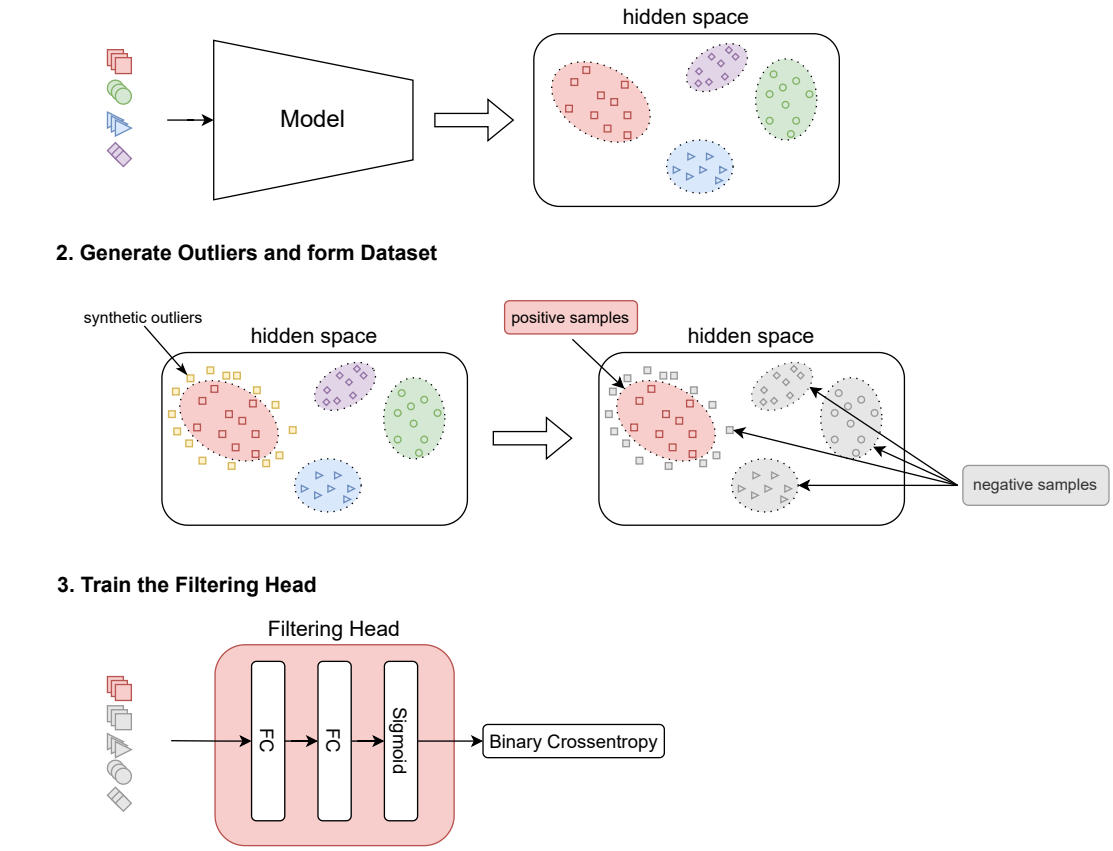


Figure 4.1: Procedure of training a Filtering Head to detect the red squares. The first step is to obtain a hidden space representation of the inputs; one can use both ID and supplemental OOD data. The second step is synthesizing outliers from the hidden space representation of the class red and converting all features into a labeled dataset. Lastly, a small Filtering Head is trained on the binary decision: Red square or not?

Positive samples

The positive samples for the c-th Filtering Head are all images from the training dataset labeled with class c. Formally,

$$\mathcal{D}^c_{\text{pos}} = \{(h(x), 1) \,|(x, y) \in \mathcal{D}_{\text{in}}; y = c\}$$

where $h(x)$ is the hidden space representation of the input image x based on the pre-trained model.

Negative samples

There are three different sources from which the negative samples can come from. The goal is to obtain negative samples that are as diverse as possible such that the confidence band for the class is narrow (see the example in Figure 1.2).

Other classes Firstly, any sample from class $j \in C \setminus c$ that is not the positive class can be used as a negative sample. This is the conventional way of doing OVA Classification. Therefore, the negative sample contain

$$\mathcal{D}^c_{\text{other}} = \{(h(x), 0) \,|(x, y) \in \mathcal{D}_{\text{in}}; y \neq c\}$$

Synthetic outliers Furthermore, one can generate synthetic outliers in the feature space using an approach very similar to VOS [23]. The main difference in generating the outliers between VOS and OVAF is that OVAF uses a class-individual covariance matrix Σ_c whereas VOS [23] uses a shared covariance matrix. The synthetic outliers are referred to as $\mathcal{D}^c_{\text{syn}}$. The empirical class mean $\hat{y}_c$ and class covariance $\hat{\Sigma}_c$ can be calculated with:

$$\hat{y}_c = \frac{1}{N_c} \sum_{i:y_i=c} h(x_i)$$

$$\hat{\Sigma}_c = \frac{1}{N_c} \sum_{i:y_i=c} (h(x_i) - \hat{y}_c)(h(x_i) - \hat{y}_c)^\top$$

where N_c describes the number of samples for class c.
Those are used to sample synthetic outliers s_c from the low-density region of the distribution.

$$\{s_c | \frac{1}{(2\pi^{m/2})|\hat{\Sigma}_c|^{1/2}} \exp\left(-\frac{1}{2}(v_c - \hat{\mu}_c)^\top \hat{\Sigma}_c^{-1}(v_c - \hat{\mu}_c)\right) < \epsilon\},$$

