Towards Robust Deep Learning
Deep Latent Variable Modeling against Out-of-Distribution and Adversarial Inputs

Glazunov, M.

DOI
10.4233/uuid:beacb059-da33-42ed-a094-8d7674f2ec26

Publication date
2025

Document Version
Final published version

Citation (APA)
Glazunov, M. (2025). *Towards Robust Deep Learning: Deep Latent Variable Modeling against Out-of-Distribution and Adversarial Inputs*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:beacb059-da33-42ed-a094-8d7674f2ec26

Towards Robust Deep Learning
Deep Latent Variable Modeling against Out-of-Distribution and Adversarial Inputs

Mikhail Glazunov

Towards Robust Deep Learning

Deep Latent Variable Modeling against
Out-of-Distribution and Adversarial Inputs

Mikhail Glazunov

Towards Robust Deep Learning

Deep Latent Variable Modeling against
Out-of-Distribution and Adversarial Inputs

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus,
prof. dr. ir. T.H.J.J. van der Hagen,
chair of the Board for Doctorates
to be defended publicly on
Wednesday 5 February 2025 at 15:00 o'clock

by

Mikhail GLAZUNOV

Master of Science in Data Science for Decision Making,
Maastricht University, the Netherlands
born in Kirishi, Russia

Keywords: Deep Learning Robustness, Out-of-Distribution, Adversarial Attacks, Outlier Detection, Latent Representation

Printed by: Proefschrift Printing

Cover by: Dall-E

Copyright © 2025 by M. Glazunov

ISBN 978-94-6510-478-2

An electronic copy of this dissertation is available at
https://repository.tudelft.nl/.

What I cannot create, I do not understand.

Richard Phillips Feynman

CONTENTS

ACRONYMS

AI Artificial Intelligence. 32, 148

BBB Bayes by Backpropagation. 24, 61, 62, 68, 73–78, 98, 126, 127, 144

CIA Confidentiality, Integrity, and Availability. xv, xvii, 24, 31, 47, 143

CNN Convolutional Neural Network. 38, 46, 52, 53, 102, 126

CW Carlini-Wagner. 15, 16, 39, 40, 114, 115, 128, 129

DGLVM Deep Generative Latent Variable Model. 17

DGM Deep Generative Modeling. 49, 50, 54, 61–65, 69, 74, 75, 84, 86, 88–90, 113, 116, 117, 121, 122, 133, 141, 148

DI Dataset Inference. 53

DNN Deep Neural Network. xv, xvii, xviii, 2–4, 15–17, 21, 22, 24, 32–45, 47–52, 54, 61–63, 65, 68, 72, 74, 83, 87–89, 91, 92, 96, 98, 103, 111, 112, 114–117, 121, 123, 125, 127–129, 132, 141–143, 147–149

ELBO Evidence Lower Bound. 19, 20, 22, 65–68, 74, 86, 89, 118–122, 128

ERM Empirical Risk Minimization. 2, 6, 9–11

ESA Equation-Solving Attack. 46

FGSM Fast Gradient Sign Method (FGSM. 39, 40, 49, 114, 115, 128, 129, 133

HMC Hamiltonian Monte Carlo. 66, 68, 123, 124, 132, 145, 146

HVAE Hyperspherical Variational Autoencoder. 91, 96, 97, 101, 104

ID in-distribution. xv, xvii, 13, 14, 62, 70, 72, 74, 76, 117, 123, 128, 133

JSMA Jacobian-based Saliency Map Attack. 40, 114, 115, 128, 129

k-NN k-Nearest Neighbors. 18

MCMC Markov chain Monte Carlo. 24, 66, 68, 117, 118

MLE Maximum Likelihood Estimation. 19, 62, 65, 121, 122

MMA Meta-Model Attack. 46

MSSSIM Multi-Scale Structural Similarity. 124, 130, 131, 145

NTKs Neural Tangent Kernels. 22

OoD Out-of-Distribution. xv, xvii, 2, 3, 12–14, 16, 17, 23–25, 61–65, 69–76, 84, 88–90, 111–114, 116, 117, 121, 124, 126–128, 131–133, 141, 142, 144–149

PAC Probably Approximately Correct. xv, xvii, 7–11, 14, 20, 22, 141

RGB Red, Green, Blue. 72, 126

RLM Regularized Loss Minimization. 10, 11

ROC Receiver operating characteristic. 127

SGD Stochastic Gradient Descent. 22, 66, 75

SGHMC Stochastic Gradient Hamiltonian Monte-Carlo. 61–63, 66, 72–78, 144

SRM Structural Risk Minimization. 10, 11

Stds of LLs Standard Deviation of Log-Likelihoods. 95, 104, 127

SVM Support Vector Machine. 18

SWA Stochastic Weight Averaging. 66

SWAG Stochastic Weighted Averaging-Gaussian. 24, 61, 62, 66, 68, 72, 74, 75, 144

UAT Universal Approximation Theorem. 21, 32, 84, 88–90, 99

ULLN Uniform Law Of Large Numbers. 9

UMI Unique Model Identifier. 53, 54

VADE Variational Deep Embedding. 126

VAE Variational Autoencoder. xv, xvii, 19, 20, 22–25, 50, 51, 54, 62–68, 71–77, 83–86, 88–90, 92, 93, 95–99, 101, 103–105, 111, 113, 114, 116, 117, 120–130, 132, 133, 141, 144–149

VC Vapnik–Chervonenkis. 21

WAIC Watanabe–Akaike information criterion. 24, 61, 64, 69, 73, 75, 88, 98, 144

SUMMARY

Adversarial examples and Out-of-Distribution (OoD) inputs pose a severe risk for Deep Neural Network (DNN) employed in safety-critical tasks. This thesis aims to alleviate these threats by developing robust models that are capable of detecting and mitigating the impact of such inputs. To achieve this goal, we first identify stability as a crucial ingredient that strongly relates to the generalization of models to in-distribution (ID) data. Based on this insight, we focus on the model sensitivity analysis addressing two core aspects: epistemic uncertainty estimation and identifying an appropriate inductive bias.

This thesis starts with Chapter 1, where we thoroughly substantiate our modeling assumptions with regard to imposed inductive bias and choose Variational Autoencoder (VAE) as our candidate for tackling problematic inputs. Moreover, we lay a rigorous theoretical foundation for learnability and generalization, acknowledging known limitations related to the impossibility of OoD detection learnability in a discriminative setting, which motivates switching to generative modeling.

Consequently, this thesis contextualizes in Chapter 2 the considered problematic inputs and mitigations within a broader spectrum of available DNN attacks and defenses. Specifically, we classify the outliers as a type of malicious input by identifying their appropriate category within the Confidentiality, Integrity, and Availability (CIA) triad.

Chapter 3 defines the first core aspect: uncertainty estimation based on the parameter sensitivity analysis with respect to OoD versus ID inputs. The derived uncertainty scores achieved promising results across several benchmarking datasets, confirming our insight about the significance of the stability.

Subsequently, in Chapter 4, this thesis delves deep into the topological properties of VAE related to both the continuity of encoder and decoder mappings and VAE's latent space compactness. This has a deep motivation in a tighter PAC-Bayes generalization bound for the VAE optimization objective, which is discussed in detail in Chapter 1. The most revealing discovery is the fact that OoDs tend to land on latent holes within the latent representation. Based on this insight, a new score has been devised that allows easy detection of the OoD inputs.

In Chapter 5, we take into consideration adversarial examples. The experiments demonstrate that they gravitate towards the latent holes as OoD. Furthermore, an algorithm has been developed to differentiate between inliers, outliers, and two varieties of adversarial inputs accurately.

The thesis concludes by reviewing the key results and discoveries. It also highlights the primary limitations of the research and suggests areas for future improvements in deep learning reliability. This development is not only technically significant but also of considerable societal relevance, as resilient DNNs form the core of reliable applications in safety-critical areas.

SAMENVATTING

Adversariële voorbeelden en Out-of-Distribution (OoD) inputs vormen een ernstig risico voor diepe neurale netwerken (DNNs) die worden gebruikt bij veiligheid-kritische toepassingen. Dit proefschrift heeft tot doel deze risico's te verminderen door robuuste modellen te ontwikkelen die in staat zijn dergelijke inputs te detecteren en hun impact te beperken. Om dit doel te bereiken, identificeren we eerst stabiliteit als een cruciaal aspect dat direct gerelateerd is aan de generalisatie van modellen naar identiek verdeelde (ID) data. Op basis van dit inzicht richten we ons op de gevoeligheidsanalyse van het model, waarbij twee kernaspecten worden aangepakt: epistemische onzekerheidsschatting en het identificeren van een passende inductieve bias.

Dit proefschrift begint met Hoofdstuk 1, waarin we onze modelaannamen over de opgelegde inductieve bias onderbouwen en Variational Autoencoders (VAEs) kiezen als de kandidaat voor het aanpakken van adversariële en OoD inputs. Bovendien leggen we een theoretische basis voor leerbaarheid en generalisatie van modellen. Daarbij houden we rekening met de bekende beperking die onmogelijkheid aangeeft van leerbaarheid van OoD-detectie in discriminerende taken. Deze beperking motiveert de overstap naar generatief modelleren in dit proefschrift.

Vervolgens contextualiseert dit proefschrift in Hoofdstuk 2 de adversariële en OoD inputs, en beschrijft tegenmaatregelen binnen een breder spectrum van beschikbare DNN-aanvallen en verdedigingen. In het bijzonder classificeren we outliers als een vorm van kwaadwillende input door hen in de juiste categorie binnen de confidentialiteit-integriteit-beschikbaarheid (CIA) triade te classificeren.

Hoofdstuk 3 definieert het eerste kernaspect van dit proefschrift: onzekerheidsschatting gebaseerd op de gevoeligheidsanalyse van parameters onder invloed van OoD versus ID inputs. De afgeleide onzekerheidsscores bereiken veelbelovende resultaten op verschillende benchmarkdatasets, wat ons inzicht over het belang van stabiliteit bevestigt.

Vervolgens gaat dit proefschrift in Hoofdstuk 4 dieper in op de topologische eigenschappen van VAE met betrekking tot zowel de continuïteit van encoder- en decoderafbeelding, als de compactheid van de latente ruimte van VAE. Dit heeft een diepe motivatie voor een scherpere PAC-Bayes generalisatiegrens voor het VAE optimalisatiecriterium, wat in detail wordt besproken in Hoofdstuk 1. De belangrijkste bevinding van dit proefschrift is het feit dat OoDs de neiging hebben om op latente gaten binnen de latente representatie afgebeeld te worden. Op basis van dit inzicht wordt een nieuwe score ontwikkeld die eenvoudige detectie van de OoDs mogelijk maakt.

In Hoofdstuk 5 verbreden we het onderzoek van OoD inputs naar adversariële voorbeelden. Experimenten laten zien dat ze als net als OoD inputs neigen naar een afbeelding op gaten in de latente ruimte van de VAE. Tevens wordt een algoritme ontwikkeld om inliers, outliers en twee soorten adversariële inputs nauwkeurig te onderscheiden.

Het proefschrift wordt afgesloten met een overzicht van de belangrijkste resultaten en conclusies. De voornaamste beperkingen van het onderzoek worden besproken en een aantal verbeterpunten voor de betrouwbaarheid van deep learning wordt voorgesteld. Deze ontwikkelingen beschreven in dit proefschrift zijn niet alleen technisch van belang, maar hebben ook aanzienlijke maatschappelijke relevantie, omdat in toenemende mate DNNs gebruikt worden in veiligheid-kritische toepassingen.

1 INTRODUCTION

Deep learning is applied to a rather diverse set of safety-critical tasks ranging from autonomous car driving to automatically-assisted medical diagnosis. Such pervasiveness inherently accentuates the vulnerabilities of deep learning, revealing a critical challenge: deep learning models are not necessarily robust to specific inputs. It may result in either a malicious exploitation of the model or in an unexpected model behavior. To compound the existing challenge, it concerns both the training and testing stages of the model, resulting in two different types of robustness and separate appropriate approaches to their mitigation.

In this dissertation, we specifically focus on the second type of resilience, i.e., the one that deals with the testing or, from the perspective of probabilistic modeling, the inference stage. The ultimate goal of this work is to significantly enhance this type of model's robustness, which includes its detection and mitigation capabilities.

This venture implies making several pivotal decisions and choices to constrain the branching diversity of the available research and experimental paths since the current state of cutting-edge knowledge accumulated for the last several decades in the domain of both machine and deep learning cannot be feasibly addressed in one study. From one perspective, such constraints would necessarily limit the scope of applicability and, as a result, the scope of potential discoveries. From another perspective, however, choosing these constraints forces a researcher to make a more principled and informed decision while considering which options should be kept and which should be filtered out.

This dissertation boldly endeavors to find an equilibrium for the aforementioned contrastive selection so that the suggested solution is still valid for the majority of the deep learning models with respect to their enhanced robustness.

To achieve this goal, we start by exploring a deeper understanding of the problem of learning, learnability, and generalization in this introduction. This exploration helps categorizing the problem statement within this universal approach. In addition, it provides a key insight into the essential elements of our suggested solution.

First, we consider the classical results of the generalization of machine learning models and learning theory. We introduce the necessary notation and formalize the learning framework, addressing the important question of learnability in the case of independent and identically distributed (i.i.d.) samples.

Second, we pose the main problem of our research and investigate the learnability of OoD detection. We recapitulate a recently formally proven impossibility result indicating that there is no universally consistent solution for OoD detection in the case of classification tasks where learnability is characterized by a uniform convergence using Empirical Risk Minimization (ERM) [1].

Third, considering these impossibility results in the case of a supervised discriminative setting, we focus on the unsupervised generative setting instead. Furthermore, rather than ensuring uniform convergence of ERM within the hypothesis class, we employ an alternative approach to learnability by focusing on the variance of the learning rule. This approach leads to the notion of stability of a particular machine learning algorithm. Such a shift in viewpoint is unsurprising since stability has long been identified as the reliable tool to overcome overfitting. It allows for the measurement of the sensitivity of the inferred results with respect to slight variations in the input. We formulate a dual problem of measuring the sensitivity of the pre-trained model to the slight variation in the parameters given a particular fixed input. Hence, we change the perspective from learnability to stability of the model.

Based on the guiding principle of stability, we focus on two core aspects:

1. an epistemic uncertainty estimator

2. the choice of an appropriate inductive bias for ensuring stability

Given the inductive nature of learning, the first aspect addresses the problem of uncertainty estimation of a model. It allows us to infer the model's confidence about its output, indicating a potential direction for tackling the problematic inputs and shedding light on the link between the uncertainty of the model and its stability. The second aspect includes a diverse set of choices related to one's modeling assumptions that allow for improving the underlying stability of the model in question.

Following our discussion of stability as a crucial aspect of model robustness, we introduce the main object of our study in the next section: inputs to DNNs that pose challenges to their robustness.

1.1. PROBLEMATIC INPUTS FOR DEEP NEURAL NETWORKS

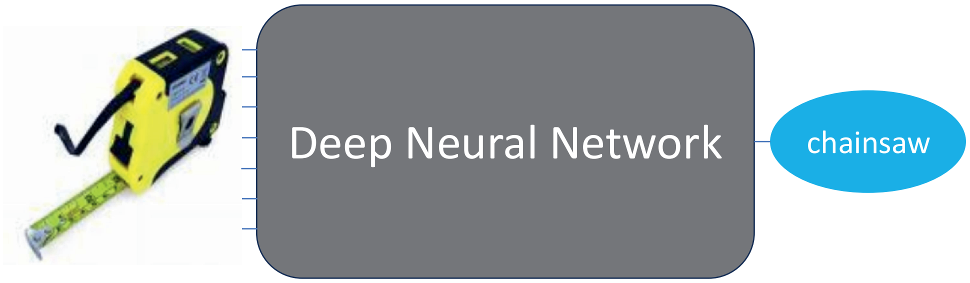
Consider the following scenario: a DNN classifier is trained on a particular image dataset (e.g., Imagenet [2]). The dataset contains images categorized into several categories (e.g., a popular version of Imagenet consists of over a million training images categorized into a thousand different classes). The trained model demonstrates high accuracy values on a test set, i.e., on the images that have never been used during training but are independently sampled from the identical distribution as the images used for training. Although an initial dataset can vary and contain various categories that span a vast diversity of potential objects there are still classes that are not included in the dataset. For example a category of a measuring tape is not present among the available classes in Imagenet. If we use an image of a measuring tape as an input then it would classify it into the category of a chainsaw (see Figure 1.1a). Remarkably, it would do it with a high value of a corresponding probability indicating that the model is rather confident in the resulting output.

This behavior is persistent not only for the specifically chosen category of a measuring tape but for an arbitrary class that is not included into the limited set of the training classes. The phenomenon of assigning too high posterior probabilities is called overconfidence of the model. It may be a significant problem if one is confronted with a necessity of deployment of such a model in the wild since there is almost a certain guarantee that there will be inputs under categories outside of the training set. This could confuse and mislead the interpretation of the model's outputs. These inputs are commonly called outliers or Out-of-Distribution (OoD) inputs. Please note that we will use these terms interchangeably and consider them as synonyms. For a precise definition of the term we refer to Chapter 2.

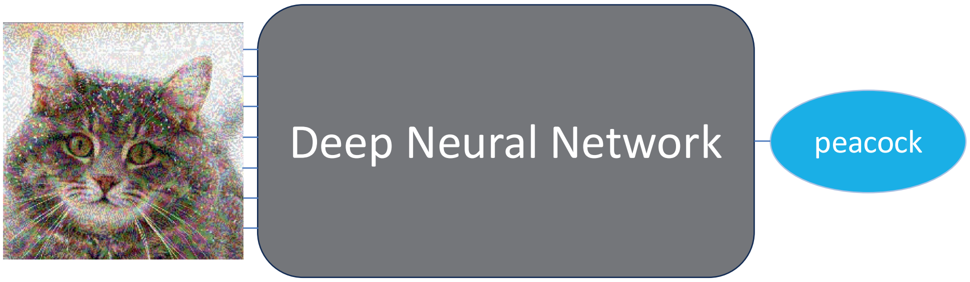
Another example of inputs that are problematic for DNNs constitute the so-called adversarial examples. These are specially forged inputs by an adversarial player who intentionally aims to mislead the classifier. These attacks may be quite difficult to notice since the input quite often resembles a particular benign category from the limited set of allowed classes. However, the DNN classifies it in a totally different category which poses a significant risk for the robustness of the model (see Figure 1.1b). Adversarial examples, attacks and inputs are used interchangeably as synonyms in this dissertation for naming this type of problematic inputs.

1.2. FORMALIZING MACHINE LEARNING

Before improving DNNs' robustness, it is necessary first to understand and formalize the general task of machine learning and its components.



(a) Measuring tape as an input image



(b) Adversarial attack on a cat image

Figure 1.1: (*a*) DNN trained on ImageNet classifies an outlier with a probability $\geq 90\%$ as a chainsaw; (*b*) Untargeted adversarial attack on a cat image that is classified by the same DNN as a peacock again with a high probability $\geq 90\%$.

1.2.1. LEARNING FRAMEWORK AND NOTATION

There are several constituents of the machine learning framework:

- An input metric space $(\mathcal{X}, d_{\mathcal{X}})$, where $\mathcal{X}$ represents the domain of inputs and $d_{\mathcal{X}}$ is a corresponding metric. Let X be a random variable taking values in $\mathcal{X}$. This random variable represents an input to our model. X is distributed according to some unknown probability distribution $\mathbb{P}_X(\mathbf{x})$, we denote a particular realization of X as $\mathbf{x}$. Furthemore, where it is clear from the context, we use $\mathbb{P}(\mathbf{x})$ to denote distribution of X to avoid clutter in notation.

- An output metric space $(\mathcal{Y}, d_{\mathcal{Y}})$ with a metric $d_{\mathcal{Y}}$. In this dissertation we consider only cases when $\mathcal{Y} \subseteq \mathbb{R}^d$, e.g., for a binary classification task: $\mathcal{Y} = \{+1, -1\}$ or $\mathcal{Y} = \{0, 1\}$, in the case of multiclass classification $\mathcal{Y} = [D]$, in the case of regression $\mathcal{Y} = \mathbb{R}^d$. We denote Y being a random variable taking values in $\mathcal{Y}$, and we use $\mathbb{P}(\mathbf{y})$ to denote its distribution.

- A training dataset $S^m = ((\mathbf{x}_1, \mathbf{y}_1) \ldots (\mathbf{x}_m, \mathbf{y}_m))$ sampled from an unknown underlying joint distribution over $\mathcal{X} \times \mathcal{Y}$ such that each $(\mathbf{x}_i, \mathbf{y}_i) \sim$

$\mathbb{P}_{XY}(\mathbf{x}, \mathbf{y})$. This joint distribution consists of two parts: the aforementioned probability over inputs $\mathbb{P}(\mathbf{x})$ and a conditional probability over outputs for each domain point $\mathbb{P}(\mathbf{y}|\mathbf{x})$, i.e., $\mathbb{P}_{XY}(\mathbf{x}, \mathbf{y}) = \mathbb{P}(\mathbf{y}|\mathbf{x})\mathbb{P}(\mathbf{x})$.

- An optimal predictor $f^*(\mathbf{x})$ that achieves the least irreducible error possible for the predicted output given the joint distribution $\mathbb{P}_{XY}$.
- A hypothesis space $\mathcal{H}$, which is a set of functions that map input space to output space $h : \mathcal{X} \to \mathcal{Y}$.
- A learning algorithm A returns a hypothesis (or a set of hypotheses) based on a training dataset $A(S^m)$ in a deterministic or probabilistic manner. It attempts to allocate a single $\hat{h}(\mathbf{x})$ (or a set of hypotheses) that functionally substitutes $f^*(\mathbf{x})$.

Note that this formulation is provided for a supervised machine learning, in the case of unsupervised generative machine learning, it is sufficient to set $\mathbf{y} := \mathbf{x}$ and instead of sampling from $\mathbb{P}_{XY}$ to sample directly from $\mathbb{P}_X$. To unify notation in this introduction whenever possible we will use $\mathbb{P}_D$ and $\mathbf{z}_i$ instead of either $(\mathbf{x}_i, \mathbf{y}_i)$ or $(\mathbf{x}_i, \mathbf{x}_i)$ to cover both supervised discriminative, where $D := XY$, and unsupervised generative cases, where $D := X$.

The task of learning in such case is formulated as the ability of the model to generalize to the unseen datapoints. The general assumption is that the unseen data is sampled from the same distribution $\mathbb{P}_D$ as the one available in the training set and that they are all independent and identically distributed.

1.2.2. RISK, EMPIRICAL RISK AND LEARNING OBJECTIVE

In order to learn, we need first to define a cost of an error event or an error value returned by a model in question. It is natural to represent this cost by a non-negative real value, i.e., for an appropriate hypothesis space $\mathcal{H}$ and a domain $\mathcal{Z}$, let ℓ be $\ell : \mathcal{H} \times \mathcal{Z} \to \mathbb{R}^+$. Such functions are called *loss functions*.

The overall measure of success of a machine learning model, i.e., how good a particular model is at predicting the resulting output for both seen and unseen i.i.d. data can be formulated utilizing a risk function:

$$R(h) := \mathbb{E}_{\mathbf{z} \sim \mathbb{P}_D} \ell(h, \mathbf{z}). \tag{1.1}$$

It is the expected value with respect to the data distribution so they represent the true error of the model, which is called the *generalization error*. And that is exactly the error that one aims to minimize during training. However, the problem is that it cannot be computed due to the fact that the data distribution is unknown. The only available data is a sample S^m based on which one can estimate the so-called empirical risk:

$$\hat{R}(h) := \frac{1}{m} \sum_{\mathbf{z} \in S^m} \ell(h, \mathbf{z}). \tag{1.2}$$

Hence, the final learning objective will be[1]:

$$h^* = \arg\min_{h \in \mathcal{H}} \hat{R}(h). \tag{1.3}$$

This objective and the appropriate learning paradigm is called Empirical Risk Minimization (ERM). It encapsulates many machine learning algorithms and covers all of the algorithms we are implementing in this thesis.

As a result, machine learning can be summarized as follows: it is an optimization for a hypothesis $h \in \mathcal{H}$ that should be as close as possible to the optimal predictor $f^*(\mathbf{x})$. This is achieved via estimation of the risk function $R(h)$ based on the empirical risk function $\hat{R}(h)$. Empirical risk is evaluated on a random sample S^m from the unknown data distribution $\mathbb{P}_D$ where each datapoint is considered as a particular realization of a random variable that is independent and identically distributed. The questions arise when such an approach to learning is feasible, what are the guarantees for a particular model to learn on a particular dataset, and what do these guarantees depend on?

1.2.3. A LEARNABLE PROBLEM

The principal aim is to select a hypothesis $h \in \mathcal{H}$ that achieves the lowest possible risk based on a limited set of observations. Unless training data is randomly labeled, it is reasonable to assume that the risk decreases with increasing sample size. Learning algorithms that identify such hypotheses are called *consistent*. To elaborate, a learning algorithm A is consistent at a monotonically decreasing rate $\varepsilon_{\text{cons}}(m)$ under a particular distribution $\mathbb{P}_D$ if for every sample size m and a bounded loss it is true that:

$$\mathbb{E}_{S^m \sim \mathbb{P}_D^m} \left[R(A(S^m)) - \min_{h \in \mathcal{H}} R(h) \right] \leq \varepsilon_{\text{cons}}(m).$$

Since we do not know the data distribution, the problem is feasible to learn if it holds true for every possible distribution, which is summarized in the following definition [3].

[1]Note that this objective applies only when either hypothesis space is finite or when the minimum is attainable within the set (e.g. if the set is closed). Otherwise, this objective is not feasible, and we aim at infimum, which is not a part of the hypothesis space. In practice, the corresponding loss function is minimized until convergence, obtaining the resulting hypothesis as close as possible to the infimum.

Definition 1.2.1 (Learnable Problem, Shalev). A problem is *learnable* if there exists a consistent rule A and a monotonically decreasing sequence $\varepsilon_{\text{cons}}(m)$ such that

$$\varepsilon_{\text{cons}}(m) \xrightarrow{m\to\infty} 0 \text{ and } \forall \mathbb{P}_D \quad \mathbb{E}_{S^m \sim \mathbb{P}_D^m}\left[R(A(S^m)) - \min_{h\in\mathcal{H}} R(h)\right] \leq \varepsilon_{\text{cons}}(m).$$

This single reasonable requirement for a learning procedure implies that we can control the quality of our approximation in advance. To understand this, consider the following intuitive reformulation. Given that $[R(A(S^m)) - \min_{h\in\mathcal{H}} R(h)] \geq 0$, we can apply Markov's inequality to obtain:

$$\mathbb{P}\left(R(A(S^m)) - \min_{h\in\mathcal{H}} R(h) < \varepsilon\right) > 1 - \frac{\mathbb{E}_{S^m\sim\mathbb{P}_D^m}[R(A(S^m)) - \min_{h\in\mathcal{H}} R(h)]}{\varepsilon} \geq 1 - \frac{\varepsilon_{\text{cons}}(m)}{\varepsilon}.$$

Given that $\varepsilon_{\text{cons}}(m)$ decreases monotonically, it is possible to identify the smallest integer m' satisfying $\varepsilon_{\text{cons}}(m') \geq \delta\varepsilon$ and $\varepsilon_{\text{cons}}(m'-1) < \delta\varepsilon$, where $\delta \in (0,1)$. Hence, for every $\varepsilon > 0$ and $\delta \in (0,1)$, a function $m'(\varepsilon,\delta)$ can be defined which represents a sample complexity. As a result, when the size of a sample for our training dataset has size $m > m'(\varepsilon,\delta)$, one can guarantee with the probability at least $1-\delta$ that a hypothesis obtained by a consistent learner will be within an arbitrary close neighborhood of the true risk:

$$R(A(S^m)) - \min_{h\in\mathcal{H}} R(h) < \varepsilon.$$

Such a reformulation provably allows obtaining an arbitrarily close approximation of optimal risk by setting an appropriate value for ε with an arbitrary level of confidence by choosing an appropriate significance level δ which in turn gives us a sample complexity m' to learn a particular problem. This approach constitutes the essence of Probably Approximately Correct (PAC) learnability or PAC learning framework. To summarize the idea, we provide the following definition [4].

Definition 1.2.2 (PAC learnability, Shai[2]). Given a hypothesis class $\mathcal{H}$, a set $\mathcal{Z}$, and a loss function $\ell : \mathcal{H}\times\mathcal{Z} \to \mathbb{R}^+$, we say that $\mathcal{H}$ is *PAC learnable* if there exists a function $m_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ and a learning algorithm A such that for all $\varepsilon, \delta \in (0,1)$ and for every distribution $\mathbb{P}_D$, if A is run on $m \geq m_{\mathcal{H}}(\varepsilon,\delta)$ i.i.d. samples from $\mathbb{P}_D$, then it produces an $h \in \mathcal{H}$ satisfying:

$$\mathbb{P}\left(R(A(S^m)) - \min_{h\in\mathcal{H}} R(h) < \varepsilon\right) \geq 1-\delta.$$

where the probability is taken over the choice of the m training examples from $\mathbb{P}_D$.

Based on that, we can identify two types of errors related to PAC learnability. The first error relates to the quality of our hypothesis space and the lowest possible risk attainable by a predictor in this space $\min_{h\in\mathcal{H}} R(h)$. This error is incurred due to the restriction to a particular class. It is called the *approximation error*. The second error is the so-called *estimation error*, manifested as the difference between the risk of an optimal hypothesis $R(A(S^m)$ obtained by a learning algorithm and the risk of a true minimum hypothesis $R(h)$ within a hypothesis class.

Note that the PAC learnability definition does not say anything about how to achieve this learnability, i.e., how to choose an appropriate learner (e.g., how to ensure that it is consistent). Moreover, it says nothing about the choice of an appropriate hypothesis class or a loss function.

1.2.4. GENERALIZATION VIA INDUCTIVE BIAS

Can we devise a universal learner to attain PAC learnability on any data? The No-Free-Lunch theorem (and its alternatives) proves that it is impossible [4–6]. Specifically, in a scenario where no further information is provided beyond a finite training sample S^m, it is logical to assume that every potential predictor $f(\mathbf{x})$ consistent with S^m is plausible so it is chosen uniformly, i.e., the distribution over the set of consistent predictors $\mathbb{P}(f)$ is uniform. Hence, the No-Free-Lunch theorem claims, in general, that there is no distinction in the performance between any two learning algorithms, denoted as $A_1(S^m)$ and $A_2(S^m)$, averaged across all the potential predictors:

$$\mathbb{E}_{\mathbb{P}(f)}\left[R(A_1(S^m)\right] = \mathbb{E}_{\mathbb{P}(f)}\left[R(A_2(S^m)\right].$$

To avoid the uniformity assumption over hypotheses we have to assume some prior knowledge. In other words, we need an inductive bias to allow our learning algorithm to generalize [7]. It also means that PAC learnability alone is only necessary but not sufficient for generalization. The reason lies in the selection of the hypothesis space; if the inductive bias of this space is strong and happens to be incorrectly chosen, then the approximation error will be substantial, irrespective of the sample size. This is because the optimal value $\min_{h\in\mathcal{H}} R(h)$ is also poor. Therefore, even though PAC learnability applies, generalization remains unattainable.

The most common bias is based on Occam's razor, which represents a bias towards parsimony [8].

In practice, inductive bias relates to any modeling assumption we make, so it comprises the following items:

- the choice of the hypothesis space
- the choice of the loss function

- the choice of the learning algorithm

Every particular choice for any of the items stated above will bias our solution, so it will be substantiated further in this chapter.

1.2.5. LEARNING ALGORITHMS, GENERALIZATION BOUNDS AND STABILITY

Recall that PAC-learnability formulation tells us nothing about how to come up with an appropriate learner. It simply allows for the bounding of the true risk to an arbitrary level of precision with a desired level of confidence. Classical statistical learning theory states that ERM is a consistent learner if and only if it converges uniformly (the so-called key theorem of learning [9]). Moreover, based on the Uniform Law Of Large Numbers (ULLN), it is possible to derive generalization bounds for risk function bounded by ERM and a ratio between the complexity of the appropriate hypothesis class and the number of training examples. These bounds vary depending on the underlying task to be solved, but for a binary classification task with a (0-1) loss, the most prominent result states that:

Theorem 1.2.3 (**Vapnik–Chervonenkis Bound**)**.** *Assume that $\mathcal{H}$ is finite or has a finite Vapnik–Chervonenkis (VC) dimension. If the ERM algorithm is trained with m i.i.d. examples sampled from $\mathbb{P}_D$, then with probability $1-\delta$, the risk of its output h^* is bounded by:*

$$R(h^*) \leq \hat{R}(h^*) + \sqrt{\frac{\log|\mathcal{H}| + \log(1/\delta)}{2m}}.$$

where $|\mathcal{H}|$ is the complexity of the hypothesis class, which is either the number of elements in the case of finite $\mathcal{H}$ or VC-dimension in the case of infinite $\mathcal{H}$. Hence, PAC learnability over class $\mathcal{H}$ is achievable with the following sample complexity $m(\varepsilon,\delta) = \frac{1}{2\varepsilon^2}\left(\ln|\mathcal{H}| + \ln\frac{1}{\delta}\right)$.

There are numerous other generalizations of this bound, e.g., in the case of multiclass classification, it is possible to utilize the Natarajan dimension [10], the conditions on loss functions can be relaxed, etc. To summarize, one can control the generalization of the learning up to an arbitrary level of precision utilizing this bound if uniform convergence holds true and the appropriate hypothesis class is used.

When the hypothesis space does not have a uniform convergence property, the likelihood of overfitting becomes much more apparent since there is no more guarantee that an optimal ERM hypothesis will be in the vicinity of true risk [4]. Note that the generalization bound has a tension between an ERM and the complexity of a hypothesis class. This tension is similar to the bias-variance tradeoff in the statistics and is dubbed the bias-complexity tradeoff. The standard learning procedure in such a case is Structural Risk Minimization (SRM), when the hypothesis class is divided into subclasses of increasing or decreasing complexity, each of which satisfies the uniform convergence property. Subsequently, via fitting ERM and computing the bound, one can identify a sweet spot (a minimum) where the bias-complexity tradeoff achieves the optimal balance.

In the case of SRM, we identify some prior discrete ranking of hypothesis classes. There is also a possibility to make a continuous version of this prior by utilizing a PAC-Bayes approach. It is a generalization of the classical PAC learnability which allows to define a prior p on hypotheses that does not depend on the data S^m. Let $q(S^m) = q$ denote a posterior distribution on hypotheses that can depend on the data. Then risk and empirical risk functions can be defined as follows:

$$R_q = \mathbb{E}_{h\sim q}[R(h)] \text{ and } \hat{R}_{S^m,q} = \mathbb{E}_{h\sim q}[\hat{R}(h)|S^m].$$

Based on these assumptions, a PAC-Bayes generalization bound can be formulated [11]:

Theorem 1.2.4 (**PAC-Bayes Bound, McAllester**). *For any loss function $\ell \in \{0,1\}$, arbitrary data distrubtion $\mathbb{P}_D$, any hypothesis class $\mathcal{H}$ and a probability measure p supported on $\mathcal{H}$, for all q probability measures supported on $\mathcal{H}$, it holds that:*

$$R_q \leq \hat{R}_{S^m,q} + \sqrt{\frac{KL[q\|p] + \log\frac{2\sqrt{m}}{\delta}}{2m}}.$$

where $KL(q\|p)$ denotes the Kullback-Leibler divergence between the probability distributions q and p.

Since it is true for all q simultaneously, we can minimize the bound with respect to q to find the appropriate posterior.

An alternative approach to a learning problem is the so-called Regularized Loss Minimization (RLM). It represents a learning rule where one jointly minimizes the empirical risk and a regularization function. The regularization function usually bounds the model parameters aiming at a more predictable and stable behavior. A classic example of such a learner is a Tikhonov regularization [12]. RLM constitutes an instance of the learning approach based on the stability of the learning algorithm. Stability is a desirable condition for generalization. There are various formalizations of stability that prove that it is a necessary condition or, in some cases, even a necessary and sufficient condition for generalization [3, 13, 14].

To summarize, the discussed learning approaches address generalization within the following categories:

1. Approaches based on Vapnik-Chervonenkis theory, including ERM with uniform convergence over the hypothesis space, and SRM.

2. Approaches based on stability, such as RLM.

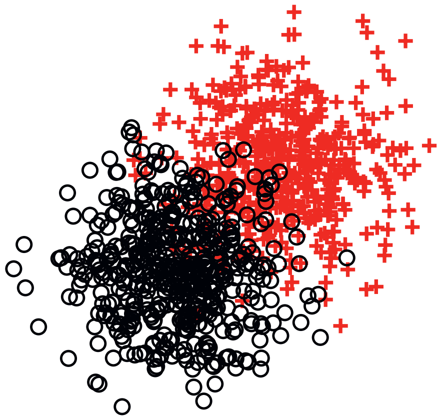
3. Approaches based on information-theoretic bounds, exemplified by PAC-Bayes methods.

These three categories encompass the majority of learning theory approaches to addressing generalization. To further explore these concepts, let us rely for the moment on an intuitive understanding of stability as a mechanism to prevent overfitting, without delving into formal definitions. Consider the uniform convergence requirement first. This can be viewed as the most stringent condition of stability in relation to the true solution. It implies that the learning algorithm consistently stays within a maximum ϵ-neighborhood of the true solution, representing the highest achievable stability. In the context of SRM, we utilize our prior knowledge to rank the complexities of models. This process involves sequentially optimizing for the minimum value of a generalization bound, ultimately selecting the most stable model. The PAC-Bayes framework builds on this idea by extending the ranking of models to continuous cases, thereby ensuring stability through probabilistic bounds. For RLM, stability is explicitly incorporated into the learning process. This is achieved by adding a stability term directly to the objective function, which typically involves bounding the norms of the model parameters. All this evidence strongly indicates that stability is a critical condition for generalization.

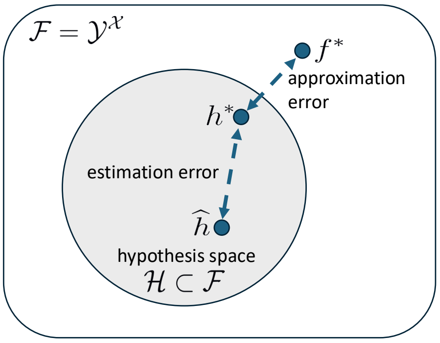
1.3. UNCERTAINTY ESTIMATION

There are two types of uncertainties that are commonly associated with any machine learning task [15]: *aleatoric* and *epistemic*.

The first type is related to the inherent random effects in our data or data collection routines, which are unavoidable. For instance, consider rolling a die. The outcome is inherently unpredictable due to the stochastic nature of the process, and no amount of additional information can predict the exact result, only the probabilities of each face appearing. Another example is when there is an overlap between two classes, making it uncertain for prediction in this region of overlap. The second type is frequently referred to as systematic uncertainty, i.e., the one that is caused by a lack of knowledge about the best model. Since this type of uncertainty reflects the decision maker's ignorance rather than any intrinsic randomness, it comprises both estimation and approximation errors (see Figure 1.2). Thus, while aleatoric uncertainty is due to the fundamental randomness of the process and cannot be reduced with more information, epistemic uncertainty is due to incomplete knowledge and can be decreased as more data and insights are obtained.



(a) Aleatoric uncertainty


$\mathcal{F} = \mathcal{Y}^{\mathcal{X}}$
f^*
approximation error
h^*
estimation error
$\widehat{h}$
hypothesis space
$\mathcal{H} \subset \mathcal{F}$

(b) Epistemic uncertainty

Figure 1.2: (a) Aleatoric uncertainty is inherent to the randomness in data or data collection process; e.g., there is no clear decision boundary. (b) Epistemic uncertainty stems from both estimation and approximation errors where $\widehat{h}$ is a hypothesis obtained by a learning algorithm A, h^* is an optimal hypothesis within the chosen hypothesis space, and f^* is a real optimal predictor.
Adapted from Hüllermeier and Waegeman [15].

In prediction, aleatoric uncertainty is modeled via a posterior distribution over labels $\mathbb{P}(\mathbf{y}|\mathbf{x})$. As for the epistemic uncertainty, a common approach is to compute a posterior distribution over hypotheses $\mathbb{P}(h|\mathbf{x})$ with a subsequent marginalizing over hypothesis space to calculate a model predictive posterior. In the classic example of the classification problem it results in the following estimation:

$$\mathbb{P}(\mathbf{y}|\mathbf{x}) = \int_{\mathcal{H}} \mathbb{P}(\mathbf{y}|\mathbf{x}, h) d\mathbb{P}(h|\mathbf{x}). \tag{14}$$

Since marginalization over the whole hypothesis space is typically infeasible, using a Monte-Carlo averaging over several models is common practice.

1.4. PROBLEM STATEMENT

In this section, we formulate the problem that we aim to solve, namely, to enhance the robustness of deep learning classifiers with respect to OoD and adversarial inputs, enabling their detection and providing the means for their mitigation. For OoD, we base our approach on the formalization posited by Fang et al. [1]. Following this approach, we formalize

OoD detection learnability and state the corresponding impossibility theorem. For adversarial examples, we formalize the essence of adversarial attacks, providing several commonly used examples. Finally, we state our research questions.

1.4.1. OOD DETECTION LEARNABILITY

First, we extend the previously introduced notation for classic inlier learning to encompass the OoD detection learning problem. We start with defining inlier and outlier distributions. Recall that we have denoted an input metric space $(\mathcal{X}, d_X)$ with a metric d_X that represents a set of possible inputs and a corresponding output metric space $(\mathcal{Y}, d_Y)$ which in the case of multiclass classification problem represents a discrete label space $\mathcal{Y} := \{1, \dots, K\}$ where K stands for number of classes. We denote in-distribution random variables as $X_I \in \mathcal{X}$ and $Y_I \in \mathcal{Y}$ with a joint distribution $\mathbb{P}_{X_I Y_I}(\mathbf{x}, \mathbf{y})$ over $\mathcal{X} \times \mathcal{Y}$. Note that K covers only in-distribution classes; hence, for the task of out-of-distribution detection, we introduce an additional class $K+1$ for all OoD inputs. For this class, we introduce an OoD random variable Y_O. The OoD random variable in the input space is correspondingly denoted as X_O and their joint distribution: $\mathbb{P}_{X_O Y_O}(\mathbf{x}, \mathbf{y})$ over $\mathcal{X} \times \{K+1\}$.

We train our model only on inlier joint distribution. However, during the inference stage, we deal with a mixture of inlier and outlier joint distributions, namely: $\mathbb{P}_{XY}(\mathbf{x}, \mathbf{y}) = (1-\pi)\mathbb{P}_{X_I Y_I}(\mathbf{x}, \mathbf{y}) + \pi \mathbb{P}_{X_O Y_O}(\mathbf{x}, \mathbf{y})$. Moreover, we receive as an input only instances coming from the marginal distribution $\mathbb{P}_X(\mathbf{x}) = (1-\pi)\mathbb{P}_{X_I}(\mathbf{x}) + \pi \mathbb{P}_{X_O}(\mathbf{x})$, where $\pi \in (0,1)$ stands for the unknown prior of the outlier distribution versus the inlier distribution.

Definition 1.4.1 (**OoD Detection, Fang et al.**)**.** Assume an in-distribution $\mathbb{P}_{X_I Y_I}(\mathbf{x}, \mathbf{y})$ and a dataset $S^m := \{(\mathbf{x}_1, \mathbf{y}_1), \dots, (\mathbf{x}_n, \mathbf{y}_n)\}$, with each $(\mathbf{x}, \mathbf{y})$ pair sampled independently from $\mathbb{P}_{X_I Y_I}(\mathbf{x}, \mathbf{y})$. The objective for detecting OoD instances is to devise a classifier f using dataset S^m. Given that $\mathbf{x}$ is drawn from the mixed marginal distribution $\mathbb{P}_X(\mathbf{x})$, this classifier should be capable of: 1) accurately categorizing a test instance $\mathbf{x}$ to the appropriate in-distribution class when $\mathbf{x}$ originates from $\mathbb{P}_{X_I}(\mathbf{x})$; and 2) recognizing a test instance $\mathbf{x}$ as an OoD input when $\mathbf{x}$ derives from $\mathbb{P}_{X_O}(\mathbf{x})$.

Let $\mathscr{D}_{XY}$ denote a domain space that comprises a collection of joint distributions $\mathbb{P}_{XY}(\mathbf{x}, \mathbf{y})$, each formed by a mixture of ID joint distributions $\mathbb{P}_{X_I Y_I}(\mathbf{x}, \mathbf{y})$ with a OoD joint distributions $\mathbb{P}_{X_O Y_O}(\mathbf{x}, \mathbf{y})$. In this context, the mixed joint distribution $\mathbb{P}_{XY}(\mathbf{x}, \mathbf{y})$, which encompasses both ID and OoD elements, is defined as the domain. We denote $\mathscr{D}_{XY}^{\text{all}}$ as a domain space that consists of all domains.

Let $\mathcal{Y}_{all} = \mathcal{Y} \cup \{K+1\}$ to account for the additional OoD class.

Given a loss function $\ell : \mathcal{Y}_{all} \times \mathcal{Y}_{all} \to \mathbb{R}_0^+$ satisfying $\ell(\mathbf{y}_1, \mathbf{y}_2) = 0$ if and only if $\mathbf{y}_1 = \mathbf{y}_2$, and any $h \in \mathcal{H}$, then the risk with respect to $\mathbb{P}_{XY}$ is

$$R(h) := \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathbb{P}_{XY}}[\ell(h(\mathbf{x}),\mathbf{y})]. \tag{1.4}$$

The separate risks for IDs $R^{in}(h)$ and OoDs $R^{out}(h)$ are defined as:

$$R^{in}(h) := \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathbb{P}_{X_I Y_I}}[\ell(h(\mathbf{x}),\mathbf{y})], \quad R^{out}(h) := \mathbb{E}_{\mathbf{x}\sim\mathbb{P}_{\mathcal{X}_O}}[\ell(h(\mathbf{x}),K+1)]. \tag{1.5}$$

Definition 1.4.2 (**Learnability of OoD Detection, Fang et al.**)**.** OoD detection is learnable in $\mathscr{D}_{XY}$ for $\mathcal{H} \subseteq \{h : \mathcal{X} \to \mathcal{Y}_{all}\}$, if there exists an algorithm $A : S^m \to \mathcal{H}$ and a monotonically decreasing sequence $\varepsilon_{cons}(m)$, such that $\varepsilon_{cons}(m) \to 0$, as $m \to +\infty$, and for any domain $\mathbb{P}_{XY} \in \mathscr{D}_{XY}$,

$$\mathbb{E}_{S\sim\mathbb{P}^m_{X_I Y_I}}\left[R(A(S^m)) - \min_{h\in\mathcal{H}} R(h)\right] \leq \varepsilon_{cons}(m), \tag{1.6}$$

Such an algorithm A is consistent with respect to $\mathscr{D}_{XY}$.

Note that when $\pi = 0$ then Definition 1.4.2 is reduced to the PAC-learnability. Hence, it is an extension of PAC-learnability 1.2.1 to the OoD scenario. While supervised agnostic PAC-learnability is distribution-free, meaning the domain space $\mathscr{D}_{XY}$ includes all possible domains, the learnability of OoD detection is inherently not distribution-free due to the absence of OoDs during training.

Definition 1.4.3 (**Overlap Between ID and OoD**)**.** There is an overlap between in-distribution $\mathbb{P}_{X_I}$ and out-of-distribution $\mathbb{P}_{X_O}$ if there exists at least one event or subset $A \subseteq X$ such that:

$$\exists A \subseteq X : (\mathbb{P}_{X_I}(A) > 0) \wedge (\mathbb{P}_{X_O}(A) > 0)$$

In other words, there exists at least one subset of the feature space where both $\mathbb{P}_{\mathcal{X}_\mathcal{I}}$ and $\mathbb{P}_{\mathcal{X}_\mathcal{O}}$ assign a non-zero measure, indicating the presence of an overlap between the two distributions.

Finally, we are ready to state the impossibility theorem for OoD detection learnability:

Theorem 1.4.4 (**Impossibility Theorem, Fang et al.**)**.** *Let $\mathcal{H}$ represent a hypothesis space and $\mathscr{D}_{XY}$ denote a prior-unknown space of distributions. Suppose there exists a distribution $\mathbb{P}_{XY}$ within $\mathscr{D}_{XY}$ where overlaps occur between ID and OoD. If the minimum in-distribution error $\min_{h\in\mathcal{H}} R^{in}(h)$ and the minimum out-of-distribution error $\min_{h\in\mathcal{H}} R^{out}(h)$ are both zero, then it is impossible to learn OoD detection for $\mathcal{H}$ in the space $\mathscr{D}_{XY}$.*

Since inliers and outliers occupy the same support in all of the examples that we will consider, this result guarantees that OoD detection cannot be learnt in such cases. Moreover, since learnability is a necessary condition for generalization, it implies that there is no general solution for the problem of OoD detection.

1.4.2. ADVERSARIAL EXAMPLES

Adversarial examples are specially crafted inputs (typically images) that cause DNNs to misclassify them, even though to humans, these inputs appear almost identical to the original images. It should be noted, however, that from the attacker's perspective, it is not always a concern to make these modifications imperceptible to humans. For them, the focus is on ensuring the success of their attacks without necessarily adhering to the constraint of making changes unnoticeable. We provide three primary techniques for creating such adversarial examples starting from normalized inputs $\mathbf{x} \in [0,1]^n$:

1. **Fast Gradient Sign Method (FGSM):** This approach exploits the gradient-based learning of DNNs to generate adversarial examples. It aims to maximize the loss by adjusting the pixels of the input image through the sign of the gradient of the loss function concerning the input. The equation details the method:

$$\mathbf{x}' = \mathbf{x} + \varepsilon \cdot \mathrm{sgn}(\nabla_{\mathbf{x}} \ell(h_{\boldsymbol{\theta}}(\mathbf{x}), y_s)), \mathbf{x}' \in [0,1]^n$$

where $\nabla_{\mathbf{x}}\ell$ is the gradient of the loss function, with respect to the original input $\mathbf{x}$, y_s is the true or source label for $\mathbf{x}$, $\boldsymbol{\theta}$ stands for the model parameters that are constant, and ε is a small step size. The method also allows for targeted attacks by altering the gradient direction and the label.

2. **Carlini-Wagner (CW) Attack:** Focused on minimizing the added noise for effective misclassification, the CW attack formulates an optimization problem to find the smallest possible perturbation that leads to misclassification into a specified target class. The optimization objective encapsulates this approach:

$$\text{minimize } ||\delta||_p \text{ subj. to } h_{\boldsymbol{\theta}}(\mathbf{x} + \delta) = y_t, \quad \mathbf{x} + \delta \in [0,1]^n$$

Here, δ represents the noise added to the original input $\mathbf{x}$, aiming to achieve minimal perturbation, and y_t is the intended target class label of a resulting attack.

3. **Jacobian-based Saliency Map Attack (JSMA):** This method uses the DNN's Jacobian matrix to identify and modify the most influential features on the classification output. By altering features with high saliency values, the attack iteratively pushes the input towards misclassification, adhering to a perturbation limit. The attack focuses on making significant, yet limited, changes to the input to deceive the DNN.

These techniques illustrate different strategies to create adversarial examples, highlighting the balance between effectiveness and perceptibility and the diverse approaches attackers can employ to challenge DNN image classifiers.

1.4.3. RESEARCH QUESTIONS

Building upon the described and formulated problems, we proceed to pose the following research questions (RQs):

RQ1: How can we categorize problematic inputs in the context of attacking and defensive methods in deep learning?

RQ2: Is detecting problematic inputs utilizing epistemic uncertainty estimation possible?

RQ3: How problematic inputs are interpreted from the perspective of their internal latent representation?

1.5. OUR APPROACH ADDRESSING THE PROBLEM

We observe the fact that adversarial examples that aim at optimality, such as a CW attack that produces minimum perturbation for misclassification, may differ in a very small subset of pixels with the original image. The fact that even so small perturbations can still mislead prediction results is indicative that these examples rely on some sort of instability in the behavior of DNNs. Furthermore, considering the persistent phenomenon of DNNs' overconfidence with respect to both OoD and adversarial inputs, we hypothesize that they may be considered similarly in the DNNs internal representation. Taking these insights into account, we elaborate our method based on sensitivity analysis of a trained model.

First, we motivate our approach by focusing on stability with respect to parameters, rather than the training data discussed previously. This approach introduces a novel perspective in our research, shifting the focus from data to parameters. To that end, we reiterate the definition of algorithmic stability based on mutual stability between the inferred hypotheses and the training sample which is measured utilizing an overlap coefficient [16].

Definition 1.5.1 (**Overlap Coefficient**). The overlap coefficient between two probability distributions $\mathbb{P}$ and $\mathbb{Q}$ on a measurable space $(\mathcal{X}, \mathcal{F})$ with a σ-algebra $\mathcal{F}$ is defined as:

$$\langle \mathbb{P}, \mathbb{Q} \rangle = \int_{\mathcal{X}} \min\left(\frac{d\mathbb{P}}{d\mu}, \frac{d\mathbb{Q}}{d\mu}\right) d\mu,$$

where μ is a reference measure and $\frac{d\mathbb{P}}{d\mu}$ and $\frac{d\mathbb{Q}}{d\mu}$ are the Radon-Nikodym derivatives of $\mathbb{P}$ and $\mathbb{Q}$ with respect to μ. And both $\mathbb{P} \ll \mu$ and $\mathbb{Q} \ll \mu$.

Definition 1.5.2 (**Mutual Stability**). Mutual stability between two random variables $\mathbf{x} \in \mathcal{X}$ and $\mathbf{y} \in \mathcal{Y}$ is defined as:

$$\mathcal{S}(\mathbf{x}; \mathbf{y}) := \langle \mathbb{P}(\mathbf{x})\mathbb{P}(\mathbf{y}), \mathbb{P}(\mathbf{x}, \mathbf{y}) \rangle$$

where $\langle \mathbb{P}, \mathbb{Q} \rangle$ stands for overlap coefficient between the corresponding probability distributions $\mathbb{P}$ and $\mathbb{Q}$.

It measures the stability of the distribution of one random variable, e.g., X before and after observing an instance of another random variable, e.g., Y. The lower the value of $\mathcal{S}$, the less mutual stability there is between these random variables.

Definition 1.5.3 (**Algorithmic Stability, Alabdulmohsin**)**.** Stability of a learning algorithm A trained on a finite dataset S^m drawn i.i.d from an unknown data distribution $\mathbb{P}(\mathbf{z})$ is defined as $\mathbb{S}(A) = \inf_{\mathbb{P}(\mathbf{z})} \mathcal{S}(A(S^m), S^m)$ where the infimum is taken over all possible distributions of observations $\mathbb{P}(\mathbf{z})$. A learning algorithm is called *algorithmically stable* if $\lim_{m\to\infty} \mathbb{S}(A) = 1$.

Second, we observe that mutual stability is symmetric with respect to its input random variables, namely:

$$\langle \mathbb{P}(\mathbf{x})\mathbb{P}(\mathbf{y}), \mathbb{P}(\mathbf{x}, \mathbf{y}) \rangle = \mathbb{E}_{\mathcal{X}} \langle \mathbb{P}(\mathbf{y}), \mathbb{P}(\mathbf{y}|\mathbf{x}) \rangle = \mathbb{E}_{\mathcal{Y}} \langle \mathbb{P}(\mathbf{x}), \mathbb{P}(\mathbf{x}|\mathbf{y}) \rangle$$

Consequently, we can adopt a dual perspective on stability for the algorithm during the inference stage when the training is finalized. Since stability is necessary for learning, we assume that a trained model is stable with respect to the input data and has good generalization potential. Hence, provided that input data is fixed, we can, in theory, measure the algorithmic stability with respect to the distribution over the hypotheses, i.e., taking infimum over h: $\inf_{\mathbb{P}(h)} \mathcal{S}(h, S^m)$ instead of $\mathbf{z}$. To that end, we devise several corresponding scores that are feasible to compute in practice, including the so-called hole indicator that demonstrated the most promising results (see Chapter 4 for details).

Furthermore, considering the impossibility of OoD learnability, we aim to minimize generalization errors for inliers so that only relevant features play a role in the subsequent sensitivity tests that detect problematic inputs.

In the next sections, we provide our motivation for choosing an appropriate *inductive bias* to tackle this problem. Since we consider Deep Generative Latent Variable Models (DGLVMs), we distinguish two parts of our solutions: probabilistic modeling and parameterization of the probabilistic models via DNNs.

1.6. PROBABILISTIC VERSUS NON-PROBABILISTIC

The first distinct choice connected to the modeling assumptions within both supervised and unsupervised branches of machine learning relates to the output of the designed model. In the cases when the output represents a parameterized probability density or mass function, we refer to such models as probabilistic models. In all other cases, we speak about non-probabilistic models.

The examples of non-probabilistic machine learning can be coarsely subdivided into logical and metric-based modeling. The first one incorporates the rule-based approach based on if-else comparisons. In the cases when rules can be used for branching, and the branching results can be represented as nodes and leaves, the corresponding model exemplifies a classical decision tree.

In metric-based machine learning, the output result strongly depends on the local or global metric within either instance or feature space. This approach has numerous examples; we will list three of the most prominent ones. First, the k-Nearest Neighbors (k-NN) algorithm relies on the k-closest points with respect to some metric for either classification or regression. Support Vector Machine (SVM) is based on identifying a maximum margin hyperplane between the disjoint classes. Usually, after applying an appropriate kernel trick, this margin is again based on a corresponding metric in use. Third, linear regression aims to minimize the least squared error between predicted and actual values. It should be noted that some metric-based approaches can be converted to probability-based ones simply by assuring non-negativity and normalizing the corresponding results so that the sum or integral over the results is equal to 1. The most prominent example of such a conversion represents a softmax function that takes any vector of real numbers as input and produces probabilities of a categorically distributed random variable.

Moreover, some metric-based methods, such as linear regression, can be derived completely from the probabilistic perspective, making this distinction not totally disjoint, demonstrating an overlap between these modeling approaches.

DISCRIMINATIVE VERSUS GENERATIVE MODELING

There are two approaches to model the distribution of a random variable (r.v.). The first method is directly related to the modeling of the conditional distribution. Allow X be an input r.v. and Y be an output r.v, e.g., in the case of the classification task X can be any input to classify (e.g., images), and Y can be a corresponding label of a particular input. The direct approach of acquiring $\mathbb{P}_{Y|X}(\mathbf{y}|\mathbf{x})$ would constitute a mapping of an X to a Y, which represents a classical instance of *discriminative* modeling. The common case is when X stands for a high-dimensional input, which manifests a non-trivial underlying structure, whereas Y represents a low-dimensional output such as categories or classes of the resulting classification task. Both the training and the quality estimation of this mapping are performed in a supervised manner when the training and test datasets contain labeled data.