Real outliers Lastly, if available, one can obtain the hidden representation of real OOD data. The real outliers must be forwarded, and the hidden representation must be obtained:

$$\mathcal{D}^c_{\text{real}} = \{(h(x), 0) \,|x \in \mathcal{D}_{\text{out}}\}$$

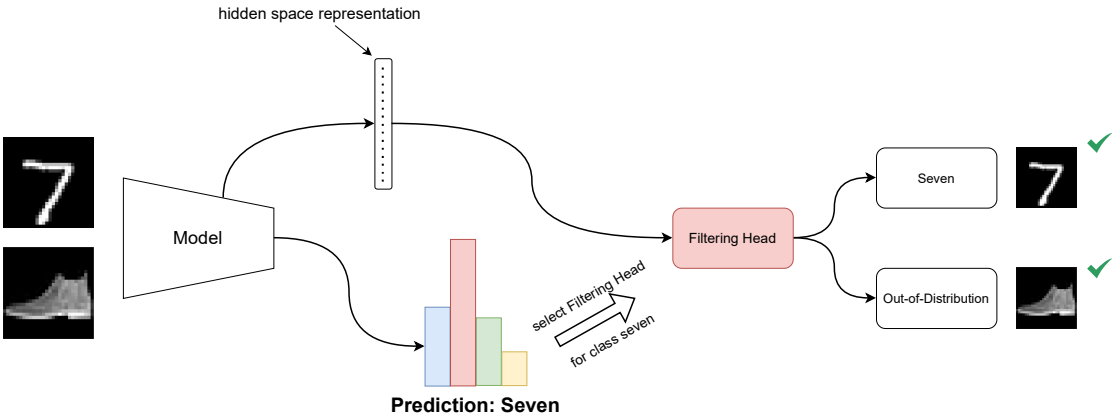


Figure 4.2: OVAF procedure at inference time. The task is to classify written digits. The model first makes a class prediction. During the forward pass, it stores a hidden space representation of the input. Based on the class prediction, the corresponding Filtering Head is selected and receives the stored hidden space representation. The Filtering Head decides whether the prediction was correct (7) or is OOD (shoe).

It does not matter whether $\mathcal{D}_{\text{out}}$ is labeled or unlabeled, as the label is overwritten with a 0 regardless. The negative part of the dataset $\mathcal{D}^c_{\text{neg}}$ will comprise samples from $\mathcal{D}^c_{\text{other}}$, $\mathcal{D}^c_{\text{syn}}$ and $\mathcal{D}^c_{\text{real}}$. The overall training dataset for Filtering Head c then consists of

$$\mathcal{D}^c \subseteq \{\mathcal{D}^c_{\text{pos}} \cup \mathcal{D}^c_{\text{neg}}\}$$

Training Objective

After all, the Filtering Head $b_c(x)$ trains on a combination of the above data. The training objective for a Filtering Head is binary cross-entropy. The loss for the n-th sample of Filtering Head c is written as

$$\mathcal{L}^c_n = [y_n * \log(b_c(x_n)) + (1 - y_n) * log(1 - b_c(x_n)]$$

where $(x_n, y_n) \in \mathcal{D}^c$ denote the hidden space representation and corresponding label, respectively.

4.3 Inference and OOD Detection

At inference time, the proposed method works as follows. First, the pre-trained model predicts a class for the input image. Then the hidden space representation of that input

is forwarded through the Head corresponding to the class that the pre-trained model predicted. The input image is an outlier if the probability is below a certain threshold.

$$g(x) = \begin{cases} \text{ID} & \text{if } b_c(h(x)) \leq \delta \\ \text{OOD} & \text{if } b_c(h(x)) > \delta \end{cases}$$

where c denotes the class that the original pre-trained model predicted for the input. Figure 4.2 depicts a visualization of the inference process.

5 Experiments

The experiments section describes the setup for evaluating OVAF and other competitive OOD detection methods. It presents datasets, architectures, implementation details, hyperparameters, and metrics used for the experiments. The first experiment is a comparison of OVAF to other state-of-the-art methods. The second experiment investigates the importance of synthetic and real outliers and how they affect performance.

5.1 Competitive Comparison

This experiment compares OVAF to other OOD detection methods. Table 5.1 describes which datasets are used in combination with which models and summarizes the experiment.

Table 5.1: Experiment Tasks to evaluate the performance of different OOD detection methods

Model Architecture	$\mathcal{D}_{in}$	$\mathcal{D}_{\text{supp}}$	$\mathcal{D}_{\text{out}}$
MLP	MNIST	FashionMNIST	notMNIST
DenseNet, WideResNet	CIFAR-10	CIFAR-100	SVHN
	CIFAR-100	CIFAR-10	SVHN

5.1.1 Datasets

Two groups of datasets are used for training and evaluating the different methods. Each group consists of a ID dataset on which the models are trained ($\mathcal{D}_{\text{in}}$), a supplemental OOD dataset for fine-tuning ($\mathcal{D}_{\text{supp}}$), and a different OOD dataset for evaluating the performance ($\mathcal{D}_{\text{out}}$). Figure 5.1 shows example images for each dataset.
The first group contains:
MNIST: A grayscale 28×28 dataset that depicts handwritten digits. It contains 60.000 training and 10.000 testing images.
FashionMNIST [32]: A grayscale 28×28 dataset of clothing items from Zalando. There

(a) MNIST (b) FashionMNIST (c) NotMNIST

(d) CIFAR-10 (e) CIFAR-100 (f) SVHN

Figure 5.1: Datasets Sample Images

are ten different categories. It contains 60.000 training images and 10.000 testing ones.
notMNIST: A dataset comprised of over 500.000 28×28 grayscale images of glyphs of letters A - J. Only the hand-cleaned version containing 18.726 images is used.