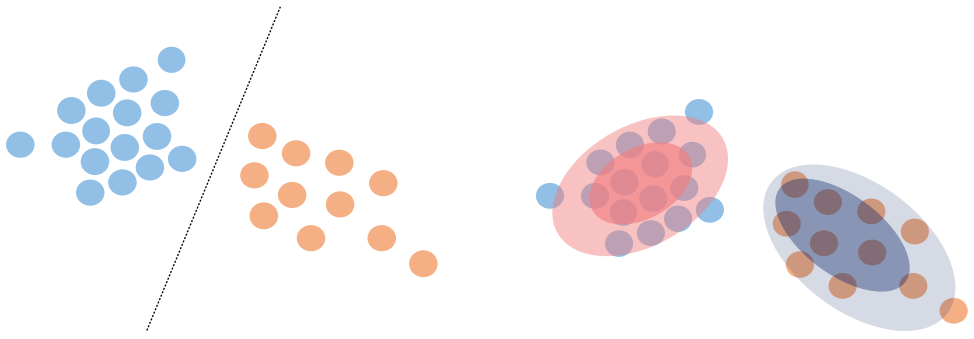


Figure 1.3: (a) Discriminative model learns the decision boundary between the classes $\mathbb{P}(\mathbf{y}|\mathbf{x})$; ($b$) generative model learns the joint distribution of the data $\mathbb{P}(\mathbf{x})$.

The second method poses a more difficult problem to solve: instead of learning a conditional distribution that maps a high-dimensional input to a lower-dimensional output, the problem becomes learning the joint distribution of the data $\mathbb{P}_X(\mathbf{x})$, i.e., modeling the data inputs themselves. This approach allows to generate new unobserved samples of the training data, hence, the name: *generative* modeling.

1.7. PROBABILISTIC MODELING CONSTITUENTS

We choose a probabilistic approach for our model and, in particular, a variant of a generative model, which introduces an additional latent random variable whose posterior is to be estimated via a subclass of Bayesian methods: amortized variational inference. In particular, our probabilistic models are based on Variational Autoencoder (VAE). We explain the rationale behind this decision by examining the major characteristics of the inductive bias inherent to VAE modeling assumptions.

First, the objective function of VAEs is based on a surrogate for Maximum Likelihood Estimation (MLE): Evidence Lower Bound (ELBO) that tends to approximate the marginal likelihood of the input data. ELBO can be expressed as the sum of two terms: the likelihood responsible for reconstruction error and the KL-divergence that regularizes locations of latent posterior within a predefined prior (see Chapter 3 for more details). It means we get a default regularizer on this level for our inductive bias.

Second, VAEs have a bottleneck layer similar to classical autoencoders that puts pressure on information flow, and that requires that the relevant information for reconstruction is compressed. The ability to properly compress your input data distilling the relevant features is an inherent constituent of successful learning.

Third, VAEs provide natural guarantees in terms of PAC-Bayes generalization bound for reconstruction loss [17]. Namely, let $g_\theta(\mathbf{z})$ be a decoder function parameterized by θ that, upon receiving a latent code $\mathbf{z}$ generates a reconstruction of $\mathbf{x}$. The reconstruction loss is defined as:

$$\ell^{\theta}_{\text{rec}}(\mathbf{z},\mathbf{x}) = \|\mathbf{x} - g_\theta(\mathbf{z})\|$$

Let the instance space's diameter $\Delta := \sup_{\mathbf{x},\mathbf{x}' \in \mathcal{X}} d_{\mathcal{X}}(\mathbf{x},\mathbf{x}')$ be finite then the following generalization PAC-Bayes bound can be stated for loss reconstruction:

Theorem 1.7.1 (**PAC-Bayes VAE Bound, Mbacke et al.**). *Let $\Delta < \infty$ be diameter of our input space $\mathcal{X}$, $\mathbb{P}_X$ the data-generating distribution, $\mathcal{Z}$ the latent space, $p(\mathbf{z})$ the prior on the latent space, θ the decoder's parameters, $\delta \in (0,1)$, $\lambda > 0$ be real numbers. With probability at least $1-\delta$ over the random draw of $S^m \sim \mathbb{P}_X$, the following holds for any posterior $q_\phi(\mathbf{z}|\mathbf{x})$:*

$$\underbrace{\mathbb{E}_{\mathbf{x}\sim\mathbb{P}_X}\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\ell^{\theta}_{rec}(\mathbf{z},\mathbf{x})\right]}_{\text{Test Rec. Loss}} \leq \underbrace{\frac{1}{m}\sum_{i=1}^{m}\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)}\left[\ell^{\theta}_{rec}(\mathbf{z},\mathbf{x}_i)\right]}_{\text{Emp. Rec. Loss}} + \underbrace{\frac{1}{\lambda}\left(\sum_{i=1}^{m} KL\left(q_\phi(\mathbf{z}|\mathbf{x}_i)\|p(\mathbf{z})\right)\right)}_{\text{Emp. KL loss}}$$
$$+ \underbrace{\lambda M_\phi M_\theta \Delta}_{\text{Compactness}} + \underbrace{\frac{\lambda^2\Delta^2}{8m}}_{\text{Reg., Comp., Data Size}} + \frac{1}{\lambda}\log\frac{1}{\delta} \tag{1.7}$$

where M_ϕ and M_θ are Lipschitz constant of encoder and decoder mappings correspondingly.

Note that this bound enables optimizing a standard ELBO objective terms of reconstruction and KL-divergence, and it gets tighter as the size of the training set grows. In addition, it contains terms related to the compactness properties of the model which includes both Lipschitz constants used in VAE mappings together with a diameter of the input space and a regularization parameter for balancing a trade-off between the strength of regularization and precision of reconstruction. We will separately cover these points in Section 1.9.

Finally, VAE allows extending Bayesian inference over parameters within the same Bayesian framework as the original ELBO objective function for epistemic uncertainty estimation.

The number of beneficial characteristics makes the choice of VAE as the major candidate for our problem rather straightforward.

1.8. PARAMETERIZING PROBABILITIES WITH DNNS

We use DNNs for learning the parameters of our deep probabilistic model. First, we state the following theorem.

Theorem 1.8.1 (**Universal Approximation Theorem**). *Let $C(\mathcal{X}, \mathcal{Y})$ denote the set of all continuous mappings from $\mathcal{X}$ to $\mathcal{Y}$. Let $\sigma \in C(\mathbb{R}, \mathbb{R})$ represent an element-wise activation function. Then let $\mathcal{N}^{\sigma}_{n,m}$ represent the class of feedforward neural networks with activation function σ, with n neurons in the input layer, m neurons in the output layer, and one hidden layer with an arbitrary number of neurons. Let $K \subseteq \mathbb{R}^n$ be compact. Then σ is nonpolynomial if and only if $\mathcal{N}^{\sigma}_{n,m}$ is dense in $C(K, \mathcal{Y})$.*

It means that an artificial neural network with a single hidden layer is able to approximate any target continuous function up to an arbitrary level of precision.

The first caveat, however, is the number of neurons in the hidden layer, which grows with the approximation power of the network. The current practice is to use DNNs instead, which consist of an arbitrary number of layers. There are available alternatives for the UAT that postulate similar approximation results for DNNs as for neural networks with a single hidden layer. Such an approach also seems to work better in learning the hierarchy of more abstract features of the input data along with deeper layers.

The second caveat relates to the stability concern. Although the approximation capabilities of DNNs lie in the usage of nonpolynomial activation functions, not all nonpolynomials are reasonable to use from the perspective of stability. In principle, according to the UAT one can achieve arbitrary close approximation utilizing exponential functions used as activations. Despite the fact that the solution may be close, such a DNN, even with three hidden layers, would already be quite unstable. As a result, there is a common practice to restrict the space of activation functions to Lipschitz continuous, e.g., sigmoid, tanh, relu are all 1-Lipschitz continuous.

The third caveat pertains to the fact that DNNs are overparameterized, meaning that the number of parameters significantly exceeds the number of training data. If we apply our knowledge from classical learning theory, we will get a very high value for the corresponding VC dimension, which makes most of the classical generalization bounds vacuous. Besides, it means that, in theory, DNNs should overfit achieving very high generalization error. However, the success of DNNs, on the contrary, reveals exceptional generalization capabilities and provides no evidence for overfitting [18].

There are some seminal works that attempt to reconcile this theoretical inconsistency. First, the theory of double descent [19] merges the classical under-parameterized and over-parameterized regimes by experimentally showing that there is a certain level of complexity when the risk value starts decreasing, and models with very high capacity do not overfit, demonstrating good generalization capabilities. This behavior is observed across many over-parameterized machine learning models, including DNNs. Second, the most prominent advance in understanding deep learning theory is made via utilizing Neural Tangent Kernels (NTKs) that connect DNNs in the infinite-width limit to the kernel methods making it possible to linearize a DNN around its initialization with a subsequent study of its training dynamics [20]. Based on this insight, numerous discoveries shed light on the generalization capabilities of DNNs. The most notable ones include the fact that DNNs learn in accordance with the Occam's razor principle, namely: they fit successively higher spectral modes of the target function as the size of training data monotonically increases [21].

To summarize, we may choose a high-capacity DNN functions and fit it to any target function and data with very good prospects for generalization. This combination of inductive biases is already providing a good start for any machine learning task, which explains the recent success of ubiquitous DNN applications.

Finally, we use a variant of the Stochastic Gradient Descent (SGD) algorithm for optimization with the reparametrization trick [22].

As a result, our inductive bias for this layer comprises DNNs that can learn any continuous function to an arbitrary level of precision with a sufficient network depth, fitting the solutions from the simplest to the most complicated as the size of the training dataset grows.

1.9. LINKING DNNS AND PROBABILISTIC MODELING

Despite the numerous beneficial modeling assumptions that both VAEs and DNNs possess, there is still room for improvement. Namely, note that according to the PAC-Bayes generalization bound on reconstruction loss 1.7.1, the term that is related to compactness is not automatically optimized when a standard ELBO optimization objective is used. Moreover, the bound depends on the diameter of the input Δ, which can also be reduced by scaling the input instances. To that end, we delve deeper into compactness and related continuity properties of both VAE's mappings, input, and resulting latent space. In particular, we enforce various Lipschitz constants on both the encoder and decoder of a VAE. Our Lipschitz constraint on DNN mapping results in both additional regularization of the model and allows for controllable compactness of the latent space. We devise a method for identifying appropriate values for these constants, taking into account the intricate peculiarities of high-dimensional representations. Note that this layer is related to both learning and inference procedures. Hence, it also constrains the degree of variation of the outputs, smoothing out the continuity of the mapping and adding to the overall stability of the model.

1.10. CONTRIBUTIONS AND THESIS OUTLINE

In this dissertation, we present the following contributions:

1. We identified that measuring the VAE stability during inference through its parameter sensitivity is a robust and effective method for detecting problematic inputs.
2. We demonstrated that both adversarial and OoD inputs tend to gravitate to holes in their latent representation.
3. We identified and addressed a fundamental theoretical flaw in VAE modeling assumptions by imposing controlled compactness on the latent space using two distinct methods, demonstrating their effectiveness in enhancing the detection of OoDs and adversarial examples. Controlled compactness achieves the following:
   a) It reduces the room available for latent holes, confining them to the compact space used by the inliers.
   b) It ensures that any input to the model, including problematic inputs, is mapped within the same compact space as inliers, increasing the likelihood of detecting these problematic inputs when they land on a hole.
4. We developed a method to distinguish between inliers, outliers, and adversarial examples.
5. Finally, we demonstrated that contrary to the established opinion, Bayesian inference over model parameters is not necessary for sensitivity analysis in VAEs.

These contributions collectively advance the understanding and capabilities of VAEs, particularly in their application to detecting and mitigating problematic inputs. The following chapters will thoroughly cover all contributions, providing theoretical background, methodological approaches, and empirical evaluations.

1.10.1. THESIS OUTLINE

This thesis is composed of multiple chapters that, with the exception of the introduction and conclusion, correspond to the corresponding individual publications. Hence, every chapter represents a separate self-contained article and can be read separately. Moreover, it means that an astute reader could detect some repetition in material among these chapters, given the requirement to introduce the problem in each individual publication. The beginning of each chapter offers details about its corresponding publication. The structural outline of the dissertation is provided below.

CHAPTER 2. VULNERABILITIES AND DEFENSES OF DEEP NEURAL NETWORKS

This chapter answers **RQ1**. In the first part of this chapter, we contextualize problematic inputs within a wider spectrum of deep learning vulnerabilities, including attacks on the training stage and potential privacy-related data leaks. In particular, we frame this range of attacks within the Confidentiality, Integrity, and Availability (CIA) triad. We specifically indicate the place of outliers within these categories by identifying possible uses of outliers in corresponding adversarial scenarios. Moreover, we consider two potential definitions of adversarial examples, which, despite their difference, are reconciled further in Chapter 5 through the lens of their latent representation. In the second part, we describe the available defensive mechanisms placing our approach within the current landscape.

CHAPTER 3. EPISTEMIC UNCERTAINTY OF VARIATIONAL AUTOENCODERS

In this chapter, we implement our first guiding idea: measuring epistemic uncertainty over parameters with regard to inliers and outliers. This chapter answers **RQ2**. We explore how Bayesian inference over model parameters might improve the reliability of DNNs in identifying OoD inputs. Our study evaluates three approaches to Bayesian inference: stochastic gradient Markov chain Monte Carlo (MCMC), Bayes by Backpropagation (BBB), and Stochastic Weighted Averaging-Gaussian (SWAG), applied to the weights of DNNs that parameterize the likelihood of the VAE. We empirically assess these methods using benchmarks for OoD detection, including the estimation of the marginal likelihood, a typicality test, a disagreement score, and the Watanabe–Akaike information criterion (WAIC). Additionally, we identify a phenomenon of difference in variation of the model with respect to outliers versus inliers. Motivated by this observation, we propose two simple scores for OoD detection that show state-of-the-art performance based on the information entropy and sample standard deviation of the likelihoods, which confirm our assumption about the difference in the sensitivity levels.

CHAPTER 4. TOPOLOGY-AWARE APPROACH TO LATENT REPRESENTATION

In this chapter, we answer to **RQ3** with respect to OoDs. We implement our second guiding idea: enforcing an appropriate inductive bias to improve stability. In addition, we employ a topological lens to dive deeper into the sensitivity phenomenon. First, we highlight a significant theoretical flaw in the classical VAE model related to the assumption of infinite latent space support. To address this, we propose enforcing compactness in the latent space, making it possible to provably bound the image within certain limits, thus squeezing both inliers and outliers. This is achieved through the Alexandroff extension and setting a fixed Lipschitz continuity constant on the encoder mapping. The properly enforced compactness alleviates the mentioned flaw. Second, the most important result is that we demonstrate that anomalous inputs tend to land on vacant holes within the compact latent space simplifying their detection. It confirms our assumption that an appropriate inductive bias is beneficial for OoD identification. To that end, we introduce a new score for hole detection and evaluate the solution against several baseline benchmarks obtaining promising results. Moreover, we link the new score with sample standard variation of the likelihoods that demonstrate comparable results and can be utilized to detect outliers based on the difference in their sensitivity level.

CHAPTER 5. ENHANCING ROBUSTNESS OF DEEP LEARNING VIA UNIFIED LATENT REPRESENTATION

This chapter answers **RQ3** with respect to adversarial examples. We experimentally identify that these attacks can be successfully detected utilizing Bayesian inference over VAE parameters in the same vein as OoD inputs utilizing sensitivity. Furthermore, we disentangle Bayesian inference over parameters versus latent codes experimentally, achieving comparable results for both of the applied methods. In addition, we analyze the behavior of the adversarial examples in the latent space. Namely, we observe that they also tend to gravitate to the latent holes as OoDs. Moreover, we implement an algorithm to properly distinguish between inliers, OoDs, and adversarial inputs.

CHAPTER 6. CONCLUDING REMARKS

This chapter provides a reflection with regard to the posed research questions, including limitations and directions for potential future work, and finally discusses societal relevance.

REFERENCES

[1] Z. Fang, Y. Li, J. Lu, J. Dong, B. Han, and F. Liu. "Is Out-of-Distribution Detection Learnable?" In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh. Vol. 35. Curran Associates, Inc., 2022, pp. 37199–37213. url: https://proceedings.neurips.cc/paper_files/paper/2022/file/f0e91b1314fa5eabf1d7ef Paper-Conference.pdf.

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[3] S. Shalev-Shwartz, O. Shamir, N. Srebro, and K. Sridharan. "Learnability, Stability and Uniform Convergence". In: *Journal of Machine Learning Research* 11.90 (2010), pp. 2635–2670. url: http://jmlr.org/papers/v11/shalev-shwartz10a.html.

[4] S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. USA: Cambridge University Press, 2014. isbn: 1107057132.

[5] D. Wolpert and W. Macready. "No free lunch theorems for optimization". In: *IEEE Transactions on Evolutionary Computation* 1.1 (1997), pp. 67–82. doi: 10.1109/4235.585893.

[6] D. Wolpert and W. Macready. "Coevolutionary free lunches". In: *IEEE Transactions on Evolutionary Computation* 9.6 (2005), pp. 721–735. doi: 10.1109/TEVC.2005.856205.

[7] T. M. Mitchell. "The Need for Biases in Learning Generalizations". In: 2007. url: https://api.semanticscholar.org/CorpusID:3237155.

[8] Y. Ma, D. Tsao, and H.-Y. Shum. *On the Principles of Parsimony and Self-Consistency for the Emergence of Intelligence*. 2022. arXiv: 2207.04630 [cs.AI]. url: https://arxiv.org/abs/2207.04630.

[9] V. N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., 1995. isbn: 0-387-94559-8.

[10] B. K. Natarajan. "On learning sets and functions". In: *Machine Learning* 4.1 (1989), pp. 67–97. issn: 1573-0565. doi: 10.1007/BF00114804. url: https://doi.org/10.1007/BF00114804.

[11] D. A. McAllester. “PAC-Bayesian Stochastic Model Selection”. In: *Machine Learning* 51.1 (Apr. 2003), pp. 5–21. issn: 1573-0565. doi: 10.1023/A:1021840411064. url: https://doi.org/10.1023/A:1021840411064.

[12] A. N. Tikhonov. “On the stability of inverse problems”. In: *Proceedings of the USSR Academy of Sciences* 39 (1943), pp. 195–198. url: https://api.semanticscholar.org/CorpusID:202866372.

[13] O. Bousquet and A. Elisseeff. “Stability and Generalization”. In: *Journal of Machine Learning Research* 2.Mar (2002), pp. 499–526. issn: ISSN 1533-7928. url: http://www.jmlr.org/papers/v2/bousquet02a.html.

[14] S. Mukherjee, P. Niyogi, T. Poggio, and R. Rifkin. “Learning theory: stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimization”. In: *Advances in Computational Mathematics* 25.1 (July 2006), pp. 161–193. issn: 1572-9044. doi: 10.1007/s10444-004-7634-z. url: https://doi.org/10.1007/s10444-004-7634-z.

[15] E. Hüllermeier and W. Waegeman. “Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods”. In: *Machine Learning* 110.3 (Mar. 2021), pp. 457–506. issn: 1573-0565. doi: 10.1007/s10994-021-05946-3. url: https://doi.org/10.1007/s10994-021-05946-3.

[16] I. M. Alabdulmohsin. “Algorithmic Stability and Uniform Generalization”. In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc., 2015. url: https://proceedings.neurips.cc/paper_files/paper/2015/file/6512bd43d9c
Paper.pdf.

[17] S. D. Mbacke, F. Clerc, and P. Germain. “Statistical Guarantees for Variational Autoencoders using PAC-Bayesian Theory”. In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine. Vol. 36. Curran Associates, Inc., 2023, pp. 56903–56915. url: https://proceedings.neurips.cc/paper_files/paper/2023/file/b29500824d22ee9bbd25e4cd97c49b55-Paper-Conference.pdf.

[18] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. “Understanding deep learning (still) requires rethinking generalization”. In: *Commun. ACM* 64.3 (Feb. 2021), pp. 107–115. issn: 0001-0782. doi: 10.1145/3446776.

[19] M. Belkin, D. Hsu, S. Ma, and S. Mandal. “Reconciling Modern Machine-Learning Practice and the Classical Bias–variance Trade-Off”. In: *Proceedings of the National Academy of Sciences* 116.32 (2019), pp. 15849–15854.

[20] A. Jacot, F. Gabriel, and C. Hongler. “Neural Tangent Kernel: Convergence and Generalization in Neural Networks”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Vol. 31. Curran Associates, Inc., 2018. url: https://proceedings.neurips.cc/paper_files/paper/2018/file/5a4be1fa34e62b Paper.pdf.

[21] B. Bordelon, A. Canatar, and C. Pehlevan. “Spectrum Dependent Learning Curves in Kernel Regression and Wide Neural Networks”. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by H. D. III and A. Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 2020, pp. 1024–1034. url: https://proceedings.mlr.press/v119/bordelon20a.html.

[22] D. P. Kingma and M. Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

2

VULNERABILITIES AND DEFENSES OF DEEP NEURAL NETWORKS

Robustness of Deep Learning is an enormously vast and diverse topic. In this chapter we first concentrate on vulnerabilities of Deep Neural Networks and challenges of their mitigation.

We describe the applicable threat models and the corresponding range of potential vectors of attacks. They include attacks on Confidentiality, Integrity, and Availability (CIA). A special emphasis is done on the category of outliers that is traditionally not considered from the adversarial perspective. However, we claim that from the perspective of malcreant these inputs can be exploited in the same way as any other adversarially forged input if they allow reaching the final goal of malicious exploitation.

In the second part we address the currently available defense mechanisms that claim to demonstrate promising results in mitigating or detecting attacks including outliers.

This chapter is published in the book *Artificial Intelligence For Security: Enhancing Protection in a Changing World*, Springer, (2024) [1].

2.1. Introduction

Deep learning, powered by Deep Neural Networks (DNNs), underlies most modern Artificial Intelligence (AI) algorithms. These models have proven highly successful in various applications, including supervised, unsupervised, and reinforcement machine learning. Furthermore, deep learning models have demonstrated state-of-the-art results in processing diverse data domains such as images, texts, audio recordings, electrocardiograms, malware, games, and many more. Despite their impressive accuracy metrics, the fundamental theory of deep learning is missing. First, the most critical lacking aspect is linked to the generalization capabilities of the modern DNNs. Generalization can be subdivided into (i) generalization from training data to test data and (ii) generalization from inliers to outliers. Currently, there is no satisfying theory, even for the first part, which would explain the surprising generalization capabilities of DNNs. Suppose one employs the classical statistical learning theory based on the Vapnik–Chervonenkis dimension [2]. In that case, it likely indicates that DNNs are overfitting, considering that the number of parameters quite often significantly exceeds the number of data points used for DNN training. Many research works contribute to the gradual progress in this direction [3–6], but no final theory is currently available. The second part is also an active direction of current research [7–9]; however, nowadays, there is no theoretically rigorous foundation for that either. As a result, both of the parts require further exploration. Moreover, the second missing cornerstone relates to the optimization routine, e.g., to the convergence regime of DNNs: pointwise vs uniform. There are several studies in this direction [10–12] and their connections to the generalization [13] but, again, there is still no rigorous theoretical basis for that. Finally, the architectural choices for DNNs are still empirical, and no theory unifies them except for the very coarse results of the well-known Universal Approximation Theorem (UAT) [14] and its variations that basically aim at rigorous bounding either the depth [15–17] or the width [18] of the DNNs. The lack of rigorous understanding of the theory of deep learning results in many relevant questions that still need to be answered. For instance, how exactly does the model produce the final result? How can we be sure about these results? What are the limits of a particular modeling approach? How can it be rectified in case of malfunction? And many more others.

Therefore, these models are often used as a black-box tool, fed with significant data to achieve the desired performance results. However, this raises serious concerns about the robustness of these models to malicious interventions because if there is no clear understanding of how modern DNNs generalize from training data to test data (i.e., on inliers and, further, on outliers), then it is an obvious direction of exploiting this lack of understanding by probing the models for the potential vulnerabilities, especially when they are used in so many critical domains, ranging from autonomous driving to medical diagnosis. It should be noted that despite the significant progress in the methods of model interpretability such as SHAP [19], GradCAM [20], CAM [21], RISE [22] and CXPlain [23], all of these methods can be applied to any model, and they treat the model in question as a black-box with only potential access to the loss function and/or gradients. The interpretations are subsequently derived from input features that indicate the decision made by the model based on these input features and their corresponding contributions or weights for a particular decision. However, none of these methods tell anything about the model's internal mechanics from the fundamental theoretical perspective, e.g., how exactly did the particular DNN come up with this particular solution and why (all the research dedicated to the generalization and optimization of the DNNs)? How does the network architecture influence the final decision and why (vast volume of different ad-hoc solutions with too few theoretical foundations)? What can be done to alleviate the wrong output and modify the decision theoretically rigorously (the operational question that logically follows from the first two)? Can we debug the model, identify why these input features are essential for this model, and fix the wrong outputs? (i.e., the practical question based on the previous one).

Consequently, many practical attacks have been discovered that exploit this lack of understanding, which, in turn, assist us a lot in more profound research and comprehension of the internal mechanisms of deep learning. Let us momentarily take the role of an attacker and consider the potentially vulnerable spots that can be exploited from the general perspective in any machine learning modeling approach. Such an approach requires at least two parts: (*i*) data to be trained on and inferred from and (*ii*) a particular model to be trained and subsequently used for inference.

Moreover, the attackers may not have access to both model and data. In addition, both the model and data may evolve with time. Furthermore, a model can be a composite one, i.e., consisting of several other machine learning sub-models, and data can be non-homogenous or multimodal. Based on various combinations of these components, different potential scenarios are available for the attacker depending on their interests. For example, the attackers may be interested in modifying the model's behavior for particular inputs, e.g., they want to bypass a spam filter to organize an advertising campaign for an illegal product. Alternatively, they may want to trick a face recognition system to access the victim's smartphone. Conversely, the attackers may be interested in the data; for example, they may have access to the model and can get inference results but lack knowledge about the dataset used to train the model, which could be of particular interest to them. Alternatively, suppose the attackers cannot access the model. In that case, they may be interested in getting it, e.g., the model architecture, the weights obtained after training, and the hyperparameters used for training.

Therefore, grounded in the distinctive attack vectors, we can discern two principal categories of potential attacks. (*i*) *Functionality-oriented attacks*, encompassing those that seek to manipulate the model's behavior. These may transpire during either the training or inference stages. (*ii*) *Privacy-oriented attacks*, encapsulating those designed to extract insights into the private data employed during both training and inference stages since models may be trained on private data, such as personal information, medical records, or financial data, which can be extracted from the trained model breaching the privacy. Moreover, even if the training data is anonymized, recovering the data the model was trained on may leak proprietary business information. In addition, this attack category may aim to unveil proprietary facets of the machine learning model itself. If the model is deployed in the cloud and used via the web, then the model architecture and parameters leakage represent a serious proprietary trade secrets privacy breach.

The first kind of functionality-oriented attack strongly relates to the unknown inputs since we assume the model is not under the attacker's control. Hence, something should be done with its input to modify the model behavior. Unknown inputs can be further divided into two main categories. The first category includes inputs that differ significantly from those on which the model has been trained. For instance, if a deep neural network classifier is trained to classify handwritten digits, how will it process images of unknown numerical systems, languages, or even natural objects such as trees or dogs? Experiments demonstrate that DNNs overconfidently fail when processing such inputs as they cannot distinguish between them and those they have been trained on [24, 25]. Such inputs are closely related to the question of the proper model generalization, and they are referred to as outliers. The model deployed in the wild should be able to deal with the outliers, i.e., with the data points coming from a different distribution than the one the model trained on [26]. In such a case, the question arises of how good the model is in dealing with such inputs and, most importantly, if it can distinguish outliers versus inliers. The second category includes inputs specifically manufactured by adversaries to change the model's output, such as adversarial examples. These examples are becoming increasingly prevalent across all learning approaches, model architectures, and data domains.

The attacker utilizes outliers and adversarial examples in the inference stage when the model is already pre-trained. Alternatively, the poisonous attacks are introduced during the training stage. These are also artificially forged inputs that tend to be covertly introduced in the training dataset with the aim of the subsequent exploit during the inference stage.

The second kind of privacy-oriented attacks includes model stealing and membership inference attacks.

In this chapter, we briefly introduce the available attacking techniques and indicate the potential directions for their mitigation and, if possible, detection.

2.2. ATTACKING DEEP NEURAL NETWORKS

In this section, we introduce appropriate threat models together with potential vectors of attacks at DNNs. It includes both functionality-oriented and privacy-oriented threats. The former comprises outliers, adversarial, and poisonous examples. The latter includes attacks that aim at stealing private or sensitive data related either to the model parameters and architecture or to the inputs used during the training of the model, i.e., to the training dataset. Finally, these attacks are mapped within the Confidentiality, Integrity, and Availability triad components.

2.2.1. THREAT MODELS

We consider the two most common threat models concerning the attacker's abilities: white-box and black-box scenarios. The former relates to the situation when the attacker knows, for instance, the DNN model, including its architecture, learned weights, used hyperparameters, input layer representation, and output layer results. This scenario implies that the attacker has full knowledge of both model and training data in use.[1] Conversely, the latter applies to cases without direct knowledge of the DNN model. The attackers may only know the inputs used for the model. Sometimes, they can also use the model as a service via a particular API call to infer the outputs. Moreover, it usually denotes the lack of exact knowledge about the data used during the training stage of the model. Nevertheless, the general understanding of the data is particularly always available or can be inferred, such as, for example, in the case of face detection on the image, the training dataset should have included images with faces.

The privacy-oriented attacks imply the black-box scenario. Functionality-oriented attacks, on the other hand, may concern both threat models. The usual approach for them in the case of a black box is to train the so-called surrogate model, which will be used for the proper adversarial or poisonous input generations and the subsequent evaluations of the attack success rates. The properties of this surrogate model depend on the DNN under attack, and it usually entails the initial reconnaissance step with respect to the model in use potentially involving privacy-oriented attacks.

[1] Please note that it is not in contradiction with our previous assumption about the absence of the control over the model by the attacker since the knowledge of the model still does not imply the access to the model in use.

Beyond the available knowledge of the attackers, as we mentioned before, there is also a distinction between their possibilities over data manipulation, namely, if they can manipulate data during training or inference stages. The first type of attacks is called poisonous or sometimes causative, and the second type is called exploratory or evasion attacks. The latter includes both adversarial examples and outliers. We will use both mentioned terms for each of the types interchangeably.

2.2.2. FUNCTIONALITY-ORIENTED ATTACKS

First, we explore the most active domain of the current research dedicated to misleading the deep learning model behavior utilizing specific inputs. Since such inputs can be provided during the training or inference stage of the DNN life cycle, we dedicate a separate section to each stage.

EXPLORATORY ATTACKS ON INFERENCE

Inference relates to the stage when the model is already trained and validated on a particular dataset. Hence, it can now *infer* the parameters of the probabilities of the general population based on particular provided inputs. Two types of attacks can be associated with the inference stage of any DNN: (*i*) the attacks utilizing the outliers and (*ii*) the attacks employing the adversarial examples. In the following sections, we consider each of them in more detail.

OUTLIERS

We begin our discussion with the outliers. We consider that outliers are generated by a different distribution than the inputs used during training. An input generated from the different distribution represents a traditional definition of outliers, and to the best of our knowledge, it was first employed by Hawkins [26].[2] The treatment of these inputs is often considered somewhat parallel to the attacker's perspective in the scientific literature. The reason is that they are more vividly related to the question of the proper model generalization, i.e., to the ability of the model to infer the actual parameters of the probability distributions of the general population rather than to the attacking technique. Despite their traditional treatment in the research community, they can also be used to mislead the model behavior by an attacker; the model deployed in the wild should likely be able to deal with the data points coming from a different distribution to the one on which the model has been trained. For a toy example, let us consider the model trained on

[2] Hawkins distinguishes two categories of outliers: (*i*) inliers and outliers adhere to the same distribution and (*ii*) inliers and outliers generated by different distributions. In this chapter and this thesis, we use the latter.

the MNIST dataset [27]. This dataset contains the handwritten digits. The outliers in such cases could be the inputs from a different dataset, such as FashionMNIST [28] that contains various garments or even randomly generated images. In such a case, the question arises: *How good is the model in dealing with such inputs, and, most importantly, can it distinguish outliers versus inliers?*

To better understand what the outliers mean, it is helpful to pose the following question about a model in plain English: *Does a trained machine learning model know what it does not know?* To answer this question, we first have to understand what it means exactly for the model to deduce if it knows or does not know something. First, recall that DNNs represent universal mapping approximators, i.e., they can learn any function on a compact domain with an arbitrarily close level of precision. However, what type of functions do they learn in most of the applied tasks? *Probability functions!* More precisely, DNNs parameterize either probability density or probability mass functions based on a particular input. This chapter considers attacks on deep learning models based on the supervised discriminative approach, such as DNN classifiers. Hence, such a modeling approach implies parameterizing a conditional distribution over target values y conditioned on the input $\mathbf{x}$: $p_{\boldsymbol{\theta}}(y|\mathbf{x})$. The training of DNNs allows identifying optimum parameters $\boldsymbol{\theta}^*$ based on a stochastic first-order optimization algorithm such as gradient descent. In the case of classification tasks, the standard choice for $p_{\boldsymbol{\theta}}(y|\mathbf{x})$ is a categorical distribution, in the case of regression—a Gaussian distribution (quite often with a constant variance). It may seem only logical to rely on the probability density in answering our question, namely, if the density is low, then the model is uncertain about the input, which can be interpreted as the model not knowing the current input.

Nevertheless, despite the neat theoretical motivation, such an approach drastically fails in practice. The DNNs tend to assign relatively high probability values to the outliers. For example, consider an ImageNet dataset used in Large Scale Visual Recognition Challenge (ILSVRC) [29]. This dataset contains over a million images for 1000 object classes, including various types of animals, aircrafts, fruits, vegetables, etc. However, despite the vast coverage of potential objects, these 1000 classes do not include images of a microscope or a measuring tape. What would be the probability value for these unknown categories if we train a DNN classifier on ILSVRC based on those 1000 classes? Surprisingly, it turns out to be relatively high! For example, in the case of an image with a microscope, the DNN is 98.18% confident that it is a joystick, and in the case of an image with a measuring tape that it is a chainsaw with a 90.54% probability. Both joystick and chainsaw are present among those 1000 classes, but what is astonishing is the certainty with which the DNN claims to detect objects it has never seen before.

This experiment demonstrates an interesting fact about DNNs: they tend to be *overconfident* with the outliers in their predictions. Such overconfidence has at least two disadvantages. First, it makes it impossible to detect the outliers during the inference stage utilizing the probability values, making such DNNs less suitable for the tasks when such functionality is necessary. Second, it allows an attacker to exploit this overconfidence by allocating particular outliers and providing them as inputs to such DNN, misleading the initially intended behavior, e.g., resulting in misclassification. It is pretty easy to imagine that the impact of such an attack might be life-threatening in many domains, ranging from self-driving cars to medical diagnosis.

ADVERSARIAL EXAMPLES

Analysis of the input-output mappings of AlexNet Convolutional Neural Network (CNN) [30] from the point of view of continuity reveals unforeseen properties of the DNNs [31]. Namely, the traversal over the manifold learned by the CNN from one category to another with parallel visualizing of the corresponding points in the input space does not result in a smooth morphing of one category to another, e.g., imagine a gradual and substantial transformation of a dog class into a cat class in the resulting images. On the contrary, the results of such a traversal turn out to be quite intriguing; namely, when the CNN classifier indicates that it already observes a different class, the input image still contains a distinct representation of the initial category (e.g., a category of a dog) with a slight amount of the added noise over the image. This means that from the human perspective, the obtained result is almost indistinguishable from the initial image. However, from the perspective of CNN, the new image represents a different category. In parallel, predictably, analysis of the DNNs' robustness against the evasion attacks at test time [32] reveals the same intriguing behavior as in [31]. Two different perspectives on this behavior give rise to two definitions of the adversarial examples: one from the perspective of the generalization properties of the DNN and the other from the attacker's perspective.

From the generalization perspective, an adversarial example [31] is a technique in which the input for the DNN image classier is intentionally modified to look almost the same as the original image to the human eye. Yet, it is perceived as something completely different by DNN. DNNs incorrectly classify such adversarial examples from the human perspective. On the other hand, the attacker perspective does not necessarily demand the part that relates to the imperceptibility of the difference. On the contrary, if the miscreants want their attack's outcome to succeed, they should not constrain themselves to the superfluous imperceptibility demands. In this chapter, nevertheless, we are more inclined to the definition that involves the imperceptibility aspect. The reason for that is the new, often overlooked, category we include in our study: the outliers. These inputs could be considered similar to the perspective of the *adversarial attack to the maximum extent* when there is nothing similar between the original data and the outliers. In particular, we consider three methods of generating adversarial examples: (*i*) fast gradient sign method [33], (*ii*) Carlini-Wagner attack [34], and (*iii*) Jacobian-based Saliency Map attack [35].

Fast Gradient Sign Method. The Fast Gradient Sign Method (FGSM (FGSM) attacks DNNs by leveraging their learning process based on gradients [33]. The following formula describes the process of generating an FGSM example:

$$\mathbf{x}' = \mathbf{x} + \varepsilon \cdot \mathrm{sgn}(\nabla_{\mathbf{x}} \mathcal{L}(f_{\boldsymbol{\theta}}(\mathbf{x}), y_s)) \tag{2.1}$$

Here $\nabla_{\mathbf{x}}\mathcal{L}$ is the gradient of the loss function with respect to the original input pixel vector $\mathbf{x}$, y_s is the actual or source label for $\mathbf{x}$, and $\boldsymbol{\theta}$ stands for the parameters of the model $f_{\boldsymbol{\theta}}$ that are constant. Gradient w.r.t. $\mathbf{x}$ is easier to calculate with backpropagation than for $\boldsymbol{\theta}$, which allows the fast generation of adversarial examples. FGSM exploits gradient ascent to increase the loss. Subsequently, the sign applies a max-norm constraint on the gradient value, and ε represents a small magnitude of the step to increase the loss. This formulation constitutes the untargeted type of adversarial attacks, a particular example depicted in Figure 2.1.

The FGSM can be converted into a targeted attack by substituting the source label with a target one y_t and doing gradient descent instead of ascent, namely:

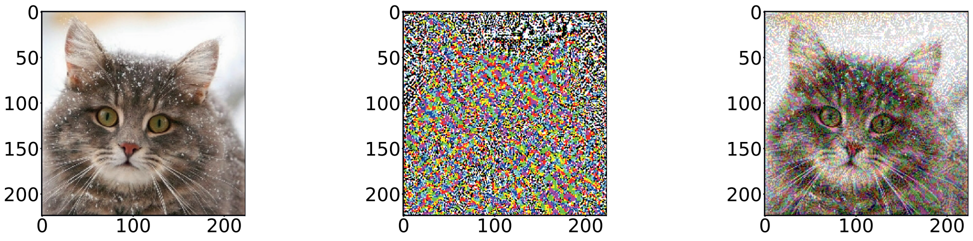
$$\mathbf{x}' = \mathbf{x} - \varepsilon \cdot \mathrm{sgn}(\nabla_{\mathbf{x}} \mathcal{L}(f_{\boldsymbol{\theta}}(\mathbf{x}), y_t)) \tag{2.2}$$

However, since FGSM is designed to be fast rather than optimal, it is not necessarily guaranteed to produce the targeted adversarial examples of minimal perturbations.

Carlini-Wagner. The Carlini-Wagner (CW) attack [34] aims at optimality in contrast to FGSM, i.e., it attempts to generate as little noise as possible to succeed in the attack. It poses the following optimization objective:

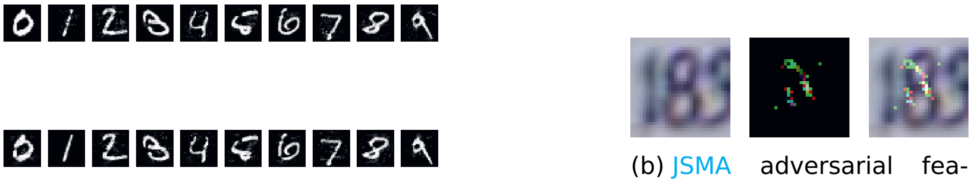
$$\text{minimize } ||\delta||_p \text{ subj. to } f_{\boldsymbol{\theta}}(\mathbf{x}+\delta) = y_t, \quad \mathbf{x}+\delta \in [0,1]^n \tag{2.3}$$

where, $\mathbf{x} \in [0,1]^n$ represents an image, $\delta \in [0,1]^n$ is the added noise to the image, and $f_{\boldsymbol{\theta}}$ is a model that returns a target class label of the image under attack. The noise level is calculated in terms of L_p norms. The authors consider several norms; our example demonstrates the L_2-norm. The CW attack represents a targeted attack with powerful properties. Till the moment, it is one of the strongest known adversarial attacks. The examples of the CW-targeted attack on MNIST digits can be observed in Figure 2.2.



(a) Benign image of Persian cat
(b) Constrained gradient of the loss
(c) Peacock ($\varepsilon = 0.25$)

Figure 2.1: (a) the original image; (b) the max-norm constrained gradient of the loss generated by FGSM; (c) the resulting adversarial image.



(a) CW MNIST adversarial examples
(b) JSMA adversarial features

Figure 2.2: (a): Carlini-Wagner targeted attacks. *Top row:* MNIST adversarial examples targeted for 0. *Bottom row:* MNIST adversarial examples targeted for 1. (b): Untargeted JSMA adversarial features visualized for central number.

Jacobian-based Saliency Map Attack. Jacobian-based Saliency Map Attack (JSMA) [35] leverages the saliency maps to devise an adversarial input. Namely, it computes the forward derivative of the whole DNN (Jacobian) w.r.t. the input, and based on this derivative, it constructs the saliency map. Large absolute values of the saliency map reveal the features significantly impacting the final output. The JSMA takes the maximum absolute value, perturbs it by a hyperparameter θ, and repeats the process. The stopping criteria are either a successful attack with misclassification or reaching the total perturbation threshold of Υ. Figure 2.2 depicts such a JSMA attack.

2.2.3. CAUSATIVE ATTACKS ON TRAINING

This type of attacks implies that the attacker can manipulate some part of the dataset to be subsequently used for training the DNN model under attack. Since this manipulation is always intended as malicious, the term *poisonous* describes this fact rather appropriately and precisely. Generally speaking, there are two possible malicious intents: they aim at intrusion without corrupting the system behavior with respect to benign inputs or target an overall regular system operation, causing a denial of service. First, we consider the way the training data is modified.

POISONING DATASET

Construction of a poisonous dataset includes collecting raw samples of the statistical population distribution of interest with their subsequent labeling. Depending on the attackers' capabilities, they may be able to modify raw samples or only their corresponding labels (or both). The first attack is called a clean-label attack, and the second is a dirty or corrupted-label attack. The clean-label attack may seem impossible at first since there is no control over the output category, and it is unclear how the attacker can exploit the training process in such a case. Nevertheless, two known attacking techniques allow us to achieve this goal. Both introduce slight disturbances to the inputs that mislead DNNs during training. These disturbances are identified through the solution of either a bi-level optimization problem or a feature-collision problem.

DATA POISONING TO DISRUPT OVERALL PERFORMANCE

This type incorporates several indiscriminate attacks, i.e., attacks that do not target a specific category or class within the training dataset. First, we describe a corrupted-label attack based on label flipping. After that, we consider a clean-label attack based on bi-level optimization.

Label Flipping Attack. The attacker can significantly reduce the resulting DNN performance only via tampering with the labels of the dataset. It means that it is optional to have access to the model and the raw data. This type of attack can be implemented by mislabelling the data. Labels can be assigned randomly, influencing the system's overall performance; the specific classification category or even several categories can be targeted. The classifier is subsequently trained on the tampered dataset without knowing which labels have been corrupted. The success rate of this type of attack heavily depends on the ratio of the poisoned dataset: the greater the ratio in favor of poisonous examples, the stronger the impact it would have on misclassification.

Attacks via Bi-level Optimization. This type aims to solve the corresponding bi-level optimization problem and represents clean-label attacks. Consider the following scenario: attackers want to attack a particular DNN model that will be trained, validated, and tested on the dataset $\mathcal{D}$. They want to poison this dataset to change the resulting model behavior. For that reason, the attackers split this dataset into training and validating subsets $\mathcal{D} = \mathcal{D}_{\text{tr}} \cup \mathcal{D}_{\text{val}}$. Subsequently, they aim at poisoning the training dataset $\mathcal{D}_{\text{tr}}$ with a specially crafted input $\mathbf{x}_p$. The crafting of the poisonous input is constrained within the space of allowed manipulations imposed by Φ, i.e., $\mathbf{x}'_p \in \Phi(\mathbf{x}_p)$. As a result, they get a poisoned trained dataset $\mathcal{D}_p = \mathcal{D}_{\text{tr}} \cup \{\mathbf{x}'_p\}$. The goal is to optimize for the optimal poisonous solution allowed within the imposed constraints such that the target input $\mathbf{x}_t$ with the original label y_t from the untampered validation dataset $(\mathbf{x}_t, y_t) \in \mathcal{D}_{\text{val}}$ is misclassified by the model trained on the poisoned dataset.

To achieve this goal, the optimization procedure should be run across two levels. First, the model should be optimized with respect to the DNN parameters on the dataset with the injected poisonous examples, which represents a classical optimization objective of any DNN training, usually referred to as a training loss $\mathcal{L}_{\text{tr}}$. Second, on top of the classical optimization objective, there is an additional adversarial loss $\mathcal{L}_{\text{adv}}$ that should be optimized for the best poisonous input modification in such a way that this adversarial loss is maximized which eventually leads to the sought-after misclassification. Note that the misclassification is untargeted in this particular case. Hence, this formulation of the optimization objectives represents an attack on the availability of the model aiming at disrupting the classification results for the poisonous inputs.

Formally, both of the optimization levels can be formulated as follows:

$$\mathbf{x}^*_p \in \arg\max_{\mathbf{x}'_p \in \Phi(\mathbf{x}_p)} \mathcal{L}_{\text{adv}}(\mathcal{D}_{\text{val}}, \boldsymbol{\theta}^*(\mathbf{x}'_p)) \tag{2.4}$$

$$s.t. \quad \boldsymbol{\theta}^* \in \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{(\mathbf{x}'_p, y) \sim \mathcal{D}_p}[\mathcal{L}_{\text{tr}}(f_{\boldsymbol{\theta}}(\mathbf{x}'_p), y))] \tag{2.5}$$

where $\mathbf{x}'_p \in \mathcal{D}_p$, i.e., it is an input from the poisoned training dataset under constrained manipulations allowed by Φ, and $\mathbf{x}^*_p$ is the resulting poisonous input to be added in the original clean dataset.

The process of optimization of this bi-level problem seems straightforward at first glance; the inner minimization procedure obtains the optimal parameters of the DNN with the subsequent maximization of the adversarial loss. This process can be repeated until the convergence. However, in the case of DNNs, this solution is not feasible due to the over-parameterization and impossibility of the exact solution of the inner problem. For that reason, the truncated back-gradient optimization is used [36]. The idea is to first optimize the inner problem for a limited number of iterations, which allows computing the necessary gradients with subsequent backpropagation for the outer objective by tracing the necessary gradients backward. This approach allows the generating of poisonous inputs for DNNs successfully.

TARGETED POISONING ATTACKS

We consider one corrupted-label targeted attack based on a bi-level optimization objective and one clean-label targeted attack utilizing feature collision.

Attacks via Bi-level Optimization. The indiscriminate bi-level optimization poisonous attack described above can be easily changed into a targeted one, if the outer objective is minimized instead, namely:

$$\mathbf{x}_p^* \in \arg\min_{\mathbf{x}_p' \in \Phi(\mathbf{x}_p)} \mathcal{L}_{\text{adv}}(\mathcal{D}_{\text{val}}', \boldsymbol{\theta}^*(\mathbf{x}_p')) \tag{2.6}$$

The dataset for validation contains the same raw inputs as $\mathcal{D}_{\text{val}}$ but with the modified labels that the attacker wants to target, hence minimizing the loss towards them.

Attacks on Feature Collision. The intuitive idea behind this type of attacks is to look for the neighbors in the feature space within a relatively close distance from the target class that simultaneously lie close in the input space to the base class, i.e., to the class under attack:

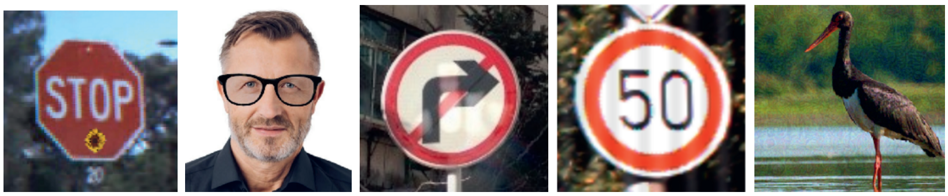
$$\mathbf{x}_p^* \in \arg\min_{\mathbf{x}} \; \|f_{\boldsymbol{\theta}}(\mathbf{x}) - f_{\boldsymbol{\theta}}(\mathbf{t})\|_2^2 + \beta \, \|\mathbf{x} - \mathbf{b}\|_2^2 \tag{2.7}$$

Given the feature space's usual complexity and high dimensionality, such a search appears feasible. Note that due to the closeness to the target class in the feature space, there is no need to modify the label, making this attack clean-label. Moreover, the closeness of the poisonous sample to the base one in the input space does not raise concerns during the visual inspection since the label is correctly assigned as if it represents a base class. Both of these factors make this attack very powerful.

DATA POISONING FOR INTRUSION

In this category, the attacker introduces a set of poisonous inputs that contains a specifically crafted trigger that serves as a backdoor, allowing the intruder to modify the behavior of the DNN in the desired direction [37].

Backdoor Patches. The first attack from this category exploits virtual or physical visible patches [38]. The training dataset is poisoned with the patched instances. Subsequently, the DNN is trained with the *corrupted-labels* for the samples that contain the corresponding patches. These patches serve as a trigger for the DNN classifier, allowing the attacker to change the classification result of the model. Since the training dataset contains benign and patched inputs, the normal behavior of the model with benign inputs and classification test errors are not influenced by these patched samples. Examples may represent a patch on the traffic



(a) Patch (b) Semantical (c) Blending (d) Blending (e) Feature Collision

Figure 2.3: Different types of backdoor attacks: (*a*) artificial patch with a corrupted label, (*b*) benign semantical trigger with a corrupted label, (*c*) blending a fake reflection with a clean label, (*d*) introducing a strong signal with a clean label, and (*e*) backdoor imperceptible to the human eye with a clean label.

stop sign to misclassify it as a traffic sign for the speed limit (see Figure 2.3a). Please note that such a patch can be imposed with the original image during the construction of the poisonous input without needing a physical photo with the patch. However, sometimes it is reasonable to rely on some physical trigger that carries a distinct semantical meaning but looks benign in contrast to the patching, for example, when a person wears glasses (see Figure 2.3b).

Backdoor as a Strong Signal. Several functional approaches could be applied to implement a *clean-labels* backdoor. First, it is possible to bind a strong signal function that any DNN can easily detect with a particular target class, e.g., a sinusoidal signal with a ramp can be mixed into the small fraction of the target images with subsequent use of this signal as a backdoor for any other image during the inference stage. The sufficient ratio of poisonous examples before training the model for the successful attack constitutes one-third of the target class for the MNIST dataset and one-fifth of the traffic signs dataset [39] (see Figure 2.3d). The problem with the sinusoidal ramp is that it is still quite visible in the resulting image, which can be detected via visual inspection of the dataset.

Another type of signal can be stealthily encapsulated into the target image representing a fake reflection [40]. The benefits of this approach are that it cannot be easily detected as the previous signal and taking into consideration that it is also a clean-label poisonous attack then it provides all the benefits of a nice backdoor in cases when the target images have reflecting surfaces (see Figure 2.3c).

Imperceptible Backdoor. Previous backdooring techniques still require a visible trigger and a relatively high ratio of poisonous examples to be added to the training set. The ultimate backdoor, however, can be forged utilizing the technique that we have already described above, namely feature collision [41]. Such an approach allows the injection of a backdoor even by poisoning one image only. Moreover, thanks to the optimizing within the deep feature representation while minimizing the difference in the input domain, the resulting poisoned image contains imperceptible perturbances to the human eye, making this backdoor one of the most potent poisoning attacks (see Figure 2.3e).

2.2.4. TRANSFERABILITY

It has been discovered that different architectures of DNNs trained to tackle the same classification problem on similar datasets tend to have similar fairly piece-wise linear decision boundaries that separate categories in the input data domain [33]. This property is called transferability. Transferability is especially dangerous since it allows devising either an adversarial example or poisonous input that universally targets all DNNs with a similar final objective in a black-box manner [42].

The thorough study of the transferability properties indicates the difference between adversarial and poisonous samples. In particular, the adversarial examples tend to transfer better when forged on a simplified surrogate model. However, for poisonous attacks, the best surrogate models are the ones that match the complexity of the target [43]. Moreover, in the case of the white-box threat model, both adversarial and poisonous inputs favor the more complex models, i.e., the success rate of the attacks is higher for a more complex model than for a simpler one.

2.2.5. PRIVACY-ORIENTED ATTACKS

These attacks aim at stealing any private data. The first type of privacy-oriented attacks involves stealing the pre-trained model's proprietary parameters, including weights and hyperparameters. The model in this scenario is usually queried remotely via the web. The second type relates to identifying if a particular input was in the training dataset, allowing the attacker to deduce the data on which the model under attack was trained. In this section, we describe both of these types.

MODEL STEALING

Nowadays, the functionality provided by various DNNs is often offered online as a service. Such practice amplifies the significance of securing the privacy-related information related to the models deployed in the cloud since its leakage may induce confidentiality-related infringements and substantial financial costs.

Attacks that aim at model stealing attempt to extract this information somehow and, eventually, to reconstruct the existing target model. It may concern the reconstruction of the entire model along with its weights, the reconstruction of the hyperparameters, or the reconstruction of a functionally equivalent alternative model. The adversary gathers the data about the target model by sending a data sample and receiving the model's prediction. This interaction is termed *query-based*.

Equation-solving Attacks. Equation-Solving Attack (ESA) is based on formulating and solving a system of equations, the solution of which yields desired values for an adversary. It aims at specific values of the target model, e.g., learned parameters or training hyperparameters. Using model outputs $y_1, \ldots, y_n$ for given data samples $\mathbf{x_1}, \ldots, \mathbf{x_n}$, it's possible to construct equations $f_{\boldsymbol{\theta}}(\mathbf{x_i}) = y_i, i = 1, \ldots, n$, revealing parameter values $\boldsymbol{\theta}$ [44]. ESA is quite efficient: depending on the target model type, 1 to 4 queries per parameter is enough. This attack requires the knowledge of the target model's architecture and the data samples to query the model. For black-box access, training hyperparameters need prior architecture and parameter extraction attacks.

Meta-model Training Attacks. The Meta-Model Attack (MMA) is the only query-based attack capable of uncovering target model architecture to date. A meta-model is trained using a set of candidate CNNs with varying architecture, optimization, and data parameters as the meta-model's dataset, predicting a model's structure, training setup, and data volume. The meta-model then links model hyperparameters and performance through specific test samples, revealing target model hyperparameters [45].

MMA's success depends on the hyperparameter's influence and their presence in the training set. For instance, to steal the convolutional layer count, the target model must be a convolutional neural network and possess a corresponding number of layers in the training set for the meta-model. Execution demands considerable computational resources: for MNIST classifiers, 10,000 CNN candidates had to be trained for over 40 days on a GPU. On average, the attack correctly predicted hyperparameters 80.1% of the time, surpassing a 34.9% guessing chance. However, as MMA extracts hyperparameters, an extra parameter-stealing attack is necessary to approximate the whole target model behavior [46].

MEMBERSHIP INFERENCE

Membership inference involves the identification of training data associated with a trained model. When provided with a data instance and granted access to a model, the objective is to detect whether this data instance was a part of the model's training dataset. The access to the model can vary, falling into either the white-box or black-box category.

Membership inference attacks stem from the fact that a machine learning model might behave differently on the training dataset than the test dataset. It is particularly evident in machine learning, especially within

DNNs, which often are over-parameterized when the number of trainable parameters exceeds the number of training instances. It enables a machine learning model to “remember” instances from the training data. Thus, it may assign significantly higher confidence to predictions for training instances than test instances. By exploiting such differences, attackers can deduce whether a given instance is likely to belong to the training dataset.

2.2.6. CIA TRIAD

The classical information security triad comprises three main components: Confidentiality, Integrity, and Availability (CIA). We can now look at the described attacks through the CIA’s prism. The functionality-oriented attacks relate to both Integrity and Availability. The integrity component involves the attacks that maximize the model Type-II errors, i.e., adversarial examples and backdoor attacks. Availability attacks aim at minimizing the utility of the DNN model, e.g., by increasing the model Type-I errors via poisonous attacks or by flooding the model with outliers, exploiting model overconfidence, and making the model predictions irrelevant in most cases. The privacy-oriented attacks, however, are connected with the Confidentiality component. It includes both model stealing and membership inference. The summary of all of these aspects, including the attacker’s capabilities, can be observed in Table 2.1.

2.3. AVAILABLE DEFENSES

This section covers several available defense strategies against the attacks mentioned above.

2.3.1. DEFENSE AGAINST FUNCTIONALITY-ORIENTED ATTACKS

As for the available defenses against functionality-oriented attacks, it should be stressed that the current state-of-the-art represents a constant race between the invention of new defenses against known attacks and the subsequent breaking of this defense with the newly discovered vulnerabilities. Nevertheless, in this section, we introduce the defense methods that demonstrate promising robustness results, and many of them have already passed the test over time.

ADVERSARIAL TRAINING

Currently, there is no readily available universal defense mechanism that provides complete protection against adversarial examples. Nevertheless, several techniques proved promising in mitigating the potential damage and consequences, one of them being adversarial training [31, 33]. The transferability not only allows a black-box adversarial attack

Table 2.1: Summary of the attacks and links to the CIA triad. Enhanced from Biggio and Roli [47]

Attacker's Capability	**Attacker's Goal**		
	Functionality-oriented		**Privacy-oriented**
	Integrity	**Availability**	**Confidentiality**
Train data	Poisoning for subsequent intrusion	Poisoning to invalidate model	—
Test data	Adversarial examples	Outliers	Model stealing, membership inference

but also means that it is possible to generate the known adversarial examples in advance automatically and add them to the training set before starting the training. Such an approach forces the DNN model to consider the adversarial perturbations and increases the robustness of DNNs to adversarial attacks.