The second group is the more complex one, with the following datasets:
CIFAR-10 [33]: A colored 32×32 dataset of 10 different classes. There are 50.000 training and 10.000 testing images.
CIFAR-100 [33]: A dataset similar to CIFAR-10, but it has 100 different classes. Images are also RBG and 32×32.
SVHN: A dataset called Street View House Numbers that consist of around 100.000 labeled real-world images taken from Google Street View. Images are 32×32, RBG, and depict the digits 0-9.

5.1.2 Architectures and Training Choices

Table 5.1 summarizes which models are used for which tasks. The first one is a simple MLP. The two others are competitive architectures for computer vision: DenseNet [9] and WideResNet [10].
Besides VOS [23], all OOD methods are based on the same pre-trained neural network classifier with the following training settings:

MLP For the easiest task of MNIST image classification, a simple 3-Layer MLP is trained for ten epochs. The batch size is 256 with an initial learning rate of 10^{-4} which decreases using Cosine Annealing [34]. The optimizer is Adam [35].

DenseNet/WideResNet The exact model architecture is DenseNet-121 [9] and WideResNet-40-2 [10]. The training settings are similar across both CIFAR-10 and CIFAR-100 as well as for the DenseNet and WideResNet Architecture; the only exception being the batch size. The models are trained for 100 epochs with an initial learning rate of 0.01. The optimizer is Adam [35], and Cosine Annealing [34] is used to schedule the learning rate decay. For WideResNet, the batch size is set to 128, and DenseNet trains on a batch size of 64 due to memory constraints.
The hyperparameters for the different OOD detection methods are the following:

Energy score The regular Energy score [15] without fine-tuning does not have any special hyperparameters to tune. The fine-tuned variant trains for 10 epochs and the hyperparameters that penalize too low or high energies are $m_{\text{in}} = -25$ and $m_{\text{out}} = -7$ (see Equation 3.3).

VOS The VOS implementation follows exactly the details as laid out in the paper [23]. After 40% of the training, it starts with generating virtual outliers and training the OOD Discriminator. Every 10.000th sample is chosen to be fed into the logistic regressor. VOS does not rely on pre-trained models and must be trained from scratch. It uses the same model architecture and training details as described above.

Mahalanobis The Mahalanobis method [17] is used with additional input processing (Equation 3.2), but without a feature ensemble over different hidden layers. The noise hyperparameter for input pre-processing is $\epsilon = 10^{-5}$.

OVAF The main hyperparameter of interest for OVAF is what negative inputs each Filtering Head receives. The notation OVAF ($\mathcal{D}_{\text{other}}/\mathcal{D}_{\text{syn}}/\mathcal{D}_{\text{real}}$) is used to describe how the dataset is composed of. For example OVAF (50/50/0) indicates that the negative points are 50% from $\mathcal{D}_{\text{other}}$, 50% from $\mathcal{D}_{\text{syn}}$ and 0% from $\mathcal{D}_{\text{real}}$. If the sum adds up to 100%, then the dataset is balanced such that there exist as many positive as negative samples.
While generating synthetic outliers, every 1.000th sampled feature vector is used as

an OOD sample. The Filtering Head consists of two fully connected layers as well as dropout [31] with $p = 0.3$ and the ReLU [8] activation function.

5.1.3 Evaluation Metrics

		Actual ID input	
		True	False
Predicted ID input	True	true positive (TP)	false positive (FP)
	False	false negative (FN)	true negative (TN)

Table 5.2: Confusion Matrix

The experiments use the following commonly used metrics to assess and compare performance among different methods: the area under the receiver operating curve (**AUROC**), the area under the precision-recall curve (**AUPR**), and the false positive rate at 95% true positive rate (**FPR95**). Table 5.2 is a confusion matrix that is necessary for calculating the mentioned metrics.

AUROC can be interpreted as the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance [36]. In the context of OOD detection, AUROC implies that a randomly chosen ID input has a higher probability of belonging to a class than a randomly chosen OOD input. The ideal classifier has an AUROC of 100%; a random classifier of 50%.

$$\textit{precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{5.1a}$$

$$\textit{recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{5.1b}$$

$$\textit{false positive rate} = 1 - \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} \tag{5.1c}$$

$$\textit{true positive rate} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{5.1d}$$

AUPR is an alternative to AUROC that is especially useful for imbalanced or skewed datasets as it highlights performance differences that are not captured in the AUROC curve [37]. The AUPR calculates the area under a precision-recall plot created by evaluating the precision (Equation 5.1a) and recall (Equation 5.1b) values for specific thresholds. The ideal and a random classifier again have 100% and 50%, respectively.

FPR95 is the classifier's false positives rate (Equation 5.1c) when the threshold is set to have a 95% true positive rate (Equation 5.1d). A perfect OOD detection method

would achieve 0% FPR95. It is the easiest to interpret and also what most of the following analysis is based on.

5.2 Synthetic vs Real Outliers

As discussed in the introduction, several methods use and require an additional OOD dataset during training. OVAF does not need them per se, as it can also generate synthetic outliers. The main question driving this experiment is whether synthetic or real outliers are better to decrease the false positive rate. The experiment is limited to the OVAF method but might still give some insights into the broader question. It also provides insights into the hyperparameters of OVAF and how to tune them. The OVAF method is applied to the detection tasks based on CIFAR-10 and CIFAR-100. The base model will be the already pre-trained WideResNet [10] and OVAF is trained using different hyperparameters regarding the composition of the dataset for the Filtering Heads. The baseline is to train the Filtering Heads only with other classes in the conventional OVA fashion. Then progressively, 25% of the dataset is replaced with synthetic or real outliers until the dataset will consist of 100% synthetic or real outliers. Synthetic and real outliers are never used together in the same dataset.

6 Results

This section presents the results for the two experiments mentioned in the chapter above. The first compares OVAF to other competitive benchmarks; the second explores the differences between synthetic and real outliers. OVAF significantly outperforms the baseline method across all tasks, reducing the FPR95 by up to 47.2% from the baseline. It establishes state-of-the-art performance on CIFAR-10 and CIFAR-100. On MNIST, the Mahalanobis method and fine-tuned Energy score perform the best.

6.1 Competitive Comparison

The OOD Detection algorithms are evaluated on three tasks in the image classification domain with increasing difficulty. The ID accuracy of the base models is presented in Table 6.1. The accuracy consistently drops from MNIST over CIFAR-10 to CIFAR-100, indicating the increasing complexity of the task.

Table 6.1: Model Accuracies

Dataset	**Model**	**Accuracy**
MNIST	MLP	98.24
CIFAR-10	DenseNet	93.64
	WideResNet	94.45
CIFAR-100	DenseNet	76.21
	WideResNet	75.32

Table 6.2: Comparison of OOD Detection Methods. Results are shown in percentages. Methods are evaluated on 10.000 MNIST and 18724 NotMNIST (OOD) samples. Numbers in **bold** are superior. Methods denoted with * had access to additional OOD data (FashionMNIST) at training.

$\mathcal{D}_{\text{in}}$	**Method**	**AUROC**	**AUPR**	**FPR95**
MNIST (3-Layer MLP)	MSP [3]	86.9	75.4	53.0
	Energy [15]	66.2	43.2	75.1
	OVNNI [30]	93.3	87.2	28.3
	Mahalanobis [17]	99.4	98.8	2.2
	VOS [23]	77.0	57.6	63.6
	OVAF (25/75/0)	90.9	77.8	25.5
	Energy (fine-tune)* [15]	**99.7**	**99.2**	**1.1**
	OVAF (50/0/50)*	92.6	84.2	26.5

6.1.1 MNIST

The first task is the image classification of MNIST images. The additional OOD dataset for fine-tuning is FashionMNIST, and the metrics are calculated using NotMNIST as the OOD dataset. Table 6.2 contains the detailed results. The Mahalanobis distance [17] and the fine-tuned Energy score [15] outperform all other methods by a large margin. They achieve a FPR95 of 2.2% and 1.1%, respectively. Both OVAF variants also significantly outperform the baseline, reducing the FPR95 from 53% to 25.5% and 26.5%. OVNNI, the second method that uses OVA classifiers, performs slightly worse than OVAF and achieves an FPR95 of 28.3%.

Table 6.3: Comparison of OOD detection methods using DenseNet [9] and WideResNet [10]. Results are shown in percentages. Methods are evaluated using 10.000 CIFAR-10 and 26.032 SVHN (OOD) images. Values in **bold** are superior. Methods denoted with * were fine-tuned with additional OOD data (CIFAR-100). VOS on DenseNet did not train properly and therefore no values are displayed.

$\mathcal{D}_{\text{in}}$	**Method**	**AUROC**	**AUPR**	**FPR95**
CIFAR-10 (DenseNet)	MSP [3]	89.3	83.7	63.7
	Energy [15]	87.1	76.8	61.1
	Mahalanobis [17]	95.1	85.3	23.1
	VOS [23]	-	-	-
	OVAF (50/50/0)	87.6	78.4	34.7
	Energy (fine-tune)* [15]	96.6	**95.5**	21.8
	OVAF (50/0/50)*	**96.9**	94.4	**18.3**
CIFAR-10 (WideResNet)	MSP [3]	91.3	87.3	58.0
	Energy [15]	92.7	85.9	36.3
	Mahalanobis [17]	98.0	95.4	9.5
	VOS [23]	96.4	93.8	18.9
	OVAF (25/75/0)	94.1	89.0	31.7
	Energy (fine-tune)* [15]	**99.2**	**98.6**	**2.4**
	OVAF (0/0/100)*	95.2	92.4	30.0

6.1.2 CIFAR-10

Moving on from MNIST to the more challenging task of CIFAR-10 image classification. The job is evaluated over two model architectures.
For DenseNet [9], OVAF tuned with real outliers outperforms all other methods on AUROC and FPR95, reducing the FPR95 by 45.4% compared to the MSP baseline [3]. The fine-tuned Energy score [15] performs best regarding the AUPR metric. VOS was not evaluated on DenseNet. It did not train the entire 100 epochs because the model returned NaN values; an issue that can, for example, be caused by exploding gradients. On WideResNet, the baseline achieved an FPR95 of 58.0%. Both OVAF approaches asignificantly outperform the baseline, reaching 31.7% and 30.0% FPR95. Fine-tuned Energy score outperforms all other methods by a large margin, achieving an FPR95 of 2.4%. VOS achieves a competitive result of 18.9% FPR95.

Table 6.4: Comparison of OOD detection methods using DenseNet [9] and WideResNet [10]. Results are shown in percentages. Methods are evaluated using 10.000 CIFAR-100 and 26.032 SVHN (OOD) images. Values in **bold** are superior and methods denoted with * were fine-tuned with additional OOD data (CIFAR-10).