The advantage of this method is its simplicity. The most significant disadvantage is the requirement to generate adversarial examples in advance, which may defend only against attacks known during the DNN model training. Since adversarial examples can be generated with a vast diversity of different techniques, this approach can be helpful only if the model retraining and redeployment procedures are relatively cheap. Thus, keeping the model up-to-date with developing new attacks is possible.

ENHANCED OPTIMIZATION OBJECTIVE FOR ADVERSARIAL TRAINING

The idea of this approach is first to enhance the standard optimization objective and then exploit one of the adversarial attacking techniques for approximating one of the stages of the optimizations. Namely, recall the standard loss objective for the classification task:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\mathcal{L}(f_{\boldsymbol{\theta}}(\mathbf{x}), y))] \tag{2.8}$$

where $\mathbf{x}$ is the input, y is the corresponding label, $\mathcal{D}$ is the training dataset, $\boldsymbol{\theta}^*$ are the optimal trained parameters of the DNN.

This objective may be improved so that its exact solution would *guarantee* the robustness against the adversarial examples [48]. Specifically, it can be achieved by requiring that every neighbor of the current point within the ε-ball adheres to the same class:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\max_{\|\mathbf{x}'-\mathbf{x}\|_p\leq\varepsilon} \mathcal{L}(f_{\boldsymbol{\theta}}(\mathbf{x}'), y))\right] \tag{2.9}$$

Unfortunately, the exact solution for optimizing this min-max objective is unattainable in a reasonable amount of time since it involves a standard process of first a non-concave inner maximization and then a non-convex outer minimization. The solution is to approximate the worst-case inner approximation problem by optimizing for adversarial examples. In particular, FGSM (see 2.2.2) can be interpreted as a single step for maximizing the inner part of this objective formulation. In order to get a more precise approximation to the solution, one can also apply a multi-step attack such as projected gradient descent (PGD) [34].

OUTLIER EXPOSURE

Similar to the idea of adversarial training is the outlier exposure approach [49], introduced for DNNs to deal with the outliers. The idea is to enhance the training dataset of the inliers $\mathcal{D}_{\text{in}}$ with the outliers. Since the dataset with the real outliers $\mathcal{D}_{\text{out}}$ is not available, it is though possible to enhance the dataset with the different available datasets for Outlier Exposure $\mathcal{D}_{\text{out}}^{\text{OE}}$. This dataset is different from the inliers. The model f is subsequently trained to learn more conservative signals of the inliers versus provided outliers that enhance robustness against previously unobserved outliers. It is achieved by introducing an additional term to the loss function, which is responsible for the outlier detector:

$$\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{in}}}[\mathcal{L}(f(\mathbf{x}),y)+\lambda\mathbb{E}_{\mathbf{x}'\sim\mathcal{D}_{\text{out}}^{\text{OE}}}[\mathcal{L}_{\text{OE}}(f(\mathbf{x}'),y)]] \tag{2.10}$$

The first term of the loss is the standard cross-entropy loss $\mathcal{L}$ used in the classification tasks where $\mathbf{x}$ represents the input, and y stands for the corresponding label. The second term includes the outlier exposure loss $\mathcal{L}_{\text{OE}}$ for the outlier detector. It represents the cross-entropy from $f(\mathbf{x}')$ to the uniform distribution. Unlike the adversarial training approaches described in the previous sections, this method allows us to mitigate and detect the outlier.

DEFENSE VIA GENERATIVE MODELS

Several defenses based on the Deep Generative Modeling (DGM) do not imply any knowledge about the type of adversarial attack in use. Moreover, they can be applied independently of the classifier DNN since they are used as a filter working with the input before passing it further to the DNN model under protection. In addition, some DGMs allow the detection of outliers in a completely unsupervised manner that can be combined with the adversarial filtering approach, enhancing the robustness against outliers and adversarial examples.

DGMs as Adversarial Filters. This approach exploits the DGMs' ability to learn a joint distribution of the data that results in a different learned representation compared to the discriminative approach, e.g., in the case of a DNN classifier. The most obvious difference with the discriminative representation is that the generative representation allows the generation of new samples from the learned joint distribution that look similar to those observed in the dataset during the training stage. Based on this representative power, it becomes possible to train a separate DGM model that will work as a filter, i.e., any input is first fed to the DGM filter, which mitigates the ongoing or attempted attack [50, 51]. The mitigation requires normal DGM training on the same dataset as the DNN under protection. As a result, it allows projecting the adversarial input onto the range of the DGMs' generator based on the learned representation by minimizing the reconstruction error. Hence, by default, this method includes filtering out the adversarial perturbations. Finally, the protected DNN classifier will get the clean input without adversarial perturbations.

DGMs as Unsupervised Outlier Detectors. The methods within this category allow the detection of outliers completely unsupervised. They can rely either on the ensemble-based epistemic uncertainty estimation [52] or the DGM latent representation [53]. In both cases, the DGM can be again used as a filter to the DNN under protection. However, this time, it is not only mitigating the attack but also detecting if the current input falls under the category of outlier or not. Moreover, it turns out that the outliers tend to gravitate toward the holes in the latent representation of DGMs (see Figure 2.4), specifically in the Variational Autoencoders (VAEs), allowing its simple detection. Finally, the same DGM that is used for the outlier detection can be applied in the same logic as the previously described DGMs for the adversarial mitigation, i.e., if the outlier is not detected, then the input is purified from the adversarial perturbations and subsequently fed to the DNN under protection. This approach is auspicious since it targets adversarial examples and outliers utilizing only one filtering model.

DEFENSE AGAINST POISONOUS ATTACKS

Defenses against poisonous attacks can be broadly categorized into one of the following groups:

1. *Data Level Defenses:* Focus on removing poisonous data from the input.
2. *Model Level:* Involve model retraining.
3. *Interaction between Data and Model Levels:* Identify patterns in the interaction between the data and model training.

Data Level Defenses. The first approach to the defense in this category is treating the poisonous samples in the tampered dataset as outliers. It implies that the defenders can access the benign dataset to train an outlier detection model. For example, they can train another model on the benign dataset with a different architecture from the original one but with comparable accuracy metrics [54]. Subsequently, any new input can be forwarded through both models, and if there is disagreement

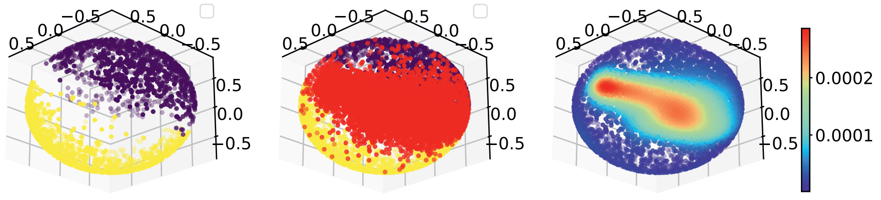


Figure 2.4: Compact spherical latent space of VAE trained only on two digits of the MNIST dataset: 0’s and 1’s. It is a vivid demonstration of the fact that the outliers densely land on the hole in the latent representation. *From left to right*: Yellow depicts means of the estimated posteriors for 1’s and purple for 0’s; Red represents the mapped means for a held-out outlier: a class of digits 9’s; Kernel density estimation of all means with the densest region in the hole packed with the outliers.

in the predictions between them, then this input is marked as potentially poisonous.

Another approach is based on the intentional input perturbation when the defender modifies the input by blending it with another benign input. It has been noticed that the poisonous backdoor inputs are still successfully classified as the target of the backdoor, whereas the blending of two benign inputs produces a random prediction result [55]. This difference is most likely due to the special optimization procedure used when forging a backdoor input, in which the objective is set so that it should overcome the original trigger “blending”, resulting in a successful attack even when additional blending is applied.

Model Level Defenses. The defenses of this type imply that the defender has access at least to the smaller subset of the original benign dataset. In such a case, the defender can run a training procedure based on fine-tuning the poisoned model utilizing this benign subset. The benign data can be simultaneously augmented with the samples by adding random Gaussian noise to make the defended model even more robust to potential poisonous input perturbations. After several iterations of such fine-tuning, the model becomes robust and immune to the previously poisonous inputs [56, 57].

An alternative approach within this type of defense is based on meta-classification when an additional DNN is trained on the features extracted from the defended DNN to classify if this model has been compromised by poisoning. The defender has to construct a set of DNNs evenly divided into poisoned and benign ones. After that, the meta classifier is trained based on the features extracted from these DNNs with a single purpose to distinguish between the benign and tampered models. This approach demonstrated good generalization results even with the before unseen poisonous techniques [58, 59].

Defenses based on the Interaction between Data and Model Training. This type of defense is relatively novel and looks like the most robust and promising one. It does not imply that the defender has access to the benign dataset for outlier detection or a benign subset of the initial dataset for subsequent model retraining using fine-tuning. It does not rely on the meta-classification either, implying that the defender has access to benign and poisonous models. Instead, this defense is working its way out utilizing the provided dataset and the given training objective interchangeably, resulting in the clustering of the samples based on the incompatibility property [60]. The intuition is that the benign inputs should improve the optimization objective during the training or at least not degrade it; however, the poisonous inputs, on the contrary, should reduce the validation accuracy. This idea leads to the clustering method that iteratively builds up clusters based on the incompatibility with respect to the training objective, i.e., benign inputs will eventually generalize to themselves as opposed to the poisonous inputs that would be clustered separately in such cases.

2.3.2. DEFENSE AGAINST PRIVACY-ORIENTED ATTACKS

In this section, we cover both mitigation and detection defensive methods against privacy-oriented attacks. The mitigation methods focus on minimizing the attack's impact, i.e., they do not stop the attacker from acquiring a model; their goal is to reduce the stolen model's quality to a point where it becomes unusable. The detection methods can further be subdivided into ownership verification (usually achieved by unique model identifiers or watermarking that can prove ownership of a stolen model; aims at proving past attacks) and attack detection (monitors whether a model is currently being attacked). Attack detection cannot prevent a model from being stolen, but it can inform the owner about the incident.

BASIC DEFENSES

We begin with a listing of several simple basic defenses that can be applied to mitigate attacks that aim at stealing the DNN model. These defenses demonstrate good protection results despite their simplicity.

Input Modification Defenses. The first type involves input modification, i.e., a defender can modify the input provided to the DNN so that only insignificant parts of this input are modified. For example, in the case of CNNs and image classification tasks, these parts would represent insignificant pixels for the resulting classification. If the defender adds random noise to these pixels, it will make it very complicated for the attacker to steal the DNN's parameters [61]. The technique that identifies such insignificant pixels in the first place is based on the Gradient-based

CAM [20] that was initially developed for the visual interpretability of the decisions made by CNN in the input feature space.

Ouput Modification Defenses. The second type implies output modification. It is based on rounding of the predicted values [44]. This rounding prevents the attacker from stealing the model's parameters due to the impossibility of solving the corresponding system of equations. Another alternative is to utilize the so-called adaptive misinformation [62]. It is a special technique to return wrong predictions for user queries but in an adaptive way. The intuition behind this approach is to notice that model attackers quite frequently use queries that lie out-of-the-distribution. Based on this observation, it is logical to adapt a model so that it will assign wrong predictions to such queries, making it much more complicated for the attacker to steal the model.

Model Modification Defenses. In addition to the perturbation of inputs and outputs, modifying the model architecture and parameters is possible. A CNN feature extractor can be simulated to train a simpler model, namely a shallow and sequential convolutional block, via Knowledge Distillation [63]. Such an approach is akin to the source code obfuscation technique obstructing the attacker from stealing the initial proprietary model parameters and architecture. Moreover, it is possible to choose an opposite way by adding redundant layers that do not change the functionality, which makes theft of the model much more complicated [64].

UNIQUE IDENTIFIERS AND WATERMARKING

This section addresses several ownership verification defensive techniques.

Unique Model Identifier. Unique Model Identifier (UMI) is a detection method that identifies a distinctive model property that transfers to a substitute model during theft. By revealing this property, a model owner can prove the model was stolen. This technique does not require an active embedding of this unique, distinctive property in contrast to watermarking, as it is inherent to the model. One example of this technique is a Dataset Inference (DI) defense. DI detects if a model was trained on a specific dataset [65]. This method is based on the idea that training samples lie farther from the decision boundary than other samples. A subset of the original training data measures the distance of samples from the boundary in the substitute model. If they are distant, the substitute model contains the target model's identifier, indicating a potential model theft. However, DI is ineffective when the original dataset is public, misclassifying independent models trained on it as stolen [66]. Alternatively, conferrable adversarial examples can form unique fingerprints for substitute models [67]. These samples represent a unique set of adversarial examples that transfer to the substitute models but not to the other independently trained models.

Model Watermarking. Another approach is based on the active intervention in the model training process so that the owner changes the model behavior on some or all inputs so that only the owner can identify these changes. Model watermarking actively embeds concealed data to establish ownership. Unlike UMIs, it involves secret backdoors that make models predict predefined values for some data containing outliers, revealing watermarks. However, embedded watermarks also have to be persistent to model stealing, i.e., they have to be identifiable in the model after it has been stolen. This property can be achieved by training a model that extracts common features from problem-domain inputs and watermarking samples. Such an approach ensures that watermarks would also be extracted during the model theft [68].

MONITOR-BASED DEFENSES

Monitor-based detection defense involves query analysis to identify malicious users. For example, a defender can train a dedicated discriminative DNN to classify adversarial versus benign samples, using each hidden layer outputs of a protected DNN as features [69]. Alternatively, a DGM such as VAE can also differentiate benign from malicious queries [70].

2.4. CONCLUSION

In this chapter, we have covered the broad range of attacks on discriminative DNNs. We have touched upon all potentially vulnerable spots, including the influence on the model's behavior and the potential confidential data leakage. In addition, we detailed the most prominent examples of adversarial and poisonous attacks, including the potential adversarial usage of the outliers and various backdoor techniques. Furthermore, we delved into two powerful privacy-related attacks: model stealing and membership inference. Lastly, we highlighted the most promising and robust defense mechanisms that are currently available in the arsenal of the DNN developer to either mitigate or detect the undergoing attack.

REFERENCES

[1] M. Glazunov and A. Zarras. “Who Guards the Guardians? On Robustness of Deep Neural Networks”. In: *Artificial Intelligence for Security: Enhancing Protection in a Changing World*. Ed. by T. Sipola, J. Alatalo, M. Wolfmayr, and T. Kokkonen. Cham: Springer Nature Switzerland, 2024, pp. 103–127. isbn: 978-3-031-57452-8.

[2] V. N. Vapnik and A. Y. Chervonenkis. “On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities”. In: *Theory of Probability & Its Applications* 16.2 (1971), pp. 264–280. doi: 10.1137/1116025.

[3] G. K. Dziugaite. “Revisiting Generalization for Deep Learning: PAC-Bayes, Flat Minima, and Generative Models”. PhD thesis. University of Cambridge, 2020.

[4] M. Belkin, D. Hsu, S. Ma, and S. Mandal. “Reconciling Modern Machine-Learning Practice and the Classical Bias–variance Trade-Off”. In: *Proceedings of the National Academy of Sciences* 116.32 (2019), pp. 15849–15854.

[5] A. Power, Y. Burda, H. Edwards, I. Babuschkin, and V. Misra. *Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets*. 2022.

[6] S. Lotfi, M. Finzi, S. Kapoor, A. Potapczynski, M. Goldblum, and A. G. Wilson. “PAC-Bayes Compression Bounds So Tight That They Can Explain Generalization”. In: *Advances in Neural Information Processing Systems (NIPS)*. 2022.

[7] K. Ahuja, E. Caballero, D. Zhang, J.-C. Gagnon-Audet, Y. Bengio, I. Mitliagkas, and I. Rish. “Invariance Principle Meets Information Bottleneck for Out-of-Distribution Generalization”. In: *Advances in Neural Information Processing Systems (NIPS)*. 2021.

[8] H. Ye, C. Xie, T. Cai, R. Li, Z. Li, and L. Wang. “Towards a Theoretical Framework of Out-of-Distribution Generalization”. In: *Advances in Neural Information Processing Systems (NIPS)*. 2021.

[9] D. Krueger, E. Caballero, J.-H. Jacobsen, A. Zhang, J. Binas, D. Zhang, R. L. Priol, and A. Courville. “Out-of-Distribution Generalization via Risk Extrapolation (REx)”. In: *International Conference on Machine Learning (ICML)*. 2021.

[10] Y. Xu and H. Zhang. *Convergence of Deep Convolutional Neural Networks*. 2022.

[11] T. Yoneda. *Pointwise Convergence Theorem of Gradient Descent in Sparse Deep Neural Network*. 2023.

[12] Y. Xu and H. Zhang. *Uniform Convergence of Deep Neural Networks With Lipschitz Continuous Activation Functions and Variable Widths*. 2023.

[13] V. Nagarajan and J. Z. Kolter. “Uniform Convergence May Be Unable to Explain Generalization in Deep Learning”. In: *Advances in Neural Information Processing Systems (NIPS)*. 2019.

[14] G. Cybenko. “Approximation by Superpositions of a Sigmoidal Function”. In: *Math. Control Signals Syst.* 2.4 (1989), pp. 303–314.

[15] K. Hornik. “Approximation Capabilities of Multilayer Feedforward Networks”. In: *Neural Netw.* 4.2 (1991), pp. 251–257.

[16] A. Pinkus. “Approximation Theory of the MLP Model in Neural Networks”. In: *Acta Numer.* 8 (1999), pp. 143–195.

[17] A. R. Barron. “Approximation and Estimation Bounds for Artificial Neural Networks”. In: *Machine Learning* 14.1 (1994), pp. 115–133.

[18] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang. “The Expressive Power of Neural Networks: A View From the Width”. In: *Advances in Neural Information Processing Systems (NIPS)*. 2017.

[19] S. M. Lundberg and S.-I. Lee. “A Unified Approach to Interpreting Model Predictions”. In: *Advances in Neural Information Processing Systems (NIPS)*. 2017.

[20] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. “Grad-Cam: Visual Explanations From Deep Networks via Gradient-Based Localization”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2017.

[21] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. “Learning Deep Features for Discriminative Localization”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[22] V. Petsiuk, A. Das, and K. Saenko. “RISE: Randomized Input Sampling for Explanation of Black-Box Models”. In: *Proceedings of the British Machine Vision Conference (BMVC)*. 2018.

[23] P. Schwab and W. Karlen. “CXPlain: Causal Explanations for Model Interpretation Under Uncertainty”. In: *Advances in Neural Information Processing Systems (NIPS)*. 2019.

[24] G. Pereyra, G. Tucker, J. Chorowski, Ł. Kaiser, and G. Hinton. “Regularizing Neural Networks by Penalizing Confident Output Distributions”. In: *arXiv preprint arXiv:1701.06548* (2017).

[25] D.-B. Wang, L. Feng, and M.-L. Zhang. “Rethinking Calibration of Deep Neural Networks: Do Not Be Afraid of Overconfidence”. In: *Advances in Neural Information Processing Systems (NIPS)*. 2021.

[26] D. Hawkins. *Identification of Outliers*. Chapman and Hall, 1980.

[27] Y. LeCun and C. Cortes. *MNIST handwritten digit database*. http://yann.lecun.com/exdb/mnist/. 2010.

[28] H. Xiao, K. Rasul, and R. Vollgraf. “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms”. In: *arXiv preprint arXiv:1708.07747* (2017).

[29] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252.

[30] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “ImageNet Classification With Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems (NIPS)*. 2012.

[31] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. *Intriguing properties of neural networks*. 2013. arXiv: 1312.6199 [cs.CV].

[32] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli. “Evasion Attacks Against Machine Learning at Test Time”. In: *Machine Learning and Knowledge Discovery in Databases*. 2013.

[33] I. J. Goodfellow, J. Shlens, and C. Szegedy. *Explaining and Harnessing Adversarial Examples*. 2014. arXiv: 1412.6572 [stat.ML].

[34] N. Carlini and D. Wagner. *Towards Evaluating the Robustness of Neural Networks*. 2016. arXiv: 1608.04644 [cs.CR].

[35] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. “The Limitations of Deep Learning in Adversarial Settings”. In: *IEEE European Symposium on Security and Privacy (EuroS&P)*. 2016.

[36] L. Muñoz-González, B. Biggio, A. Demontis, A. Paudice, V. Wongrassamee, E. C. Lupu, and F. Roli. “Towards Poisoning of Deep Learning Algorithms With Back-Gradient Optimization”. In: *ACM Workshop on Artificial Intelligence and Security*. 2017.

[37] A. E. Cinà, K. Grosse, A. Demontis, S. Vascon, W. Zellinger, B. A. Moser, A. Oprea, B. Biggio, M. Pelillo, and F. Roli. “Wild Patterns Reloaded: A Survey of Machine Learning Security against Training Data Poisoning”. In: *ACM Comput. Surv.* 55.13s (July 2023). issn: 0360-0300. doi: 10.1145/3585385.

[38] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg. “BadNets: Evaluating Backdooring Attacks on Deep Neural Networks”. In: *IEEE Access* 7 (2019), pp. 47230–47244.

[39] M. Barni, K. Kallas, and B. Tondi. “New Backdoor Attack in CNNs by Training Set Corruption Without Label Poisoning”. In: *IEEE International Conference on Image Processing (ICIP)*. 2019.

[40] Y. Liu, X. Ma, J. Bailey, and F. Lu. “Reflection Backdoor: A Natural Backdoor Attack on Deep Neural Networks”. In: *arXiv preprint arXiv:2007.02343* (2020).

[41] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein. “Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks”. In: *Advances in Neural Information Processing Systems (NIPS)*. 2018.

[42] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. *Practical Black-Box Attacks against Machine Learning*. 2016. arXiv: 1602.02697 [cs.CR].

[43] A. Demontis, M. Melis, M. Pintor, M. Jagielski, B. Biggio, A. Oprea, C. Nita-Rotaru, and F. Roli. “Why Do Adversarial Attacks Transfer? Explaining Transferability of Evasion and Poisoning Attacks”. In: *USENIX Security Symposium*. 2019.

[44] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart. “Stealing Machine Learning Models via Prediction APIs”. In: *USENIX Security Symposium*. 2016.

[45] S. J. Oh, M. Augustin, M. Fritz, and B. Schiele. “Towards Reverse-Engineering Black-Box Neural Networks”. In: *International Conference on Learning Representations (ICLR)*. 2018.

[46] D. Oliynyk, R. Mayer, and A. Rauber. “I Know What You Trained Last Summer: A Survey on Stealing Machine Learning Models and Defences”. In: *ACM Computing Surveys* 55.14s (2023), pp. 1–41.

[47] B. Biggio and F. Roli. “Wild Patterns: Ten Years After the Rise of Adversarial Machine Learning”. In: *Pattern Recognition* 84 (2018).

[48] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. “Towards Deep Learning Models Resistant to Adversarial Attacks”. In: *International Conference on Learning Representations (ICLR)*. 2018.

[49] D. Hendrycks, M. Mazeika, and T. Dietterich. “Deep anomaly detection with outlier exposure”. In: *arXiv preprint arXiv:1812.04606* (2018).

[50] P. Samangouei, M. Kabkab, and R. Chellappa. “Defense-Gan: Protecting Classifiers Against Adversarial Attacks Using Generative Models”. In: *International Conference on Learning Representations (ICLR)*. 2018.

[51] U. Hwang, J. Park, H. Jang, S. Yoon, and N. I. Cho. “PuVAE: A Variational Autoencoder to Purify Adversarial Examples”. In: *IEEE Access* 7 (2019), pp. 126582–126593.

[52] M. Glazunov and A. Zarras. “Do Bayesian Variational Autoencoders Know What They Don’t Know?” In: *Conference on Uncertainty in Artificial Intelligence (UAI)*. 2022.

[53] M. Glazunov and A. Zarras. “Vacant Holes for Unsupervised Detection of the Outliers in Compact Latent Representation”. In: *Uncertainty in Artificial Intelligence (UAI)*. 2023.

[54] H. Kwon. “Detecting Backdoor Attacks via Class Difference in Deep Neural Networks”. In: *IEEE Access* 8 (2020), pp. 191049–191056.

[55] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal. “Strip: A Defence Against Trojan Attacks on Deep Neural Networks”. In: *Annual Computer Security Applications Conference (ACSAC)*. 2019.

[56] Y. Liu, Y. Xie, and A. Srivastava. “Neural Trojans”. In: *IEEE International Conference on Computer Design (ICCD)*. 2017.

[57] A. K. Veldanda, K. Liu, B. Tan, P. Krishnamurthy, F. Khorrami, R. Karri, B. Dolan-Gavitt, and S. Garg. “NNoculation: Broad Spectrum and Targeted Treatment of Backdoored DNNs”. In: *arXiv preprint arXiv:2002.08313* (2020).

[58] X. Xu, Q. Wang, H. Li, N. Borisov, C. A. Gunter, and B. Li. “Detecting AI Trojans Using Meta Neural Analysis”. In: *IEEE Symposium on Security and Privacy (S&P)*. 2021.

[59] S. Kolouri, A. Saha, H. Pirsiavash, and H. Hoffmann. “Universal Litmus Patterns: Revealing Backdoor Attacks in CNNs”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020.

[60] C. Jin, M. Sun, and M. Rinard. “Incompatibility Clustering as a Defense Against Backdoor Poisoning Attacks”. In: *International Conference on Learning Representations (ICLR)*. 2023.

[61] L. Guiga and A. W. Roscoe. “Neural Network Security: Hiding CNN Parameters With Guided Grad-Cam”. In: *International Conference on Information Systems Security and Privacy (ICISSP)*. 2020.

[62] S. Kariyappa and M. K. Qureshi. “Defending Against Model Stealing Attacks With Adaptive Misinformation”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[63] H. Xu, Y. Su, Z. Zhao, Y. Zhou, M. R. Lyu, and I. King. *DeepObfuscation: Securing the Structure of Convolutional Neural Networks via Knowledge Distillation*. 2018.

[64] H. Chabanne, V. Despiegel, and L. Guiga. *A Protection Against the Extraction of Neural Network Models*. 2020.

[65] P. Maini, M. Yaghini, and N. Papernot. “Dataset Inference: Ownership Resolution in Machine Learning”. In: *International Conference on Learning Representations (ICLR)*. 2021.

[66] Y. Li, L. Zhu, X. Jia, Y. Jiang, S.-T. Xia, and X. Cao. “Defending Against Model Stealing via Verifying Embedded External Features”. In: *AAAI Conference on Artificial Intelligence*. 2022.

[67] N. Lukas, Y. Zhang, and F. Kerschbaum. “Deep Neural Network Fingerprinting by Conferrable Adversarial Examples”. In: *International Conference on Learning Representations (ICLR)*. 2021.

[68] H. Jia, C. A. Choquette-Choo, V. Chandrasekaran, and N. Papernot. “Entangled Watermarks as a Defense Against Model Extraction”. In: *USENIX Security Symposium*. 2021.

[69] H. Yu, H. Ma, K. Yang, Y. Zhao, and Y. Jin. “DeepEM: Deep Neural Networks Model Recovery Through EM Side-Channel Information Leakage”. In: *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. 2020.

[70] S. Pal, Y. Gupta, A. Kanade, and S. Shevade. *Stateful Detection of Model Extraction Attacks*. 2021.

3 EPISTEMIC UNCERTAINTY OF VARIATIONAL AUTOENCODERS

The problem of detecting the OoD inputs is of paramount importance for Deep Neural Networks (DNNs). It has been previously shown that even Deep Generative Modelings (DGMs) that allow estimating the density of the inputs may not be reliable and often tend to make over-confident predictions for OoDs, assigning to them a higher density than to the in-distribution data. This over-confidence in a single model can be potentially mitigated with Bayesian inference over the model parameters that take into account epistemic uncertainty. This paper investigates three approaches to Bayesian inference: Stochastic Gradient Hamiltonian Monte-Carlo (SGHMC), Bayes by Backpropagation (BBB), and Stochastic Weighted Averaging-Gaussian (SWAG). The inference is implemented over the weights of the deep neural networks that parameterize the likelihood of the Variational Autoencoder. We empirically evaluate the approaches against several benchmarks that are often used for OoD detection: estimation of the marginal likelihood utilizing sampled model ensemble, typicality test, disagreement score, and Watanabe–Akaike information criterion (WAIC). Finally, we introduce two simple scores that demonstrate the state-of-the-art performance.

This chapter is based on the paper "Do Bayesian Variational Autoencoders Know What They Don't Know?" by Glazunov, M., & Zarras, A. in Conference on Uncertainty in Artificial Intelligence (UAI), 2022 (pp. 718-727)[1].

3.1. INTRODUCTION

Deep Neural Networks (DNNs) are trained by Maximum Likelihood Estimation (MLE) over parameters $\boldsymbol{\theta}$ given the training input data $\mathcal{D}$: $p(\mathcal{D}|\boldsymbol{\theta})$. There exist two main approaches to modeling with DNNs: *discriminative* and *generative*.

The discriminative approach implies parameterizing a conditional distribution over target values y: $p(y|\mathbf{x}, \boldsymbol{\theta})$. The training of DNNs allows identifying optimum parameters $\boldsymbol{\theta}^*$ based on a stochastic first-order optimization algorithm. In the case of classification tasks, the common choice for $p(y|\mathbf{x}, \boldsymbol{\theta})$ is a categorical distribution, in the case of regression—a Gaussian distribution (quite often with a constant variance). As it has been recently discovered: such models tend to be over-confident in their predictions with Out-of-Distribution (OoD) inputs [2, 3]. This discovery may not be surprising since MLE results in a point estimate and does not account for epistemic uncertainty. Taking into consideration the fact that in modern DNNs $|\boldsymbol{\theta}| \gg |\mathcal{D}|$, there may be several models $\boldsymbol{\theta}^*$ that generated $\mathcal{D}$.