$\mathcal{D}_{\text{in}}$	**Method**	**AUROC**	**AUPR**	**FPR95**
CIFAR-100 (DenseNet)	MSP [3]	83.3	75.9	74.6
	Energy [15]	86.6	81.5	76.6
	Mahalanobis [17]	79.0	61.3	70.9
	VOS [23]	72.2	55.5	84.5
	OVAF (50/50/0)	**90.2**	**91.8**	**47.2**
	Energy (fine-tune)* [15]	74.5	71.9	98.6
	Ours (50/0/50)*	87.6	90.6	56.8
CIFAR-100 (WideResNet)	MSP [3]	74.1	64.1	83.5
	Energy [15]	82.7	74.3	79.5
	Mahalanobis [17]	87.2	78.0	58.5
	VOS [23]	76.6	63.2	82.0
	OVAF (25/75/0)	**93.5**	**93.9**	**36.3**
	Energy (fine-tune)* [15]	68.5	60.4	96.5
	OVAF (75/0/25)*	92.9	93.4	40.8

6.1.3 CIFAR-100

On the most challenging task based on CIFAR-100 classification, OVAF outperforms all other methods on both DenseNet and WideResNet. Using DenseNet, the baseline achieves an FPR95 of 74.6%. The regular OVAF without real outliers reduces the baseline by 27.4%. Also, OVAF with real outliers performs 56.8% on FPR95. All other methods are not able to reduce the FPR95 below 70.9%, which is the result of Mahalanobis.
On WideResNet, OVAF can reduce the FPR95 to 36.3%, which is 47.2% lower than the baseline method for OOD detection.
One should note that the fine-tuned Energy score achieves results worse than a random classifier, which would be 95% FPR95. The training procedure is, however, similar to the one on CIFAR-10, where it outperforms all other methods. The original paper [15] proposes some results for the task on CIFAR-100, which are, however, not compatible with the experiments conducted in this thesis.

6.2 Synthetic vs Real Outliers

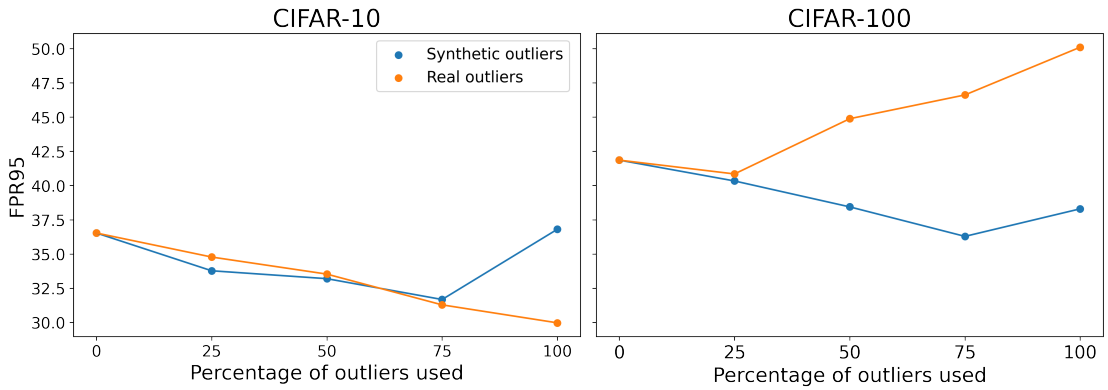


Figure 6.1: Line chart to compare the effects on the performance of using synthetic and real outliers for the Filtering Heads. The baseline only uses other classes as negative inputs. Then part of the other classes is replaced by synthetic outliers (orange) or real outliers (blue). On the left is the experiment for CIFAR-10, and on the right for CIFAR-100. Both use a pre-trained WideResNet. One can see that FPR95 performance gains are more consistent when using synthetic outliers, but overall both can improve the performance.

Results are depicted in Figure 6.1. In the CIFAR-10 experiment, OVAF performs better when using real outliers than synthetic ones. The best result is reached when 100% of the negative inputs in the Filtering Heads are real outliers taken from CIFAR-100. The usage of real outliers reduces the FPR95 from around 36% to 30%, whereas the 36% uses the conventional OVA method by using the other classes. In the case of CIFAR-100, however, the synthetic outliers perform significantly better than real outliers. Usage of synthetic outliers by up to 75% reduces the FPR95 in both experiments, whereas the real outliers are only useful in the CIFAR-10 experiment, and decrease the performance on CIFAR-100 overall.

7 Discussion

This section discusses the findings from the experiments, analyzes the Filtering Heads in more detail, and suggests areas for future work.

7.1 Competitive Comparison

Both variants of the proposed OVAF method outperform the baseline [3] by a large margin, improving the FPR95 by as much as 47.2%. The method is competitive with and outperforms other popular OOD strategies, especially on more complex tasks such as CIFAR-10 or CIFAR-100. However, it does not perform as well as other methods on MNIST.

7.1.1 Scaling in Task Complexity

An interesting observation that one can make from the three experiments is that OVAF scales very well to more complex tasks, especially when compared to other state-of-the-art methods. OVAF is outperformed by techniques such as the Energy score [15] or Mahalanobis distance [17] on the task of MNIST classification. On CIFAR-10, OVAF is already among the best-performing methods for OOD detection, and on CIFAR-100, it significantly outperforms all other methods. Figure 7.1 visualizes the FPR95 performance of all methods on the three tasks via a parallel coordinates diagram. On CIFAR-10 and CIFAR-100, average performance across both model architectures is used. It shows that other methods significantly decline in OOD detection performance when the task gets more challenging. This trend has also been observed by R. Huang and Li [38]. They point out that most methods are only evaluated on simple classification tasks. Observing the performance declines from CIFAR-10 to CIFAR-100 might indicate that current OOD detection methods depend heavily on the underlying model's performance and the task's difficulty. For example, the Mahalanobis distance [17] directly depends on the feature space representation of the underlying model; if the underlying model cannot separate classes well in feature space, the Mahalanobis distance will most likely not be a good discriminator between ID and OOD samples. OVAF, on the other hand, is not affected that much by the representation of the images, as the Filtering Heads can extract additional information out of the hidden space features that are perhaps not

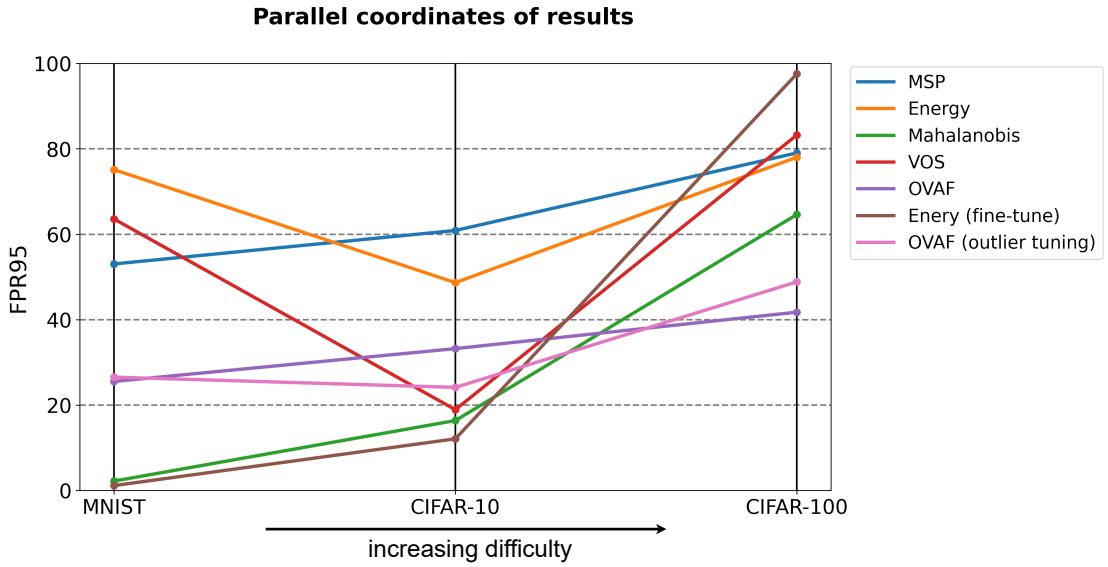


Figure 7.1: Parallel coordinates plot of FPR95 obtained during the benchmark experiment. Performance is measured over all the tasks with increasing difficulty; for CIFAR-10 and CIFAR-100, results are averaged over DenseNet and WideResNet. OVAF (outlier tuning) denotes OVAF results that used supplemental OOD data for training. Most methods decrease in performance when the task becomes more complex, but OVAF performance does not decline as steeply as the others.

considered by the underlying model. This could also be why OVAF performs worse than Mahalanobis or fine-tuned Energy score on MNIST. The features are already an extremely good discriminator between ID and OOD, as the success of the Mahalanobis distance proofs (2.2% FPR95). Then the Filtering Heads cannot extract that much information from the features, and the approach might be overcomplicating the task.

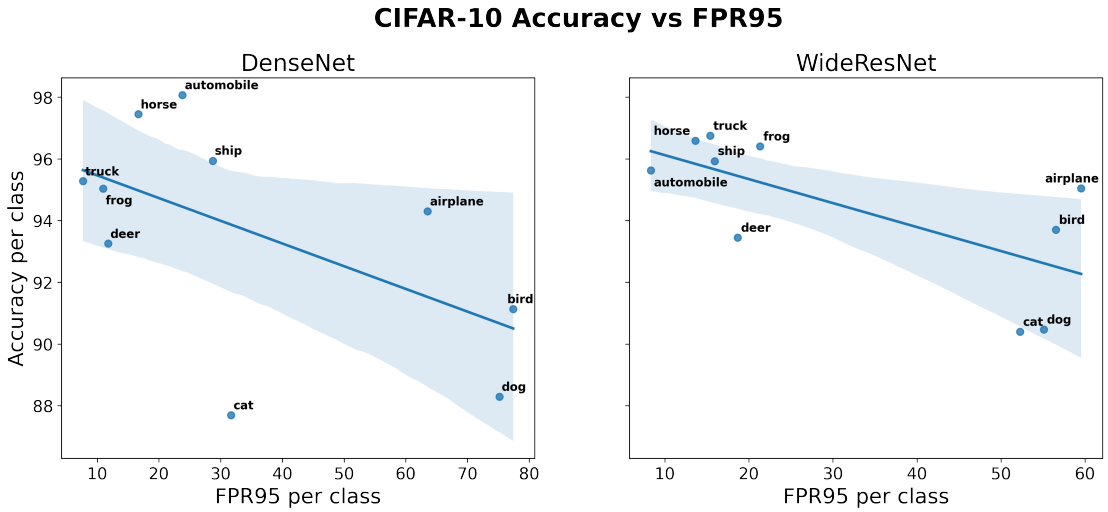


Figure 7.2: Head-wise analysis of accuracy and FPR95 of model and Filtering Heads on CIFAR-10; evaluated on DenseNet (**left**) and WideResNet (**right**). The y-axis denotes the class-wise accuracy, and the x-axis the FPR95 of the corresponding Filtering Head. One can observe a negative correlation for both architectures. The worse the predictive accuracy of the model, the worse the FPR95 of the Filtering Head.