Epistemic uncertainty can be estimated by inferring a posterior distribution: $p(\boldsymbol{\theta}|\mathcal{D})$ which can be done within the *Bayesian* frame of reference. Several promising results were achieved with the discriminative DNNs for OoD detection utilizing *Bayesian* inference over model parameters [4–6].

On the other hand, the generative approach allows learning the approximation of a true distribution over the training data: $p(\mathbf{x})$. DNNs again do the parameterization of this density, hence the name: Deep Generative Modeling (DGM). Since DGMs provide a mechanism to estimate the probability of a particular input, they should supposedly assign a low density to the OoDs. However, recent research revealed that such estimations are prone to errors as DGMs often provide higher density values to OoDs than to in-distribution (ID) data [7].

As it was the case with the discriminative deep models, to overcome this problem, one may use a *Bayesian* DGM that infers the posterior: $p(\boldsymbol{\theta}|\mathcal{D})$ for the training data $\mathcal{D}$ over the model parameters $\boldsymbol{\theta}$. Such an approach allows getting an ensemble of the approximations of a true distribution of the data where each sample from the posterior $\boldsymbol{\theta} \sim p(\boldsymbol{\theta}|\mathcal{D})$ gives a separate instance of the model in the ensemble. Based on sampling from the posterior distribution, it is possible to estimate the density of the input instance $p(\mathbf{x})$ taking into consideration epistemic uncertainty.

In this work, we implement several methods that are widely applicable to the *Bayesian* inference over DNN parameters, namely: Bayes by Backpropagation (BBB) [4], Stochastic Gradient Hamiltonian Monte-Carlo (SGHMC) [5], and Stochastic Weighted Averaging-Gaussian (SWAG) [6]. Most of the methods till now have been only applied to the discriminative supervised DNNs. It should be noted that even though the theoretical justification for *Bayesian* Variational Autoencoders (VAEs) was already present in the original paper [8], there are still very few works address-

ing this point. In fact, to the best of our knowledge: there is only one paper dedicated to the Bayesian VAEs and OoD detection where only one of the methods (i.e., SGHMC) was used [9]. Our work represents an attempt to close this gap that is currently present between the discriminative and generative approaches based on DNNs. We transfer all of the mentioned methods to the deep generative VAEs and test them against several benchmarks suggested for OoD detection on various image datasets. Finally, based on our experiments, we introduce a couple of simple scores for the OoD detection that surpass all baseline scores.
In summary, we make the following main contributions:

- We perform the first implementation of three different *Bayesian* approaches for VAEs estimating epistemic uncertainty.
- We do a practical benchmarking of the most frequently used scores for OoD detection, taking into consideration *Bayesian* inference over the parameters of the likelihood of VAE.
- We suggest and apply two simple and efficient scores for the OoD detection that outperform baseline scores.
- We empirically evaluate the suggested approach based on several datasets.[1]

3.2. BACKGROUND

3.2.1. OUT-OF-DISTRIBUTION

Deploying a successful model requires the system to detect input data that are statistically anomalous or significantly different from those used during training. This is especially important for DNNs since they tend to produce overconfident predictions for such OoD inputs [10]. The lack of reliability of supervised discriminative models based on DNNs, when faced with OoD, was recently addressed by various methods [3, 11, 12].

Unsupervised DGMs such as autoregressive models [13], *Generative Adversarial Networks* (GANs) [14], flow-based models [15–17], and VAEs [8, 18] provide the opportunity to learn the density of the input data.

We choose to apply VAE as a particular instance of DGM in our experiments for several reasons:

1. It allows to obtain a particular value for the density in contrast to GANs that can only be sampled in a black-box manner.
2. It represents a model based on the latent variable, which seems like a reasonable assumption considering the complexity of the data's

[1]The source code for the reproducibility of the results is available at https://github.com/DigitalDigger/BayesianVAEsOoD

underlying density. The latent space has a much lower dimensionality in comparison with the dimensionality of the input. Such a bottleneck allows for learning the most relevant features. It distinguishes VAE from the flow-based models where all the transformations are invertible and represent a bijective mapping between the input and the latent space [15, 17, 19].

3. It allows the parametrization of all the *Bayesian* inference constituents, including the posterior over latent variable and the likelihood of the data [8, 18]. Such separation of the constituents makes it possible to work with the particular part, as in our case, with the decoder of the VAE for weight uncertainty estimation, distinguishing VAEs from the autoregressive models.

Furthermore, as it has been shown by [20], different DGM models may not produce similar results, which suggests that it is a good idea to concentrate only on one type for the analysis, which VAEs represent in our case.

However, it has been recently discovered that even in the case of DGMs that allows estimating the density, it does not work as intended and that DGMs return higher $p(\mathbf{x})$ for input data from a different distribution [7].

There are several approaches to tackle this issue. One possible solution is to enhance the training dataset. [11] suggested incorporating the pre-selected anomalous examples employing the so-called outlier exposure technique, which achieved promising results. [21] proposed a different method: two DGMs are trained separately—one for the semantics of the images and another for the background of the same images. The background images are generated with substantial noise to make the model learn this background, discarding the image's semantics. Then by calculating the likelihood ratios between the model that learned semantics and the model that learned the background, it is possible to detect OoD. However, both of the suggested approaches rely on the knowledge of either the outlier or the image's background, which cannot be reasonably covered for all the possible inputs. Moreover, recent research reveals that the likelihood ratio method does not achieve satisfactory results when *a Bayesian* VAE is applied [9]. Due to these reasons, we do not consider methods of dataset enhancement in this work.

Another possible solution is to devise an alternative score for the OoD detection. In that vein, the Watanabe–Akaike information criterion (WAIC) was successfully used by [22]. Further, the disagreement score [9] was suggested for the same purpose. This idea was motivated within the information-theoretic framework and was also based on the posterior estimation over the model parameters. The considered scores were calculated in both works based on the densities obtained from several models. In the former case, an ensemble was trained to calculate WAIC, while the latter case used the *Bayesian* VAE. In addition, [20] introduced a typicality test of the input sequences under the conjecture that the inlier sequences should be members of the DGM's typical set.

We chose to address the OoD problem similarly: i.e., we suggest and apply new simple scores that help to detect OoDs.

3.2.2. VARIATIONAL AUTOENCODER

VAE represents a type of DGM that provides the possibility of density estimation of the input $\mathbf{x}$. The optimization objective of VAE is the Evidence Lower Bound (ELBO), which allows joint optimization with respect to both variational parameters $\boldsymbol{\phi}$ of the encoder responsible for the variational approximation of the posterior $q_{\boldsymbol{\phi}}$ over the latent variable $\mathbf{z}$, and the generation parameters $\boldsymbol{\theta}$ of the decoder responsible for the parametrization of the likelihood of the input $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$:

$$\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x}) = E_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z}) - \log q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})] \tag{3.1}$$

VAEs are trained in an unsupervised manner from data and are widely used for generative purposes.

3.2.3. ESTIMATION OF THE MARGINAL LIKELIHOOD

Marginal likelihood can be computed in the following way:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \int_{\mathbf{z}} p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z} \tag{3.2}$$

However, it is difficult to calculate it precisely due to the integration over the whole $\mathbf{z}$-space. As suggested by [18], as soon as the VAE is trained, it is possible to estimate the marginal likelihood of the input under the generative model using *importance sampling* w.r.t to the approximated posterior, namely:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) \simeq \frac{1}{N}\sum_{i=1}^{N}\frac{p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z}_{(i)})}{q_{\boldsymbol{\phi}}(\mathbf{z}_{(i)}|\mathbf{x})}, \quad \text{where} \quad \mathbf{z}_{(i)} \sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \tag{3.3}$$

As it has been discovered by [7], we cannot rely directly on the marginal likelihood estimations produced by a single DGM. This fact is not surprising for the discriminative models based on DNNs. Therefore, it should not be shocking for DGMs either, taking into consideration that they are also based on the DNNs and that they also obtain the optimal parameters $\boldsymbol{\theta}^*$ under the Maximum Likelihood Estimation (MLE) for the $p(\mathcal{D}|\boldsymbol{\theta})$ which represents a point estimate. Hence, without *Bayesian* inference over model parameters, it is impossible to estimate the epistemic uncertainty, which results in the model's inability to provide a robust estimation of the marginal likelihood for OoD inputs.

3.2.4. EPISTEMIC UNCERTAINTY

The required posterior estimation over model parameters $p(\boldsymbol{\theta}|\mathcal{D})$ in the case of discriminative DNNs is usually implemented in the following three ways:

1. By using variational posterior approximation [4];
2. By sampling from Markov chain Monte Carlo (MCMC) [5];
3. By capturing the local geometry of the posterior through fitting a Gaussian with two first moments of the Stochastic Gradient Descent (SGD) [6].

The first method represents a similar variational inference approach as in the case of VAEs when their encoders are trained to infer the posterior over the latent variable. The difference is that now it is applied to minimize the KL-divergence between the intractable posterior *over the model parameters* $p(\boldsymbol{\theta}|\mathcal{D})$ and the distribution from the family of tractable distributions $q(\boldsymbol{\theta}|\mathcal{D})$. It is also implemented by maximizing the ELBO. Since in our work we implement it in VAE, it means that we are making the variational inference for both the posterior for the latent variable conditioned on the input and the posterior for the parameters conditioned on the training data. Such an approach is called fully *Bayesian* in the case of VAEs [8]. There are various methods for approximating the posterior for the model parameters; we will use the one suggested in [4].

MCMC constructs a Markov chain with the desired posterior as its equilibrium distribution. The most known method for MCMC is based on the Metropolis-Hastings algorithm [23, 24]. This algorithm converges to the real posterior by exploiting the random walk proposal distribution. However, this convergence may be pretty slow due to the slow exploration of the state space based on the random walk. Hamiltonian Monte Carlo (HMC) [25] postulates the exploration within the framework of the Hamiltonian dynamics. It allows producing distant proposals for the Metropolis algorithm resulting in much faster convergence. We use a variant of HMC adopted for deep learning, namely SGHMC, that relies on the noisy gradient estimates [5].

Finally, it is possible to infer the desired posterior while training due to the noise in the SGD [26]. We apply the Stochastic Weight Averaging (SWA) [27] together with fitting a Gaussian using the SWA solution as the first moment and covariance that is also derived from the SGD steps: the so-called SWAG method [6]. Stochastic Weighted Averaging-Gaussian (SWAG) is easy to implement since it does not require additional sampling and can be used as a baseline for the rest of the methods.

3.3. METHODOLOGY

3.3.1. BAYESIAN VAES

We implement several possible methods for *Bayesian* VAEs. We apply the *Bayesian* inference over the model parameters of the decoder of the VAEs. Such a method allows sampling of several decoders to form an ensemble with the subsequent marginal log-likelihood estimation as indicated in Equation 3.3.

Bayes by Backpropagation. We approximate the posterior distribution of the VAE decoder parameters given the training data $p(\boldsymbol{\theta}|\mathcal{D})$ based on the method suggested by [4]. This method was initially applied to discriminative learning. In our work, we implement it in VAEs. The ELBO objective is to find distribution parameters $\boldsymbol{\lambda}$ that minimize KL-divergence between our approximation and the true posterior; hence, the ELBO is formulated in the following way:

$$\mathcal{F}_{\boldsymbol{\theta}}(\mathcal{D},\boldsymbol{\lambda}) = \mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\lambda})}\Big[\log p(\mathcal{D}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) - \log q(\boldsymbol{\theta}|\boldsymbol{\lambda})\Big] \tag{3.4}$$

$\log p(\mathcal{D}|\boldsymbol{\theta})$ represents the sum of the marginal likelihoods of the individual inputs:

$$\log p(\mathcal{D}|\boldsymbol{\theta}) = \sum_{i=1}^{N} \log p(\mathbf{x}^{(i)}|\boldsymbol{\theta}) \tag{3.5}$$

and

$$\log p(\mathbf{x}^{(i)}|\boldsymbol{\theta}) \geq \mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x}^{(i)}) \tag{3.6}$$

where $\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x})$ is the ELBO for the marginal likelihood marginalized over the latent variable and it is defined in Equation 3.1. Since ELBO is the lower bound of the marginal likelihood, we can use it for our approximation. The optimization objective is formulated as:

$$\widetilde{\mathcal{F}_{\boldsymbol{\theta}}}(\mathcal{D},\boldsymbol{\lambda}) = \mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\lambda})}\Big[\sum_{i=1}^{N}\Big[\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x}^{(i)})\Big] + \log p(\boldsymbol{\theta}) - \log q(\boldsymbol{\theta}|\boldsymbol{\lambda})\Big] \tag{3.7}$$

Now, if we plug in the right-hand side of the objective in Equation 3.1 we can get the Monte-Carlo estimation of the combined variational objective:

$$\begin{aligned}\widehat{\mathcal{F}_{\boldsymbol{\theta},\boldsymbol{\phi}}}(\mathcal{D},\boldsymbol{\lambda}) \simeq \frac{1}{L}\sum_{j=1}^{L}\Big[\sum_{i=1}^{N}\Big[\log p_{\theta^{(j)}}(\mathbf{x}^{(i)},\mathbf{z}) - \log q_{\phi}(\mathbf{z}|\mathbf{x}^{(i)})\Big]\\ + \log p(\boldsymbol{\theta}^{(j)}) - \log q(\boldsymbol{\theta}^{(j)}|\boldsymbol{\lambda})\Big]\end{aligned} \tag{3.8}$$

where $\boldsymbol{\theta}^{(j)}$ is sampled from the posterior $q(\boldsymbol{\theta}|\boldsymbol{\lambda})$, $\boldsymbol{z}$ is sampled from the posterior $q(\mathbf{z}|\mathbf{x})$ and N is taken equal to the batch size.

For the minimization objective, we use the negated estimate: $-\widehat{\mathcal{F}_{\boldsymbol{\theta},\boldsymbol{\phi}}}(\mathcal{D},\boldsymbol{\lambda})$. We assume a diagonal Gaussian distribution for both variational posteriors with parameters μ and σ. In order to make σ be always non-negative we apply the same reparametrization as it was suggested by [4], namely $\sigma = \log(1+\exp(\rho))$, yielding the following posterior parameters $\lambda = (\mu,\rho)$. For the prior over the latent variable, we use the standard normal density, for the prior over the weights we use the scale mixture of two Gaussians as in [4].

The usual reparametrization trick [8] is applied to both $\boldsymbol{\theta}$ and $\boldsymbol{z}$ for training by backpropagation.

Stochastic Gradient Hamiltonian Monte Carlo. This approach exploits sampling instead of optimization, which was the case with BBB. This sampling is done within the MCMC framework and is based on the proposals generated utilizing the Hamiltonian dynamics. Namely, assume that the posterior distribution:

$$p(\boldsymbol{\theta}|\mathcal{D}) \propto \exp(-U(\boldsymbol{\theta},\mathcal{D})) \tag{3.9}$$

where $U(\boldsymbol{\theta},\mathcal{D})$ stands for the potential energy function in the Hamiltonian.

In our work, we take:

$$U(\boldsymbol{\theta},\mathcal{D}) = -\log p(\boldsymbol{\theta},\mathcal{D}) = -\sum_{i=1}^{N} \log p(\mathbf{x}^{(i)}|\boldsymbol{\theta}) - \log p(\boldsymbol{\theta}) \tag{3.10}$$

where $\log p(\mathbf{x}^{(i)}|\boldsymbol{\theta})$ is approximated by ELBO in our experiments and $\log p(\boldsymbol{\theta})$ is the prior over parameters.

Since HMC requires the computation of the gradient for the whole batch, a stochastic gradient alternative has been suggested by [5] which relies on the noisy gradients and allows proposal generation in a faster mini-batch manner.

In our work, we also apply the improvements suggested by [28] which significantly reduce the number of hyperparameters through adaptive estimates of the parameters in question during the burn-in procedure and subsequent training. Such an approach has been previously implemented in the unsupervised generative setting with VAEs by [9].

Stochastic Weighted Averaging-Gaussian. SWAG is fitting the following Gaussian distribution:

$$\mathcal{N}\left(\theta_{\text{SWA}}, \widehat{\Sigma_{\text{SWA}}}\right) \tag{3.11}$$

where θ_{SWA} is a running average over DNN parameters and Σ_{SWA} is the sample covariance matrix that after T epochs can be calculated as:

$$\theta_{\text{SWA}} = \frac{1}{T}\sum_{i=1}^{T}\theta_i \quad \text{and} \quad \Sigma_{\text{SWA}} = \frac{1}{T-1}\sum_{i=1}^{T}(\theta_i-\theta_{\text{SWA}})(\theta_i-\theta_{\text{SWA}})^{\mathsf{T}} \tag{3.12}$$

Since Σ_{SWA} is of a very high rank it is approximated by the K last epochs during training resulting in $\widehat{\Sigma_{\text{SWA}}}$[6].

SWAG was previously applied only to the discriminative DNNs, in our work we implement it within a generative approach with VAEs.

Combining several likelihoods. After the approximation of the variational posterior over the weights, the usual practice is to estimate the expected likelihood, the exact form of which can be formulated as follows:

$$p(\mathbf{x}^*|\mathcal{D}) = \int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta \tag{3.13}$$

The unbiased estimate of which can be obtained like this:

$$\mathbb{E}_{p(\theta|\mathcal{D})}[p(\mathbf{x}^*|\theta)] \simeq \frac{1}{N}\sum_{i=1}^{N}p(\mathbf{x}|\theta_i); \quad \text{where} \quad \theta \sim p(\theta|\mathcal{D}) \tag{3.14}$$

$p(\mathbf{x}|\theta_i)$ is computed by importance sampling as in Equation 3.3. As soon as the expected likelihood is estimated, one can apply a threshold that would distinguish if the considered input adheres to the in-distribution sample or not.

In [22] the likelihoods returned by several generative models are used to estimate the WAIC:

$$WAIC(\mathbf{x}^*|\mathcal{D}) = \mathbb{E}_{p(\theta|\mathcal{D})}[p(\mathbf{x}|\theta)] - Var_{p(\theta|\mathcal{D})}[p(\mathbf{x}|\theta)] \tag{3.15}$$

WAIC estimates the gap between the expected likelihood and the variance between the obtained likelihoods, which should benefit the small variance cases.

Another alternative is calculating the disagreement score $D[\cdot]$ suggested by [9]. This score measures the variation in the likelihoods $\{p(\mathbf{x}^*|\boldsymbol{\theta}_i)_{i=1}^N\}$ which captures the uncertainty of the models within the ensemble about the particular input:

$$D_\Theta[\mathbf{x}^*] = \frac{1}{\sum_{\theta\in\Theta} w_\theta^2}; \quad \text{where} \quad w_\theta = \frac{p(\mathbf{x}^*|\boldsymbol{\theta})}{\sum_{\theta\in\Theta} p(\mathbf{x}^*|\boldsymbol{\theta})} \tag{3.16}$$

The lower the score, the more informative the input is about the parameters $\boldsymbol{\theta}$, and consequently, the uncertainty value is higher.

The weights represent the normalized likelihoods between 0 and 1. The disagreement score sums up the squares of the weights and takes the reciprocal. If the score is large, it means that all models return close values of the likelihoods. On the contrary, if the score is 1, then there is one model that dominates.

Finally, [20] conjectured that due to the high dimensionality of inputs, the over-confidence of DGMs may be due to the fact that in-distribution images lie in the typical set in contrast to the tested OoDs that concentrate in the high-density region. Based on this conjecture, they introduced the test for typicality that treats all input sequences of length M as inliers if their entropy is sufficiently close to the entropy of the model, i.e., if the following holds for small ϵ, then the given M-sequence is in-distribution:

$$\left| \frac{1}{M} \sum_{m=1}^{M} -\log p\,(\mathbf{x}_m;\boldsymbol{\theta}) - \mathbb{H}[p(\mathbf{x};\boldsymbol{\theta})] \right| \leq \epsilon \tag{3.17}$$

We applied this score to one-element sequences since it is the most realistic scenario in practical applications of OoD detection.

Our scores. Based on the results of the experiments with the available metrics on all of the considered methods, we decided to apply our scores that capture the variation. In our work, we apply two simple scores for the same purpose. First, we measure the information entropy of the normalized likelihoods, namely:

$$\mathbb{H}_\Theta[\mathbf{x}^*] = -\sum_{\theta\in\Theta} w_\theta \log w_\theta; \quad \text{where} \quad w_\theta = \frac{p(\mathbf{x}^*|\boldsymbol{\theta})}{\sum_{\theta\in\Theta} p(\mathbf{x}^*|\boldsymbol{\theta})} \tag{3.18}$$

It is a standard information-theoretic metric that measures the average information of the distribution: the lower the entropy, the more one of the models is confident about the predicted value. The entropy measure is applied to the normalized likelihoods. Such normalization may be considered as the categorical distribution over the obtained marginal likelihoods of the models.

Secondly, we calculate the sample standard deviation of the marginal log-likelihoods returned by the models within the ensemble:

$$\Sigma_{\Theta}[\mathbf{x}^*] = \sqrt{\frac{1}{N-1}\sum_{\theta\in\Theta}(\log p(\mathbf{x}^*|\boldsymbol{\theta}) - \overline{\log p(\mathbf{x}^*|\boldsymbol{\theta})})^2} \tag{3.19}$$

It measures the variation within the log-likelihoods directly without normalizing step as in the case of the entropy, so if the variation persists along with the considered methods and datasets, the standard deviation will capture this difference: the higher the value, the more uncertainty there is between the models about a particular input.

Thresholding. There remains an open question of the appropriate threshold selection for the model evaluation. Since we are working in the unsupervised setting, intuitively, the ideal situation would be some threshold between the values of the scores returned by the model that successfully divides the inputs into OoDs vs. IDs. In order to validate the efficiency of the scores in achieving this task, we tackle this problem in the same way as [3] by using three different metrics: *Area Under the Receiver Operating Characteristic Curve* (AUROC), the *Area Under Precision-Recall* (AUPR) curve, and the *False-Positive Rate at 80% of True-Positive Rate* (FPR80). These metrics are threshold-independent because they compute the true positives and false positives for all possible thresholds providing a final single value of the efficiency of the used non-thresholded decision values in dividing them into two separate classes.

3.4. EVALUATION

We run all of our experiments on the four image datasets: MNIST [29], Fashion-MNIST [30], SVHN [31], and CIFAR-10 [32]. As it has been observed [7, 21], the likelihood estimations may be misled by the dataset they have been trained on. For instance, if the model has been trained on MNIST and OoD detection has been performed on the Fashion-MNIST, then the researcher, only by chance, may obtain good results. To avoid such mistakes, we train our models on all the datasets and check the following in-distribution vs. OoD: MNIST vs. Fashion-MNIST and vice versa, SVHN vs. CIFAR-10 and vice versa.

The following hardware infrastructure was used in all of our experiments: Xeon Platinum 8160 2.1 GHz 32 GB of RAM, 1 GPU NVIDIA Volta V100.

First, we estimate the impact of the latent space's number of dimensions on the loss function. The dimensionality is closely connected with the dataset the model is trained on. MNIST and FashionMNIST results

Table 3.1: Scoring values across all types of *Bayesian* VAEs trained on Fashion-MNIST data and tested on MNIST as OoD

	Fashion-MNIST vs. MNIST								
	BBB			**SGHMC**			**SWAG**		
	ROC AUC↑	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**
Expected LL	40.43	45.46	95.20	40.43	45.18	94.99	25.09	38.01	99.54
WAIC	59.53	59.35	71.88	55.79	53.86	74.56	19.90	35.14	99.83
Typicality test	40.51	43.40	86.36	41.02	43.85	86.05	56.40	50.32	64.88
Disagreement score	96.44	97.22	1.11	95.25	96.31	2.50	79.98	80.74	38.24
Entropy (ours)	97.97	98.43	**0.19**	97.28	97.92	**0.53**	**82.50**	**84.05**	**35.61**
Stds of LLs (ours)	**99.64**	**99.55**	0.34	**99.56**	**99.50**	0.55	19.90	34.22	94.42

Table 3.2: Scoring values across all types of *Bayesian* VAEs trained on CIFAR-10 data and tested on SVHN as OoD

	CIFAR-10 vs. SVHN								
	BBB			**SGHMC**			**SWAG**		
	ROC AUC↑	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**
Expected LL	59.73	53.27	58.99	60.39	53.74	58.08	60.31	53.51	57.05
WAIC	61.15	54.22	57.15	62.39	55.38	55.07	64.29	55.81	50.59
Typicality test	63.73	60.89	65.53	64.44	61.05	64.01	64.93	61.52	64.33
Disagreement score	81.16	84.82	38.47	80.41	83.00	40.61	73.27	76.95	54.95
Entropy (ours)	84.76	**88.21**	29.31	84.56	86.90	29.12	**76.51**	**80.54**	49.37
Stds of LLs (ours)	**89.98**	85.83	**16.03**	**92.52**	**91.48**	**12.27**	71.26	64.65	**44.34**

Table 3.3: Scoring values across all types of *Bayesian* VAEs trained on MNIST data and tested on Fashion-MNIST as OoD

	MNIST vs. Fashion-MNIST								
	BBB			**SGHMC**			**SWAG**		
	ROC AUC↑	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**
Expected LL	99.98	99.98	0.00	99.93	99.92	0.04	**96.83**	**96.20**	**5.18**
WAIC	99.99	99.99	0.00	**99.94**	**99.94**	0.02	80.37	76.25	33.56
Typicality test	99.98	99.98	0.00	99.88	99.90	0.00	94.91	96.47	1.58
Disagreement score	98.95	99.01	0.23	97.32	97.70	1.37	94.88	93.97	8.99
Entropy (ours)	99.42	99.47	0.02	98.50	98.75	0.29	95.72	95.20	8.37
Stds of LLs (ours)	**99.99**	**99.99**	**0.00***	99.91	99.91	**0.00***	80.37	82.78	39.12

* 0's are possible since it is a value for false-positive rate at 80% of true-positive rate

Table 3.4: Scoring values across all types of *Bayesian* VAEs trained on SVHN data and tested on CIFAR-10 as OoD

	SVHN vs. CIFAR-10								
	BBB			**SGHMC**			**SWAG**		
	ROC AUC↑	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**
Expected LL	58.65	61.79	77.72	57.09	60.56	80.18	58.98	62.06	76.52
WAIC	64.46	66.01	68.39	62.17	64.38	72.45	62.84	68.42	75.25
Typicality test	44.63	44.28	81.46	43.35	43.63	82.45	44.28	44.13	81.96
Disagreement score	85.20	88.35	30.26	85.31	88.52	28.66	77.58	80.36	45.60
Entropy (ours)	87.80	90.63	20.77	87.89	90.76	19.91	**80.01**	**83.24**	**41.58**
Stds of LLs (ours)	**93.29**	**91.51**	**10.99**	**94.70**	**93.95**	**8.67**	59.31	53.36	61.78

reveal no need to go over 10 latent dimensions since loss function did not significantly decrease after that value. For SVHN and CIFAR-10, we experimented with the number of latent dimensions up to 100; the most optimal results have been achieved with dimensionality equals 20 for

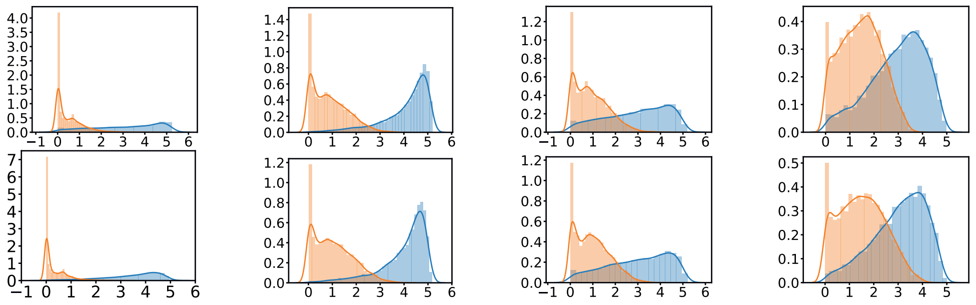


Figure 3.1: Histograms of the entropies of the marginal likelihoods. They are estimated based on sampling from the *Bayesian* VAEs, blue depicts in-distribution (ID) and orange - Out-of-Distribution (OoD). **From left to right**: MNIST as ID vs Fashion-MNIST as OoD, Fashion-MNIST as ID vs MNIST as OoD, SVHN as ID vs CIFAR-10 as Ood, CIFAR-10 as ID vs SVHN as OoD. **Top:** Sampling is done from Bayes-by-backprop VAE. **Bottom:** Sampling is done from SGHMC VAE.

SVHN and 70 for CIFAR-10.

We experimented with two different architectures for all our tests: one for the grayscale images and the second for the RGB images with 1 and 3 channels correspondingly. All models have been trained for 1000 epochs. To evaluate the inputs, we sampled 200 different models for our ensemble and evaluated them on a separate test data split of 5120 images that models have not been trained on. We used test splits of both ID and OoD datasets for all scores and metrics.