7.1.2 Analysis of Individual Heads

Since OVAF uses class-specific Filtering Heads, it is interesting to analyze the behavior and performance of different Heads. A key observation is that some Filtering Heads perform way better than others on FPR95. The performance of a Filtering Head appears to correlate with the accuracy of the class for the underlying model. The observation is visualized in Figure 7.2. For example, the visualization on the left shows that a prediction of the class horse is way more reliable (~17% FPR95) than the prediction of a dog (~75% FPR95). This is valuable information for the interpretability of the different Heads. One can use the information to quantify how reliable a prediction for a particular class is. It also allows to individually adjust the threshold based on the reliability of a specific head.

7.2 Synthetic vs Real Outliers

The experiment's results suggest that real outliers are not necessarily superior to synthetic ones. Figure 6.1 shows that real outliers can help to improve the FPR95 on CIFAR-10, but on CIFAR-100 it does not work at all and using real outliers makes the performance worse. This might indicate that good OOD data can effectively restrain the confidence region but it is not fully guaranteed. On the other hand, synthetic outliers have the same constant trend; adding more synthetic outliers is beneficial up to some point, e.g., 75%. One of the main advantages of synthetic outliers might be that they are truly randomly sampled and therefore more diverse than real OOD data, which mostly come from a specific distribution. Also the experiment shows that OVAF does not rely on real outliers, which is important as they are often hard or impossible to obtain.

7.3 Future Work

Different lines of further research appear to be promising for OOD detection.

Firstly, as indicated in the introduction, the curse of dimensionality prevents to train a classifier where outliers can narrow the high-confidence region to only focus on the class. In the 2-dimensional space, fitting a circle around the cluster of features is sufficient to train a narrow confidence region (see Figure 1.2). In the 3-dimensional case, a ball could be enough to achieve the same. In a high-dimensional space, however, it is impossible to surround the ID-class's features entirely. Yet this might not be necesseary to create a good OOD discriminator. It could be the case that there are sub-regions in the high-dimensional space in which most of the outliers are located. One could use this potential bias of the model to target the generation of synthetic outliers into that specific sub-region. Another approach could be to reduce the size of the hidden space in which the synthetic outliers are created and therefore get closer to surrounding the ID cluster. For visual understanding and interpretation, one could train an autoencoder, synthesize outliers in the latent space representation and decode them into images again.

Another exciting line of research could be to think about activation and loss functions that do not enforce a closed-world assumption on the training algorithm. This could, for example, work by using OVA classifiers [5]; preferably in an effective fashion that does not require training C classifiers from scratch. Another approach could be to find a replacement for the softmax activation function, which has, for example, already been proposed by Macêdo *et al.* [24].

8 Conclusion

This thesis investigated the problem of OOD detection and approached it via OVA classifiers. The issues of the softmax normalization for OOD detection were identified and characterized, mainly that it causes the closed-world assumption and forces the model to make a decision. The conceptual benefits of OVA classifiers, namely no closed-world assumption and the ability to extract class-specific features, led to the motivation of this thesis: Investigating how OVA classifiers can be used to improve OOD detection. During the thesis, a new technique to detect outliers called OVAF was developed and presented; it trains C different Filtering Heads on top of the hidden space representation of a pre-trained model and uses the Filtering Heads as the component to detect outliers. Additionally, both synthetic and real outliers were used to improve the performance. Extensive experiments showed its superiority over the baseline method. Furthermore, OVAF established state-of-the-art performance on CIFAR-10 and CIFAR-100. It reduced the baseline's false positive rate by up to 47.2% and achieved a false positive rate of 36.3% on CIFAR-100; 22.2% better than the best other approaches. Other experiments demonstrated that OVAF does not rely on real-world OOD samples to train the Filtering Heads, which makes them very flexible and applicable in almost all cases. Finally, future work was suggested to further investigate the underlying mechanisms of OOD detection and how to synthesize outliers effectively.

List of Figures

List of Tables

Bibliography

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems*, vol. 25, Curran Associates, Inc., 2012.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017.

[3] D. Hendrycks and K. Gimpel, "A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, OpenReview.net, 2017.

[4] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On Calibration of Modern Neural Networks," Aug. 3, 2017. arXiv: `1706.04599 [cs]`.

[5] S. Padhy, Z. Nado, J. Ren, J. Liu, J. Snoek, and B. Lakshminarayanan, "Revisiting One-vs-All Classifiers for Predictive Uncertainty and Out-of-Distribution Detection in Neural Networks," Jul. 9, 2020. arXiv: `2007.05134 [cs, stat]`.

[6] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning Transferable Visual Models From Natural Language Supervision," in *Proceedings of the 38th International Conference on Machine Learning*, PMLR, Jul. 1, 2021, pp. 8748–8763.

[7] F. Moller, D. Botache, D. Huseljic, F. Heidecker, M. Bieshaar, and B. Sick, "Out-of-Distribution Detection and Generation Using Soft Brownian Offset Sampling and Autoencoders," presented at the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 46–55.

[8] X. Glorot, A. Bordes, and Y. Bengio, "Deep Sparse Rectifier Neural Networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, JMLR Workshop and Conference Proceedings, Jun. 14, 2011, pp. 315–323.

[9] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708.