For our implementation of Bayes by backpropagation, we noticed that random normal initializer of the DNNs weights suggested as a prior in the original paper by [4] resulted in very slow convergence. To speed up the process, we also experimented with the following parameters: random normal initializer with 0 mean and 0.1 standard deviations for μ and constant initializer for $\rho = -3$, which improved the training speed [33].[2]

In case of SGHMC we adhere to the same protocol as in [9], namely, we use the same scale-adapted sampler implementation with learning rate 10^{-3} for the training and momentum decay 0.05.[3] We also place Gaussian priors over decoder parameters with precision $p(\theta) = \mathcal{N}(0, \lambda^{-1})$ and with Gamma hyperprior over the precision $p(\lambda) = \Gamma(\alpha, \beta)$ with $\alpha = \beta = 1$ that are resampled on each epoch.

For the experiments with SWAG, we set $K = 40$ and kept the default values for all the rest hyperparameters as in the original SWAG imple-

[2]For BBB we used the PyTorch *Bayesian* layers available at https://github.com/IntelLabs/bayesian-torch

[3]The SGHMC sampler that we used is available at https://github.com/automl/pybnn

mentation.[4]

All the experiments are done within the framework implemented by [19]. The results of the experiments against the benchmark scores for all datasets and models can be observed in the Tables 3.1 - 5.10.

3.5. DISCUSSION

As shown in the tables, *Bayesian* methods used in the discriminative approach can be successfully transferred to the generative models such as VAEs for OoD detection. Moreover, from the point of view of the scores: the variation among the model densities turns out to be persistent across all of the types of the *Bayesian* VAEs and all the datasets. The best results are achieved for BBB and SGHMC types of VAEs. The simple entropy score consistently demonstrates state-of-the-art results while detecting OoDs by better capturing the variation compared to the previously introduced baseline scores, the histograms for both BBB and SGHMC VAE results are shown in Figure 3.1. In addition, the sample standard deviation significantly outperforms the entropy score in the case of BBB and SGHMC methods.

Interestingly, almost all scores achieve comparably good results when trained on the MNIST dataset and tested on Fashion-MNIST (Table 5.9), but many of the baseline scores demonstrate substantially worse values when the experiments are conducted the other way around, i.e., trained on Fashion-MNIST and tested on MNIST (Table 3.1). The reason is that many of these scores are biased to a particular type of data. However, the bi-directional experiments easily identify these biases among all datasets and benchmark scores.

It can be observed that the worst-performing scores either intrinsically depend on the mean of the ensemble (such as WAIC) or on the log-likelihood itself returned by the model (such as a typicality score). In the first case, it results in the dominance of the variation of the particular values of the likelihoods for different inputs over the variation between the models within the ensemble for a single input, e.g., the range of variance of estimated likelihoods in the case of Fashion-MNIST is at least twice greater than in case of MNIST. In general, the more complex dataset is used for model training, the greater variance of the resulting likelihoods that the model assigns to the inputs; hence, there is potentially less influence of the variation between the models within the ensemble (that is completely lost in the case of WAIC for example). On the other hand, our scores measure the variance *within* the ensemble. It allows catching even a slight difference in such variation. In the second case with typicality, the log-likelihoods of inputs are used directly without any ensembling, which is susceptible to the well-known issue with modern deep

[4]SWAG sampler that we used is available at `https://github.com/wjmaddox/swa_gaussian`

generative models discovered by [7]. The same applies to the expected log-likelihood metric.

From the point of view of the speed performance, we consider the run-time required for the training convergence to get the same values of the likelihoods as for the vanilla VAEs. In such a case, the overhead of SWAG is almost negligible compared with the vanilla VAE. BBB and SGHMC, on the contrary, both take much longer time, with BBB requiring up to five times longer than the vanilla VAE training. SGHMC performs relatively faster than BBB but still lags far behind SWAG. There is also a clear tradeoff between the training performance and resulting accuracy in distinguishing between OoD vs. ID inputs: the fastest method in training (i.e., SWAG) results in the lowest OoD detection scores; however, the much slower training method (i.e., SGHMC) results in the best OoD detection scores for the most complex dataset that we experimented with (i.e., CIFAR-10). BBB turns out to be the slowest in training, and the results for OoDs are slightly worse than those obtained by SGHMC. Since SWAG has a minor overhead during training, it implies better scalability of this method to bigger datasets compared with SGHMC or BBB. If we also consider the speed performance from the point of view of the scalability to the bigger models, then it becomes clear that all of the methods rely on sampling the weights, so there is no clear winner, i.e., the more parameters a particular DNN would have the slower sampling would be for all of the methods.

It should be emphasized that these discrepancies in speed performance between the suggested *Bayesian* approaches can be seemingly treated as limitations in comparison with the vanilla VAEs and with the different ensemble techniques. First, let us consider the time required for the convergence of the model. As mentioned above, it may take five times longer in the worst case to obtain the values comparable to the ones of the vanilla VAE for both components of the ELBO loss. However, this disadvantage is disappearing, and the proposed *Bayesian* methods become even advantageous when one is training an ensemble of separate models $\{p(\mathbf{x}^*|\boldsymbol{\theta}_i)\}_{i=1}^N$ with $N > 5$ under the similar hardware constraints. Second, suppose we look at the resulting detection of OoD, which is slowed down with the several estimations of the marginal likelihood within the ensemble and subsequent calculation of the required score. In that case, this limitation primarily concerns the performance comparison with a single DGM without using ensembles. However, as we mentioned before, such a point estimate cannot be reliably used to estimate epistemic uncertainty. Therefore, if we compare with the traditional ensembling techniques, the main difference with the suggested approach stems from the way *how* the estimated marginal likelihoods are used. The former computes the expected likelihood, and the latter calculates the entropy or sample standard deviation of the marginal log-likelihoods, which means that we get $\mathcal{O}(N)$ computations in either of these scenarios.

3.6. CONCLUSION

The ability to detect OoD inputs by DGMs is of significant importance for robust inference, especially in practical applications. Our work concentrates on a specific type of such DGMs: Bayesian VAEs. We addressed this issue from three different perspectives:

1. ***Method-wise.*** We implemented three methods for estimating epistemic uncertainty in the generative setting based on VAEs utilizing *Bayesian* inference over model parameters: BBB based on variational inference, SGHMC based on Monte-Carlo sampling, and SWAG based on the noise in the SGD. Most methods have been previously applied exclusively to the discriminative models, and our paper bridges this gap between two modeling approaches.

2. ***Score-wise.*** We benchmarked all methods against the frequently used OoD benchmarks: expected log-likelihood, WAIC, disagreement score, and typicality test. Moreover, during our experiments, we noticed that the most promising score was based on the idea of the variation of marginal likelihoods. Built on that, we proposed using two simple scores: one is based on the information entropy, and the second is on the standard deviation for the robust unsupervised OoD detection. We achieved state-of-the-art results with them across all the benchmarked methods and considered datasets.

3. ***Experiment-wise.*** We did thorough experiments with all methods and scores on several datasets. Moreover, to avoid potential errors, we evaluated the results bi-directionally, e.g., if we trained a model on the MNIST dataset and used Fashion-MNIST as OoD, then we also trained a model on the Fashion-MNIST dataset and checked its ability to detect MNIST inputs as OoD. Such a check is necessary to avoid the bias of any particular scoring method to either more complicated or more simplified data.

The results of the experiments convincingly support the idea of the beneficial usage of the epistemic uncertainty estimation based on the variation for successful OoD detection in the case of VAEs. We observed that both BBB and SGHMC demonstrated comparable performance. While SWAG was always worse for the new OoD scores compared with other methods, we still conclude that it can be used as a simple baseline for epistemic uncertainty in the case of VAEs in the same manner as in the case of the discriminative approach. Moreover, from the point of view of the training convergence, SWAG turned out to be the fastest among the all considered *Bayesian* methods. Future work may revolve around a deeper understanding of the sources of variation within the ensemble from the point of view of the latent space of the VAE: e.g., is there a correlation between “holes” in the latent manifold and greater variance of the likelihoods.

3.7. SUPPLEMENTARY MATERIALS

3.7.1. SAMPLE STDS OF THE MARGINAL LOG-LIKELIHOODS

The sample standard deviations of the marginal log-likelihoods for BBB and SGHMC methods can be observed in Figure 3.2.

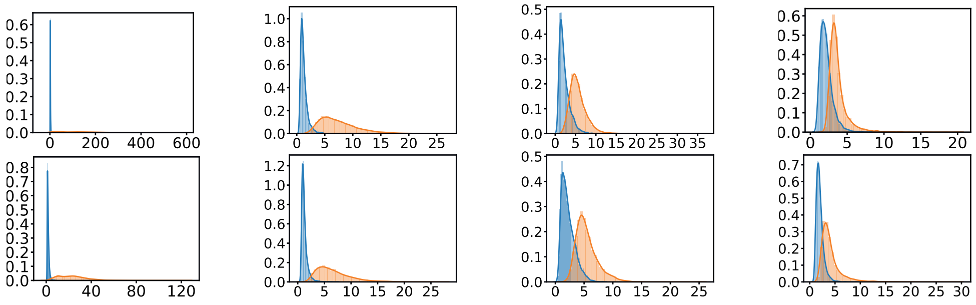


Figure 3.2: Histograms of the sample standard deviations of the marginal log-likelihoods, blue depicts in-distribution (ID) and orange - out-of-distribution (OoD). **From left to right**: MNIST as ID vs Fashion-MNIST as OoD, Fashion-MNIST as ID vs MNIST as OoD, SVHN as ID vs CIFAR-10 as Ood, CIFAR-10 as ID vs SVHN as OoD. **Top:** Sampling is done from Bayes-by-backprop VAE. **Bottom:** Sampling is done from SGHMC VAE.

3.7.2. VAE DISTRIBUTIONS

- For prior we used a standard multivariate Gaussian without parameters: $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I})$

- For variational distribution we used a multivariate factorized Gaussian with learned mean and variance: $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, diag(\boldsymbol{\sigma}^2))$

- For likelihood we used a multivariate factorized Bernoulli distribution:

$$p(\mathbf{x} \mid \mathbf{z}) = \prod_{j=1}^{D} p\left(x_j \mid \mathbf{z}\right) = \prod_{j=1}^{D} \text{Bernoulli}\left(x_j; p_j\right) \tag{3.20}$$

3.7.3. CNN ARCHITECTURES USED

For MNIST and FashionMNIST datasets with a single channel we used the following architectures.

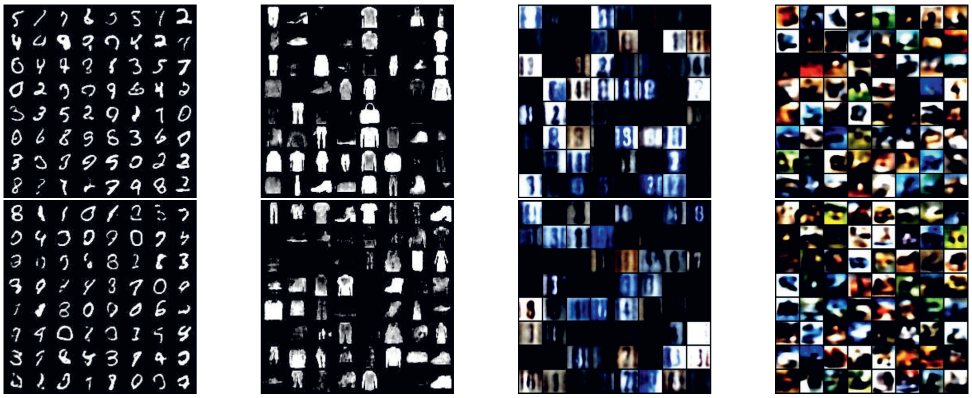


Figure 3.3: **From left to right**: MNIST, Fashion-MNIST, SVHN, CIFAR-10 **Top:** Random samples from BBB VAE. **Bottom:** Random samples from SGHMC VAE.

Table 3.5: Encoder CNN

Operation	**Kernel**	**Strides**	**Feature Maps**
Convolution	3 x 3	1 x 1	32
Convolution	3 x 3	1 x 1	16
Max pooling 2D	2 x 2	2 x 2	—
Linear for $\boldsymbol{\mu}$	—	—	10
Linear for log $\boldsymbol{\sigma}$	—	—	10

Table 3.6: Decoder CNN

Operation	**Kernel**	**Strides**	**Feature Maps**
Linear for sampled $\mathbf{z}$	—	—	2306
Upsampling nearest 2D	—	—	—
Max pooling 2D	2 x 2	2 x 2	—
Transposed Convolution	3 x 3	1 x 1	32
Transposed Convolution	3 x 3	1 x 1	1

For SVHN and CIFAR10 datasets with three channels we used the following architectures with additional padding = 1 and no bias for every convolutional layer. For SVHN latent dimensionality = 20, for CIFAR10 = 70.

Table 3.7: Encoder CNN

Operation	**Kernel**	**Strides**	**Feature Maps**
Convolution	3 x 3	1 x 1	16
Batch normalization	—	—	16
Convolution	3 x 3	2 x 2	32
Batch normalization	—	—	32
Convolution	3 x 3	1 x 1	32
Batch normalization	—	—	32
Convolution	3 x 3	2 x 2	16
Batch normalization	—	—	16
Linear	—	—	512
Batch normalization	—	—	512
Linear for $\boldsymbol{\mu}$	—	—	20 / 70
Linear for log $\boldsymbol{\sigma}$	—	—	20 / 70

Table 3.8: Decoder CNN

Operation	**Kernel**	**Strides**	**Feature Maps**
Linear for sampled $\mathbf{z}$	—	—	512
Batch normalization	—	—	512
Linear	—	—	1024
Batch normalization	—	—	1024
Transposed Convolution	3 x 3	2 x 2	32
Batch normalization	—	—	32
Transposed Convolution	3 x 3	1 x 1	32
Batch normalization	—	—	32
Transposed Convolution	3 x 3	2 x 2	16
Batch normalization	—	—	16
Transposed Convolution	3 x 3	1 x 1	3

For all architectures we used ReLU as a non-linearity. In addition, all pixels of the images have been normalized to [0,1] range for each channel for both training and testing phases.

3.7.4. RUNTIMES OF DIFFERENT METHODS

The runtimes for the training convergence of the different *Bayesian* methods are available in Table 3.9

Table 3.9: BVAE runtimes for learning

Method	**Time (mins)**
BBB	1628
SGHMC	1473
SWAG	371
Vanilla	345

3.7.5. SAMPLES FROM TRAINED MODELS

Random samples from all of the trained models for both BBB and SGHMC can be seen on Figure 3.3.

REFERENCES

[1] M. Glazunov and A. Zarras. “Do Bayesian Variational Autoencoders Know What They Don’t Know?” In: *Conference on Uncertainty in Artificial Intelligence (UAI)*. 2022.

[2] A. Nguyen, J. Yosinski, and J. Clune. “Deep neural networks are easily fooled: High confidence predictions for unrecognizable images”. In: *IEEE conference on computer vision and pattern recognition*. 2015, pp. 427–436.

[3] D. Hendrycks and K. Gimpel. “A baseline for detecting misclassified and out-of-distribution examples in neural networks”. In: *arXiv preprint arXiv:1610.02136* (2016).

[4] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. “Weight uncertainty in neural network”. In: *International conference on machine learning*. PMLR. 2015, pp. 1613–1622.

[5] T. Chen, E. Fox, and C. Guestrin. “Stochastic gradient hamiltonian monte carlo”. In: *International conference on machine learning*. PMLR. 2014, pp. 1683–1691.

[6] W. J. Maddox, P. Izmailov, T. Garipov, D. P. Vetrov, and A. G. Wilson. “A simple baseline for bayesian uncertainty in deep learning”. In: *Advances in Neural Information Processing Systems* 32 (2019).

[7] E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, and B. Lakshminarayanan. “Do deep generative models know what they don’t know?” In: *arXiv preprint arXiv:1810.09136* (2018).

[8] D. P. Kingma and M. Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

[9] E. Daxberger and J. M. Hernández-Lobato. “Bayesian variational autoencoders for unsupervised out-of-distribution detection”. In: *arXiv preprint arXiv:1912.05651* (2019).

[10] K. Lee, K. Lee, H. Lee, and J. Shin. “A simple unified framework for detecting out-of-distribution samples and adversarial attacks”. In: *Advances in neural information processing systems* 31 (2018).

[11] D. Hendrycks, M. Mazeika, and T. Dietterich. “Deep anomaly detection with outlier exposure”. In: *arXiv preprint arXiv:1812.04606* (2018).

[12] S. Liang, Y. Li, and R. Srikant. “Enhancing the reliability of out-of-distribution image detection in neural networks”. In: *arXiv preprint arXiv:1706.02690* (2017).

[13] A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, *et al.* “Conditional image generation with pixelcnn decoders”. In: *Advances in neural information processing systems (NIPS)*. 2016.

[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. “Generative adversarial nets”. In: *Advances in neural information processing systems* 27 (2014).

[15] L. Dinh, J. Sohl-Dickstein, and S. Bengio. “Density estimation using real nvp”. In: *arXiv preprint arXiv:1605.08803* (2016).

[16] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. “Improved variational inference with inverse autoregressive flow”. In: *Advances in neural information processing systems* 29 (2016).

[17] D. P. Kingma and P. Dhariwal. “Glow: Generative flow with invertible 1x1 convolutions”. In: *Advances in neural information processing systems* 31 (2018).

[18] D. J. Rezende, S. Mohamed, and D. Wierstra. “Stochastic backpropagation and approximate inference in deep generative models”. In: *International conference on machine learning*. PMLR. 2014, pp. 1278–1286.

[19] D. Nielsen, P. Jaini, E. Hoogeboom, O. Winther, and M. Welling. “Survae flows: Surjections to bridge the gap between vaes and flows”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 12685–12696.

[20] E. Nalisnick, A. Matsukawa, Y. W. Teh, and B. Lakshminarayanan. “Detecting out-of-distribution inputs to deep generative models using typicality”. In: *arXiv preprint arXiv:1906.02994* (2019).

[21] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. Depristo, J. Dillon, and B. Lakshminarayanan. “Likelihood ratios for out-of-distribution detection”. In: *Advances in neural information processing systems* 32 (2019).

[22] H. Choi, E. Jang, and A. A. Alemi. “Waic, but why? generative ensembles for robust anomaly detection”. In: *arXiv preprint arXiv:1810.01392* (2018).

[23] W. K. Hastings. “Monte Carlo sampling methods using Markov chains and their applications”. In: *Biometrika* 57.1 (1970), pp. 97–109.

[24] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. “Equation of State Calculations by Fast Computing Machines”. In: *The Journal of Chemical Physics* 21.6 (1953), pp. 1087–1092.

[25] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. “Hybrid monte carlo”. In: *Physics letters B* 195.2 (1987), pp. 216–222.

[26] S. Mandt, M. D. Hoffman, and D. M. Blei. “Stochastic gradient descent as approximate bayesian inference”. In: *arXiv preprint arXiv:1704.04289* (2017).

[27] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. G. Wilson. “Averaging weights leads to wider optima and better generalization”. In: *arXiv preprint arXiv:1803.05407* (2018).

[28] J. T. Springenberg, A. Klein, S. Falkner, and F. Hutter. “Bayesian optimization with robust Bayesian neural networks”. In: *Advances in neural information processing systems (NIPS)*. 2016.

[29] Y. LeCun and C. Cortes. *MNIST handwritten digit database*. http://yann.lecun.com/exdb/mnist/. 2010.

[30] H. Xiao, K. Rasul, and R. Vollgraf. “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms”. In: *arXiv preprint arXiv:1708.07747* (2017).

[31] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. “Reading Digits in Natural Images with Unsupervised Feature Learning”. In: *Advances in neural information processing systems (NIPS)*. 2011.

[32] A. Krizhevsky, V. Nair, and G. Hinton. *Cifar-10 (canadian institute for advanced research)*. http://www.cs.toronto.edu/kriz/cifar.html. 2010.

[33] R. Krishnan, P. Esposito, and M. Subedar. *Bayesian-Torch: Bayesian neural network layers for uncertainty estimation*. https://github.com/IntelLabs/bayesian-torch. 2020.

4 TOPOLOGY-AWARE APPROACH TO LATENT REPRESENTATION

Detection of the outliers is pivotal for any machine learning model deployed and operated in real-world. It is essential for the Deep Neural Networks (DNNs) that were shown to be overconfident with such inputs. Moreover, even deep generative models that allow estimation of the probability density of the input fail in achieving this task. In this work, we concentrate on the specific type of these models: Variational Autoencoders (VAEs). First, we unveil a significant theoretical flaw in the assumption of the classical VAE model. Second, we enforce an accommodating topological property to the image of the deep neural mapping to the latent space: compactness to alleviate the flaw and obtain the means to provably bound the image within the determined limits by squeezing both inliers and outliers together. We enforce compactness using two approaches: (i) *Alexandroff extension and* (ii) *fixed Lipschitz continuity constant on the mapping of the encoder of the VAEs. Finally and most importantly, we discover that the anomalous inputs predominantly tend to land on the vacant latent holes within the compact space, enabling their successful identification. For that reason, we introduce a specifically devised score for hole detection and evaluate the solution against several baseline benchmarks achieving promising results.*

This chapter is based on the paper "Vacant Holes for Unsupervised Detection of the Outliers in Compact Latent Representation" by Glazunov, M., & Zarras, A. in Conference on Uncertainty in Artificial Intelligence (UAI), 2023 (pp. 701-711)[1].

4.1. INTRODUCTION

Deep Generative Modelings (DGMs) allow for estimating the probability density of the input. This capability may appear tempting to utilize in the tasks of the detection of the outliers by casting all of the inputs that lie Out-of-Distribution (OoD) with the low density as anomalous. Nevertheless, empirical evidence shows that DGMs may sometimes be overconfident in their density estimation over OoDs [2]. Overconfidence is observed in all types of DGMs, including autoregressive models [3], normalizing flows [4], and VAEs [5, 6]. This fact may appear especially intriguing, considering the difference in the techniques used for density estimation among these three distinct modeling approaches. However, from the theoretical perspective, there is nothing peculiar in such performance. It can be easily demonstrated that it is possible to learn an invertible reparametrization of the actual density of the data in a way that assigns an arbitrary density to each point in the new representation even in the models with perfect densities and in a low-dimensional setting [7]. It means that the outlier detection is infeasible while relying only on the arbitrary learned probability density.

There are several alternative approaches aiming at tackling this issue that can be coarsely classified into one of the following categories: (*i*) methods that augment the input data by outliers [8, 9], (*ii*) ensemble-based methods [10–12], (*iii*) methods that introduce new scores [13, 14], (*iv*) methods based on the model modification [15, 16], (*v*) and methods that involve retraining of the models [17].

In this work, we refrain from augmenting data with outliers during training since it is not always feasible; we do not retrain the model to check every input as it is time-consuming, and due to the same reason, we do not apply ensemble-based methods. Instead, we utilize a model modification by introducing a new score. Specifically, we address the outlier detection from the perspective of general topology. Namely, we consider the property of compactness of the mapped image in the latent space. This property satisfies the necessary condition for the modeling assumption of a classical VAE from the viewpoint of the Universal Approximation Theorem (UAT) [18–20]. First, we implement compactification using the Alexandroff extension of a flat subspace to a hypersphere. Second, we utilize a related topological property: bounded continuity. It equips us with two additional valuable tools. In particular, it lets enforce the Lipschitz-continuity constraints on the mappings used in the model. These constraints, in turn, permit both to establish the compactness of the mapped image and simultaneously control its boundaries in the case of the flat latent space. In addition, it helps to identify if the continuity holes in the latent prior play a significant role in the outlier detection during the ablation study.

Constraining the mapped image of the encoder may at first sound counterintuitive since the common choice of a prior over the latent is used to be the standard normal distribution with the infinite support that explicitly implies that outliers should be placed in some different location, distinctly separated from the inliers. It includes the low-dimensional cases where such inputs are placed in the tails far from the mode and the high-dimensional cases where the outliers are located outside the typical set. However, as we already indicated, there is no guarantee for such behavior even in perfect density models since any density function can be manipulated by an arbitrary choice of representation. Since there is no control over the mapped compact in the latent space, the choice of the bounds of the learned factors of variations of the VAE is basically arbitrary. In some situations, it can be the case that the outliers are indeed placed far from the inliers, which gives an excellent separation based only on the density values; however, in other situations, the outliers and inliers may overlap, which in some cases results in the overconfidence of the model. Hence the purposeful control over the compactness of the mapped image enforces the model to bind the learned factors of variations for *any* input within the predefined limits. If these limits are chosen in such a way that enforces the model to squeeze all of its inputs in the properly bounded space, then the model would have no other choice than to map the outliers somewhere *within* the same space that is used for the inliers in the latent representation. Experimental evidence shows that when the model is confronted with such tight condensing, it tends to place the outliers into the vacant latent continuity holes allowing their successful detection.

In summary, we make the following main contributions:

- We reveal the persistent theoretical flaw in the modeling assumption of VAEs.
- We mitigate this shortcoming by enforcing controlled compactness of the latent space.
- By bounding the image of the encoder, we discover that the outliers tend to gravitate toward the vacant latent holes and devise an appropriate score for their detection.
- We empirically evaluate the suggested approach based on several datasets.

4.2. Background

4.2.1. Notation

We use nonbold x's to denote elements of general topological spaces, including the ones equipped with the appropriate metric. In the case of the normed vector spaces and random vectors within such spaces, we adhere to traditional usage in the literature, namely $\mathbf{x}$. When it comes to the particular elements comprising the random vector, we utilize x. The spaces are denoted as a pair $(\mathcal{X}, \mathcal{T})$ for topological spaces with the corresponding topology $\mathcal{T}$. In the specific case of metric spaces, we indicate the appropriate metric d that induces topology: $(\mathcal{X}, d_{\mathcal{X}})$.

4.2.2. VAES

VAE represents a DGM that allows to get an approximate value of the density of the input $\mathbf{x}$. It is based on the optimization of the Evidence Lower Bound (ELBO), that provides joint optimization w.r.t variational parameters $\boldsymbol{\phi}$ of the encoder responsible for variational approximation of the posterior $q_{\boldsymbol{\phi}}$ over the latent variable $\mathbf{z}$, and the generative parameters $\boldsymbol{\theta}$ of the decoder responsible for the parameterization of the likelihood $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$:

$$\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\Big[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) - \log \frac{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{z})}\Big] \tag{4.1}$$

This equation involves a data likelihood term (used for generative purposes) and a regularization term (the KL divergence between the variational family $q_{\phi}(\mathbf{z}|\mathbf{x})$ and the prior distribution over the latent variables).

The final estimation of the marginal likelihood is done using importance sampling. Backpropagation through the random variable $\mathbf{z}$ is performed utilizing the standard reparameterization trick [21].

4.2.3. COMPACTNESS

A topological space is **compact** or, equivalently, possesses a compactness property if every of its open cover has a finite subcover. In the case of the Eucledian spaces, the following specific result exists.

Theorem 4.2.1 (Heine-Borel Theorem). Let $K \subset \mathbb{R}^n$ then K is compact if and only if K is closed and bounded.

Compactification is the process of turning a topological space into a compact one.

Definition 4.2.2. Let $(\mathcal{X}, \mathcal{T})$ be a topological space and let $(\mathcal{X}^*, \mathcal{T}^*)$ be a compact topological space s.t. $\mathcal{X}$ is homeomorphic to a dense subspace of $\mathcal{X}^*$. Then $(\mathcal{X}^*, \mathcal{T}^*)$ is called a compactification of $(\mathcal{X}, \mathcal{T})$. Thus, a compact space $(\mathcal{X}^*, \mathcal{T}^*)$ is a compactification of a space $(\mathcal{X}, \mathcal{T})$ if and only if there exists a mapping f of $\mathcal{X}$ into $\mathcal{X}^*$ s.t. f is homeomorphism of $\mathcal{X}$ onto the subspace $f(\mathcal{X})$ of $\mathcal{X}^*$ and $f(\mathcal{X})$ is dense in $\mathcal{X}^*$.

An illustrative example of a frequently used compactification is an extension of $\mathbb{R}$ to $\mathbb{R} \cup \{-\infty, +\infty\}$.

Besides, there is a specific type of compactification by adjoining only one point: the Alexandroff extension.

Let $(\mathcal{X}, \mathcal{T})$ be a topological space and let ∞ be an object not belonging to $\mathcal{X}$. Let $\mathcal{X}^* = \mathcal{X} \cup \infty$ and let a topology $\mathcal{T}^*$ on $\mathcal{X}^*$ defined as follows: $\mathcal{T}^* = \mathcal{T} \cup \{V \subset \mathcal{X}^* : \infty \in V \text{and } \mathcal{X} \setminus V \text{ is closed and compact in } \mathcal{X}\}$. Then $(\mathcal{X}^*, \mathcal{T}^*)$ is the **Alexandroff extension** of $(\mathcal{X}, \mathcal{T})$.

An intuitive example of the Alexandroff extension is the inverse stereographic projection from the Euclidean plane to the sphere with the addition of a point at infinity.

4.2.4. Lipschitz continuity

A map $f : \mathcal{X} \to \mathcal{Y}$, where $(\mathcal{X}, d_\mathcal{X})$ and $(\mathcal{Y}, d_\mathcal{Y})$ are metric spaces with the corresponding metrics $d_\mathcal{X}$ and $d_\mathcal{Y}$, is called **Lipschitz continuous** if for any $x_1, x_2 \in \mathcal{X}$, there exists a constant $M \in \mathbb{R}^+$ such that:

$$d_\mathcal{Y}(f(x_1), f(x_2)) \leq M d_\mathcal{X}(x_1, x_2) \tag{4.2}$$

M is called a Lipschitz constant. In this work, we refer to the Lipschitz constant as the smallest possible M. A mapping with such a constant is called an M-Lipschitz map. If not explicitly indicated otherwise, we let $\mathcal{X} = \mathbb{R}^n$ and $\mathcal{Y} = \mathbb{R}^m$.