[10] S. Zagoruyko and N. Komodakis, *Wide Residual Networks*, Jun. 14, 2017. DOI: 10.48550/arXiv.1605.07146. arXiv: 1605.07146 [cs].

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[12] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *Proceedings of the 32nd International Conference on Machine Learning*, PMLR, Jun. 1, 2015, pp. 448–456.

[13] S. Liang, Y. Li, and R. Srikant, "Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks," presented at the International Conference on Learning Representations, Feb. 10, 2022.

[14] Y.-C. Hsu, Y. Shen, H. Jin, and Z. Kira, "Generalized ODIN: Detecting Out-of-distribution Image without Learning from Out-of-distribution Data," Mar. 31, 2020. arXiv: 2002.11297 [cs, eess].

[15] W. Liu, X. Wang, J. D. Owens, and Y. Li, *Energy-based Out-of-distribution Detection*, Apr. 26, 2021. DOI: 10.48550/arXiv.2010.03759. arXiv: 2010.03759 [cs].

[16] H. Wang, W. Liu, A. Bocchieri, and Y. Li, "Can multi-label classification networks know what they don't know?" In *Advances in Neural Information Processing Systems*, vol. 34, Curran Associates, Inc., 2021, pp. 29 074–29 087.

[17] K. Lee, K. Lee, H. Lee, and J. Shin, "A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks," Oct. 27, 2018. arXiv: 1807.03888 [cs, stat].

[18] S. Fort, H. Hu, and B. Lakshminarayanan, *Deep Ensembles: A Loss Landscape Perspective*, Jun. 24, 2020. DOI: 10.48550/arXiv.1912.02757. arXiv: 1912.02757 [cs, stat].

[19] D. Hendrycks, M. Mazeika, and T. Dietterich, "Deep Anomaly Detection with Outlier Exposure," Jan. 28, 2019. arXiv: 1812.04606 [cs, stat].

[20] J. Zhang, N. Inkawhich, R. Linderman, Y. Chen, and H. Li, *Mixture Outlier Exposure: Towards Out-of-Distribution Detection in Fine-grained Environments*, Mar. 8, 2022. DOI: 10.48550/arXiv.2106.03917. arXiv: 2106.03917 [cs].

[21] S. Mohseni, M. Pitale, J. B. S. Yadawa, and Z. Wang, "Self-Supervised Learning for Generalizable Out-of-Distribution Detection," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, pp. 5216–5223, 04 Apr. 3, 2020, ISSN: 2374-3468. DOI: 10.1609/aaai.v34i04.5966.

[22] Q. Yu and K. Aizawa, "Unsupervised Out-of-Distribution Detection by Maximum Classifier Discrepancy," presented at the Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 9518–9526.

[23] X. Du, Z. Wang, M. Cai, and Y. Li, "VOS: Learning What You Don't Know by Virtual Outlier Synthesis," presented at the International Conference on Learning Representations, May 7, 2022.

[24] D. Macêdo, T. I. Ren, C. Zanchettin, A. L. I. Oliveira, and T. Ludermir, "Entropic Out-of-Distribution Detection," version 13, *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, Jul. 18, 2021. DOI: 10.1109/IJCNN52387.2021.9533899. arXiv: 1908.05569.

[25] B. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers," in *COLT '92*, 1992. DOI: 10.1145/130385.130401.

[26] Y. Liu and Y. Zheng, "One-against-all multi-class SVM classification using reliability measures," in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, vol. 2, Jul. 2005, 849–854 vol. 2. DOI: 10.1109/IJCNN.2005.1555963.

[27] R. Rifkin and A. Klautau, "In Defense of One-Vs-All Classification," *The Journal of Machine Learning Research*, vol. 5, pp. 101–141, Dec. 1, 2004, ISSN: 1532-4435.

[28] K. Saito and K. Saenko, "OVANet: One-vs-All Network for Universal Domain Adaptation," presented at the Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 9000–9009.

[29] G. Franchi, A. Bursuc, E. Aldea, S. Dubuisson, and I. Bloch, "One Versus All for Deep Neural Network for Uncertainty (OVNNI) Quantification," *IEEE Access*, vol. 10, pp. 7300–7312, 2022, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3138978.

[30] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, V. Makarenkov, and S. Nahavandi, "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," *Information Fusion*, vol. 76, pp. 243–297, Dec. 1, 2021, ISSN: 1566-2535. DOI: 10.1016/j.inffus.2021.05.008.

[31] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014, ISSN: 1533-7928.

[32] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[33] A. Krizhevsky *et al.*, "Learning multiple layers of features from tiny images," 2009.

[34] I. Loshchilov and F. Hutter, "SGDR: Stochastic Gradient Descent with Warm Restarts," presented at the International Conference on Learning Representations, Jul. 21, 2022.

[35] D. P. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization*, Jan. 29, 2017. DOI: 10.48550/arXiv.1412.6980. arXiv: 1412.6980 [cs].

[36] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, Jun. 2006, ISSN: 01678655. DOI: 10.1016/j.patrec.2005.10.010.

[37] K. Boyd, K. H. Eng, and C. D. Page, "Area under the Precision-Recall Curve: Point Estimates and Confidence Intervals," in *Machine Learning and Knowledge Discovery in Databases*, H. Blockeel, K. Kersting, S. Nijssen, and F. Železný, Eds., ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2013, pp. 451–466, ISBN: 978-3-642-40994-3. DOI: 10.1007/978-3-642-40994-3_29.

[38] R. Huang and Y. Li, "MOS: Towards Scaling Out-of-Distribution Detection for Large Semantic Space," presented at the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 8710–8719.