Recall that widely used activation functions such as sigmoid, tanh, and ReLU [22] are already generally scaled to be 1-Lipschitz. Hence, due to the composition property of the Lipschitz mappings, the first intuitive attempt to enforce the desirable Lipschitz property on the mapping would be to constrain the operator norm of the weights of each layer of the Deep Neural Network (DNN) [23, 24]. However, it was proven that such an approach could not approximate even a simple absolute value function [25]. To tackle the issue, [26] observed the critical component that influences the expressive power of any DNN, namely, the gradient-preserving property of its transformations. Therefore, they introduced the appropriate linear transformations and the 1-Lipschitz activation function, GroupSort, both of which are gradient preserving. They provably allow setting a Lipschitz constant on a DNN mapping. Moreover, DNNs utilizing them represent universal approximators of any Lipschitz mapping.

Latent holes

Continuity holes in the latent space can be detected based on the ratio of the distances between two nearly located points in the input space and the distances of their corresponding latent codes:

$$\mathcal{F}_{Lip} = d_\mathcal{Y}(f(x_1), f(x_2)) / d_\mathcal{X}(x_1, x_2) \tag{4.3}$$

Alternatively, there exist vacant regions of low density in the aggregated posterior where prior assigns a relatively high density. These regions can be detected via estimating the negative log-likelihood of the manipulated reference latent codes under the aggregated posterior:

$$\mathcal{F}_{Agg} = -\log p(\mathbf{z} \pm \epsilon) \tag{4.4}$$

where ϵ represents a magnitude of manipulation. Both of these scores are connected despite the different motivation meaning that if the hole is detected by the score $\mathcal{F}_{Agg}$ then it will be also detected by $\mathcal{F}_{Lip}$.

4.2.5. UNIVERSAL APPROXIMATION THEOREM

The theoretical underpinning of DNNs is rooted in the results obtained in the approximation theory that is commonly referred to as the universal approximation theorem [18–20].

Theorem 4.2.3 (**Universal Approximation Theorem**). *Let $C(\mathcal{X}, \mathcal{Y})$ denote the set of all continuous mappings from $\mathcal{X}$ to $\mathcal{Y}$. Let $\sigma \in C(\mathbb{R}, \mathbb{R})$ represent an element-wise activation function. Then let $\mathcal{N}^{\sigma}_{n,m}$ represent the class of feedforward neural networks with activation function σ, with n neurons in the input layer, m neurons in the output layer, and one hidden layer with an arbitrary number of neurons. Let $K \subseteq \mathbb{R}^n$ be* ***compact****. Then σ is nonpolynomial if and only if $\mathcal{N}^{\sigma}_{n,m}$ is dense in $C(K, \mathcal{Y})$.*

The activation functions currently used in DNNs are non-polynomial, so they fulfill the main requirement of the theorem. However, we deliberately emphasize that the results of the Universal Approximation Theorem (UAT) apply only in the cases when the input of the neural network is a compact set that is often overlooked.

4.3. RELATED WORK

New Scores-Based Methods. [13] conjecture that considering the high dimensionality of inputs, the over-confidence of DGMs may be because in-distribution images lie in the typical set as opposed to the tested OoDs that concentrate in the high-density region. They introduce the test for typicality that treats all input sequences as inliers if their entropy is close to the model's entropy.

Since the likelihood of generative models is biased by the complexity of the inputs, [14] propose to offset this bias by a factor that measures the input complexity and use the length of lossless compression of the image as the complexity factor, which is used to determine OoD. However, they do not evaluate their method on VAEs.

Ensemble-Based Methods. [12] use an ensemble of independently trained DGMs that allow to get the density value and score them against the WAIC.

Bayesian DGMs. Although the BDGMs represent a single model, the Bayesian inference over model parameters allows building ensembles on the fly. The theoretical justification for the Bayesian VAE has been first laid out by [5]. Several works are dedicated to OoD detection using Bayesian inference [10, 11]. They introduced new scores, such as the disagreement score and entropy. Both are based on the discrepancy between the models' estimations within the ensemble that achieved state-of-the-art results.

Lipschitz Continuity Methods. Several works utilize the Lipschitz continuity to improve the robustness of discriminative models against adversarial examples [27–29]. [30] apply the gradient-preserving transformations from [26] in a similar to our approach manner. However, their main focus is to use Lipschitz mappings for certifiable robustness against adversarial examples.

4.4. METHODOLOGY

4.4.1. COMPACTNESS OF THE LEARNED LATENT REPRESENTATION

A usual assumption for VAE models is that the prior follows the standard normal distribution: $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I})$. It is a meaningful choice from the perspective of the generative process since it provides a clear and simple way of sampling. Moreover, it is a natural candidate for the ELBO objective's regularization term in learning a Gaussian posterior per each input of VAE. However, it additionally implies an infinite support of the latent prior. We show that such an assumption contradicts the UAT (theorem 4.2.3).

Lemma 4.4.1. *Let $f : \mathcal{X} \to \mathcal{Y}$ be a continuous mapping from a topological space $\mathcal{X}$ to a topological space $\mathcal{Y}$. If $\mathcal{X}$ is compact, then its image $f[\mathcal{X}]$ is also compact (for proof, see Appendix A).*

Hence, by combining both **UAT** and **Lemma 1** it follows that the image of any DNN trained on a compact set is also compact. This conclusion contradicts the infinite support assumption of the standard normal prior in the case of VAEs. Any DNN used as an encoder will map all inputs to the compact subset of the latent space.

In the case of in-distribution inputs, this conclusion may be considered subtle since all such inputs should be assigned the appropriate density under the model learned during the DNN training. However, it plays a significant role as soon as the model starts dealing with the OoD inputs. These are the different inputs that the model has not seen before and has not been able to generalize during training. Therefore, as it was demonstrated in [7] the model is not constrained in putting those inputs anywhere within the whole available support or, more precisely, within the learned image of the encoder mapping. The properties of the compactness of the latent space become of great importance. One of the essential questions concerns the locations where the model tends to map the OoD inputs within the image compact space. As it was demonstrated by Nalisnick *et al.*, the DGMs and VAEs, in particular, tend to be overconfident with OoD inputs. There were several attempts to explain this type of behaviour [13, 31], but none addressed the issue of compactness.

In this paper, we deliberately enforce the latent space's compactness. The reason for that is twofold. First, it should alleviate the contradiction above in the modeling assumption of the VAE by providing a principled way to set the compactness of the image of the learned mapping. Furthermore, the input support for the decoder also gets a compact space during training which is again in line with UAT. Second, it allows us to conduct experiments with the outliers' detection in the controlled environment with the desirable compactness properties so that all the holes will be located within the predefined boundaries.

This approach can be implemented utilizing the following two separate methods:

- by Alexandroff extension
- by setting a predefined Lipschitz constant of the encoder

(*i*) by Alexandroff extension and (*ii*) by setting a predefined Lipschitz constant of the encoder. The first method implies a change of the intrinsic curvature of the latent space by switching from a Euclidean to a non-Euclidean manifold. On the other hand, the second method allows keeping a flat latent space by only enforcing specific bounds on a mapped compact.

4.4.2. COMPACTIFICATION OF THE LATENT SPACE

COMPACTIFICATION OF THE LATENT SPACE TO THE HYPERSPHERE

The Alexandroff extension of $\mathbb{R}^n$ can be done by adjoining a single point at infinity, turning the flat Euclidean space into a hypersphere $\mathcal{S}^n$ embedded into $\mathbb{R}^{n+1}$.

Lemma 4.4.2. *Let* $\mathcal{S}^n := \{\mathbf{x} \in \mathbb{R}^{n+1} : ||\mathbf{x}|| = 1\}$ *be a hypersphere with radius* $r = 1$ *centered at* $\mathbf{0}$ *and embedded in* $\mathbb{R}^{n+1}$ *then* $\mathcal{S}^n$ *is compact (for proof, see Appendix B).*

The appropriate type of distribution that can be utilized on the hyperspherical surface is the von Mises-Fisher distribution which is parameterized by mean μ and concentration κ. If the concentration parameter κ is greater than zero, then the distribution has properties similar to normal; however, when $\kappa = 0$, then it is a uniform distribution. It allows choosing the uniform prior and calculating the corresponding KL-divergence term for the regularization within the latent space. We utilize the same algorithm introduced by [32] for the sampling and reparametrization trick. They named it a Hyperspherical Variational Autoencoder (HVAE).

ENFORCING COMPACTNESS BY SETTING A LIPSCHITZ CONSTANT ON THE ENCODER MAPPING

Although the Alexandroff extension of the Euclidean space to the hypersphere is theoretically appealing, it has an issue with the surface area collapse, which makes it infeasible to use in high-dimensional settings. To alleviate these issues, we implement our own method of ensuring the compactness of the latent codes. This method is beneficial since it keeps the flat Euclidean space for the latent representation and provides the necessary means to control the boundaries of the resulting compact.

Theorem 4.4.3. *Image of an M-Lipschitz mapping $f : \mathcal{X} \to \mathcal{Z}$ from a compact $K \subseteq \mathcal{X}$ with $x, y \in K$: $\|f(x) - f(y)\| \leq M\|x - y\|$ is bounded by both a corresponding Lipschitz constant M* ***and*** *by a radius R of a closed ball in the input support.*

Proof. By the Heine-Borel theorem, a compact $K \subset \mathbb{R}^n$ is closed and bounded, meaning that the set is contained in some closed ball with a finite radius R. Hence, for any $x, y \in K$: $\|x - y\| \leq R$. Therefore, by combining the two inequalities above, we get $\|f(x) - f(y)\| \leq MR$, so the mapping f is bounded by the constant MR. □

Note that it is necessary to consider three components simultaneously to set a bound on the DNN output: bounds of the input compact, a norm being used, and, finally, an M-Lipschitz constant. In this work, we normalize the input support to the following compact vector space: $[0, 1]^n$. It conveniently allows constraining $R \leq 1$ by applying an L_∞-norm.

Moreover, to preserve both the generative functionality and the comparable log-likelihood values with the non-compact latent prior, it is important to consider the properties of the standard normal prior distribution. In the case of the low-dimensional setting, it is natural to bind the resulting compact with some standard deviation multiplier depending on the condensing tightness one wants to obtain. However, in the high-dimensional setting, the typical set should be considered. For that reason, the actual values for the Lipschitz constant of the encoder should be based on the dimensionality of the latent space. Namely, an upper bound on the mapped image should depend on the location of the typical set of the prior and its width. Recall that the center of the typical set of a centered normal distribution is located at the distance of $\sigma\sqrt{m}$ from the mode. In our experiments, we set the width equal to two standard deviations, and we choose the closest whole number:

$$M := \lfloor \sigma\sqrt{m} + 2\sigma \rceil \tag{4.5}$$

where m is the dimensionality of the latent representation and $\sigma = 1$ for the standard deviation of the prior.

In our work, we ensure the Lipschitz constant of the mapping utilizing the GroupSort activation function together with a projection of the weights of each layer on L_∞-ball during the forward-pass of the DNN. The constant is set layerwise in the following way: for a DNN with K number of layers in order to guarantee the M-lipschitz constant of the entire network mapping, we enforce the $M^{\frac{1}{K}}$ constant per each of its layer. It relies on the fact that the finite composition of Lipschitz functions is also Lipschitz with the product of the corresponding constants used in composition:

$$M = \prod_{n=1}^{K} M^{\frac{1}{K}} \tag{4.6}$$

The main building blocks are both 1-Lipschitz non-linearity and 1-Lipschitz linear mapping per each layer. The appropriate scaling of the results makes them equal to $M^{\frac{1}{K}}$. For the complete algorithm, see Appendix E.

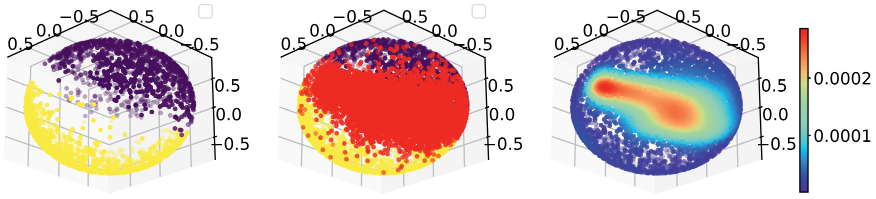


Figure 4.1: Compact S^2 latent space of VAE trained only on the two digits of the MNIST dataset 0's and 1's. The outliers densely land on the hole. **From left to right**: yellow depicts means of the estimated posteriors for 1's and purple for 0's; red represents the mapped means for a held-out outlier: a class of digits 9's; kernel density estimation of all means with the densest region in the hole packed with the outliers.

4.4.3. LATENT HOLES

We look at the holes from two different viewpoints as mentioned in section 4.2.4. First, we apply the following operational perspective to the definition of the hole: if two closely located latent points produce two distant samples in the input space, then we say that there is a hole in the latent space. This definition is similar to the one introduced by [33]. Second, from the conceptual perspective, we treat the holes as the regions where there is a discrepancy between the aggregated posterior and the prior [34], i.e., the hole appears when the regions with the high prior density have a low density of the aggregated posterior. Despite the seemingly different motivations for both definitions, it has been demonstrated by [35] that they are, in fact, connected. Moreover, if it is possible *to squeeze* all of the model's inputs within the high-density region of the prior, then the only "free" space within the latent compact turns out to be these holes.

4.4.4. WHY SQUEEZING?

The reason for that is at least two-fold. First, because of the arbitrariness of the mapping of outliers, it appears only logical to limit the whole image for any input (including outliers) within the same constrained space as for the inliers in order to eliminate this arbitrariness. The opposite approach, i.e., the widening of the compact, will not provide any benefits, only allowing for the model to use more "free" space where the outliers can be mapped to. Also, considering the well-known overconfidence issue [2], the wide compact does not guarantee the usage by the model of this available free space for any input. Some of the inputs can indeed be placed in the available space far from the mode; however, some will still be placed close to the mode (see Figure 4.2). Second, recall that VAEs, beside being probabilistic models, are also autoencoders. So they can be viewed from the perspective of the information bottleneck principle, i.e., when the information is put under pressure using the low-dimensional bottleneck layer to extract the relevant factors of variations of the input data in question. The compactness can be considered as a supplementary constraint to the low latent dimensionality (note that the dimensionality is also a topological property). By low dimensionality, we mean in comparison with the dimensionality of the input. Hence, by putting additional pressure in the form of a tight condensing of the mapped image within the predefined limits of the compact, we force the model to learn the bounded factors of variations for *any* input in a controlled and principled manner, eliminating the unnecessary "free" space for the model where it can potentially place outliers. The experimental evidence reveals that in such case the model indeed tends to place the outliers within the only available "free" space—the latent holes which, in turn, can be easily detected.

4.4.5. SCORES

As we indicated before, currently available scores for the holes' detection are based either on the availability of the suitable metric in the input space [33] or on the computationally expensive estimation of the aggregated posterior based on all the training samples [34]. The motivation for that was clearly because these scores are based on the inlier inputs; hence the search for the holes starts from their corresponding latent codes. However, in the case of outlier detection, we can directly check if the mapped input lands within the hole. For this purpose, we sample the approximated posterior $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ with several latent codes $\mathbf{z}$ under a particular input $\mathbf{x}$ and compute the sample standard deviation of the log-likelihoods $\log p(\mathbf{x}|\mathbf{z})$ (see Appendix G).

The approximated posterior under the input provides a locality within the latent space. Based on this locality—the samples from the posterior give us the notion of *nearness* around this specific locality. Finally, the standard deviation of the log-likelihoods based on the samples indicates *how far* from each other the sampled codes are mapped back into the input space. As a result, it becomes a beneficial indicator because it does not require making a particular traversal along some path (as was the case in [33]) or doing a thorough search through the latent space for all available holes (as was done in [35]). On the contrary, it allows direct checking if we are within the hole or not for a particular input.

There is also an alternative but still connected way of scoring the presence of a hole. Recall that density calculation of the given input under probability models with latent variables can be done through marginal likelihood. It is defined as the expected model likelihood marginalized over the latent's prior:

$$p(\mathbf{x}) = \mathbb{E}_{p(\mathbf{z})}[p(\mathbf{x}|\mathbf{z})] \tag{4.7}$$

First, let $\mathbf{z} \in \mathbb{L}$ and $|\mathbb{L}| < \infty$ then the marginal likelihood can be considered as a finite mixture of different $p(\mathbf{x}|\mathbf{z})$ with different constant weights $w = p(\mathbf{z})$ s.t. $\sum_{i=1}^{|\mathbb{L}|} w_i = 1$:

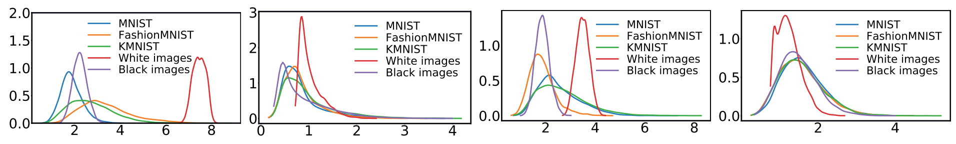
$$p(\mathbf{x}) = \sum_{i=1}^{|\mathbb{L}|} w_i p(\mathbf{x}|\mathbf{z}_i) \tag{4.8}$$

And $|\mathbb{L}|$ is the size of the components in the considered finite mixture of the likelihoods. Now suppose that $p(\mathbf{x}|\mathbf{z})$ is fully factorized, then the variance of the mixture of individual random components x's comprising $\mathbf{x}$ is given by:

$$\begin{aligned} \mathrm{Var}_{p(\mathbf{x})}(\mathrm{x}) = & \underbrace{\sum_{i=1}^{|\mathbb{L}|} w_i \mathrm{Var}_{p(\mathbf{x}|\mathbf{z}_i)}[\mathrm{x}]}_{\text{Weighed individual variances}} + \\ & + \underbrace{\sum_{i=1}^{|\mathbb{L}|} w_i \left(\mathbb{E}_{p(\mathbf{x}|\mathbf{z}_i)}[\mathrm{x}]\right)^2 - \left(\sum_{i=1}^{|\mathbb{L}|} w_i \mathbb{E}_{p(\mathbf{x}|\mathbf{z}_i)}[\mathrm{x}]\right)^2}_{\text{Jensen's gap}} \end{aligned} \tag{4.9}$$

The first term is a weighted sum of variances of individual model likelihoods under all latent codes. Note that the difference of second and third terms is always non-negative due to Jensen's inequality. This difference represents a *Jensens's gap* and can be interpreted as the variance of the means of the likelihoods weighted by the appropriate prior probabilities of the latent. Hence, by computing the variance of the marginal likelihood under importance sampling due to this Jensen's gap, it is possible to estimate the variance of the means of the likelihoods, which can be utilized for hole detection with outlier inputs. For that reason, we apply the sample standard deviation of the estimated marginal likelihoods under importance sampling (see Appendix G) and test the performance of this score in our thorough experiments. Since the marginal likelihood is already quite frequently estimated under importance sampling in many practical implementations, it becomes possible to quickly adapt these implementations for practitioners to incorporate the sample standard deviation of the marginal likelihood under importance sampling to get as a handy byproduct an alternative score for the hole identification. To distinguish between the two scores, we label the first as the hole indicator and the second as the Standard Deviation of Log-Likelihoods (Stds of LLs).

Threshold. For identifying the best threshold for the scores, we utilize threshold-independent metrics (these metrics are calculated for all possible thresholds) such as the *Area Under the Receiver Operating Characteristic Curve* (AUROC), the *Area Under Precision-Recall curve* (AUPR), and the *False-Positive Rate at 80% of True-Positive Rate* (FPR80) [36].



(a) Vanilla VAE (MNIST) (b) Lipshitz VAE (MNIST) (c) Vanilla VAE (FMNIST) (d) Lipschitz VAE (FMNIST)

Figure 4.2: Estimated Gaussian kernel densities of L_∞-norms of the approximated posterior means in the latent space for datapoints from MNIST as inliers, and datapoints from FashionMNIST, KMNIST, white and black images as outliers. **From left to right**: classical VAE trained on MNIST; VAE with a fixed Lipschitz constant $M = 1$ for encoder trained on MNIST; classical VAE trained on Fashion-MNIST; VAE with a fixed Lipschitz constant $M = 1$ for encoder trained on Fashion-MNIST.

4.5. EVALUATION

4.5.1. TOY EXPERIMENTS WITH COMPACT $\mathcal{S}^2$

We begin with the simple held-out experiments based on the MNIST dataset [37]. For that reason, we utilize HVAE. [1] It is trained with the hyperspherical uniform prior on $\mathcal{S}^2$ only on two digits as inliers, namely zeros and ones. The rest of the handwritten digits are considered outliers. These experiments assist in acquiring a fundamental intuition in the way how the encoder of the model maps the outliers in the compact latent space. As it can be observed in Figure 4.1, the two inlier classes are separated from each other on the sphere surface. There is also a hole between the clusters formed by these classes. Next, we try to map to the latent space held-out classes. As a result, we visually demonstrate that the encoder is forced to place the unseen during training classes somewhere within the constrained space and choose to land the outliers into latent holes. It happens when the model is confronted with the bounded factors of variation. In addition, we run experiments with our hole detection score $\Sigma_{\mathbf{z}}[\mathbf{x}]$ first with all held-out classes as outliers and second with all classes of Fashion-MNIST [38] as outliers. In addition, we conduct the experiments for 10 separate runs and summarize the results in a 99.9% confidence interval values that can be observed in Table 4.1. The obtained result strongly support our hypothesis about holes as centers of attraction for the outliers. Moreover, we compare these results with the corresponding baseline scores using Vanilla VAEs with the same low dimensionality of the latent space and also benchmark the hole indicator on the model trained on all classes of Fashion-MNIST vs. all MNIST classes as outliers (see Appendix F).

4.5.2. EXPLORING COMPACTNESS ENFORCED BY LIPSCHITZ CONTINUITY

We continue probing compactness properties based on the constrained Lipschitz mapping to the latent space. We run experiments with both the classical VAE models and the VAE models with the enforced Lipschitz constant $M = 1$ for the encoder. We trained four separate models (for the used DNN architectures, see Appendix D): on MNIST and Fashion-MNIST, with and without continuity constraints—the dimensionality of the latent space across all models: $m = 10$. We evaluate the means of the approximated posteriors for the outliers from KMNIST [39] (and analogously from Fashion-MNIST for the models trained on MNIST and vice versa). In addition, we run the same tests with the specially forged datasets. One contains non-realistic images, but all of their pixels tend to the black color; another contains images that tend to the white color. The idea

[1]The source code of the implemented solution is freely available at `https://github.com/DigitalDigger/VAEOutliersDetectionByVacantHoles`

Table 4.1: Hole indicator (means and 99.9% confidence interval values for 10 separate runs) for toy experiments with $\mathcal{S}^2$. The held-out outliers are all digits except 0's and 1's.

	MNIST held-out	**MNIST vs. Fashion-MNIST**
ROC AUC↑	89.05 (±0.25)	94.54 (±0.09)
AUPRC↑	99.38 (±0.02)	99.01 (±0.02)
FPR80↓	16.1 (±0.72)	5.60 (±0.2)

Table 4.2: Scoring values for the Lipschitz constrained VAEs trained on MNIST, Fashion-MNIST and CIFAR10

	MNIST vs. Fashion-MNIST			**Fashion-MNIST vs. MNIST**			**CIFAR10 vs. SVHN**		
	ROC AUC↑	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**
				Vanilla VAE					
Log likelihood	99.99	99.99	0.00	54.03	57.37	84.70	61.08	53.92	56.25
Input complexity	0.00	32.91	100.00	99.17	99.24	0.00	95.87	95.36	9.09
Typicality test	100.00	100.00	0.00	53.81	50.78	70.74	59.75	64.06	80.20
				Bayesian VAE					
WAIC	99.99	99.99	0.00	59.53	59.35	71.88	61.15	54.22	57.15
Disagreement score	98.95	99.01	0.23	96.44	97.22	1.11	81.16	84.82	38.47
Entropy	99.42	99.47	0.02	97.97	98.43	0.19	84.76	88.21	29.31
				Lipschitz VAE					
Stds of LLs	99.78	99.79	0.06	99.21	99.16	0.84	86.40	84.88	21.59
Hole indicator (ours)	**99.87**	**99.87**	**0.00**	**99.69**	**99.65**	**0.28**	**91.76**	**89.58**	**12.30**

The most robust scores are in bold. The highest values are in gray.

* 0's in FPR80 are possible since it is a value for false-positive rate at 80% of true-positive rate

behind the two latter datasets is that they represent extreme values of the compact support of the input data. As shown in Figure 4.2, the possible range of the values achievable by the classical VAE is considerably broad based on the limited number of the outlier datasets. For the model trained on MNIST, it goes as far as seven standard deviations from the mean.

Meanwhile, the unconstrained model trained on Fashion-MNIST has a range with a maximum of around four standard deviations. It demonstrates the arbitrariness of the mapped compact and its limits. Note, however, that when we bound the continuity of the encoder, then both inliers and outliers are squeezed together in a compact within the appropriate limits, which experimentally confirms our theoretical result (see theorem 4.4.3). It allows the enforcement of a controlled and bounded compactness on the flat prior.

4.5.3. DETECTING OUTLIERS

As we indicated before, due to the surface collapse of the sphere, it is infeasible to use HVAE with high-dimensional priors. Hence, we apply the fixed Lipschitz mapping together with the appropriate input normalization (all inputs are normalized to $[0,1]^n$). We evaluate our approach against several baseline methods. For them, we choose the classical VAE, the ensemble-based VAE, namely, the one based on the Bayesian inference over the weights of the DNN, and several approaches based on the new scores, namely, typicality score and input complexity. For scoring the Bayesian VAE, we utilize three available scores: WAIC, a disagreement score, and entropy. Bayesian inference is implemented utilizing the Bayes by Backpropagation (BBB) [40]. The corresponding hyperparameters and the training protocol are based on the work by [11]. All models trained on MNIST and Fashion-MNIST have the dimensionality of the latent space equal to 10, and models trained on the CIFAR10 dataset have the latent of 70 dimensions. For our suggested Lipschitz-based model, we compute the appropriate Lipschitz constant for the decoder based on the dimensionality of the latent space in order to preserve the comparable log-likelihood values of the classical VAE and also to be able to sample the prior in a standard way. For MNIST and Fashion-MNIST, it is equal to 5, and for CIFAR10, it is equal to 10. The results can be observed in Table 5.11. Our hole indicator demonstrates the best results among the scores that consistently perform well across all datasets. Moreover, the standard deviation of the likelihoods is the second most robust score, which agrees with our theoretical derivation (see Equation 4.9). By robustness in this context, we mean the persistence of the state-of-the-art results, independent of the dataset used for training the model and testing for the outliers. For example, despite the high values for the typicality test on MNIST vs. Fashion-MNIST datasets and input complexity on CIFAR10 vs. SVHN datasets, they are inconsistent across all of the considered datasets, making them unreliable in practical applications.

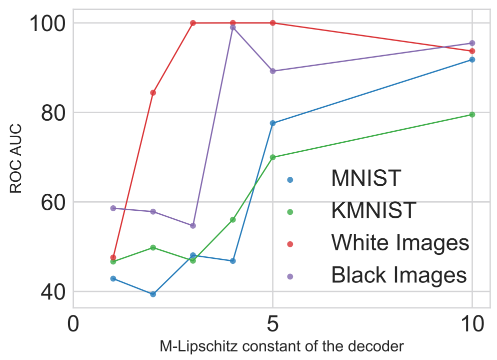


Figure 4.3: ROC AUC values from the ablation study. VAE with different Lipschitz constants enforced on the decoder, namely, $M = 1, M = 2, M = 3, M = 4, M = 5$ and $M = 10$, all plotted along x-axis. VAE is trained on Fashion-MNIST with the Encoder Lipschitz constant $M = 1$ for all tests and evaluated on several outlier datasets.

The reason behind our score’s robustness is that the model maps the outliers to the holes in the compact latent space (i.e., the only “free” space available for the learned factors of variations) that can be easily detected. Other scores rely either on the complexity of the dataset (as input complexity score), which is a data-dependent score, or on the hypothesis about the typical set, which is not always guaranteed because of the arbitrariness of the mapping of the encoder to any available “free” space including the holes in the typical set.

4.5.4. ABLATION STUDY

To check if the continuity holes are responsible for the obtained results, we conduct experiments with the gradual reduction of the holes in the latent space. This can be done by smoothing out the decoder mapping. This approach is advantageous since it affects all holes in the latent space. Hence, if our assumption is correct, then the results of the outlier detection based on the holes should degrade according to the strength of the smoothing. We enforce smoothing by setting the corresponding Lipschitz constants on the decoder mapping in the same way as it was done for the encoder in previous experiments. We train six separate models, all of which have the Lipschitz encoder with $M = 1$. Decoder is enforced with the values of the Lipschitz constants M from the following set: $\{1, 2, 3, 4, 5, 10\}$. As can be seen in Figure 4.3, there is an apparent performance degradation of the hole indicator for the outliers with decreasing of the corresponding Lipschitz constant enforced on the decoder, which is in line with our hypothesis that outliers land on the latent holes. Finally, we separately ablated the compactness component (for the results see Appendix H).

4.6. CONCLUSION

In this paper, we identified an implicit theoretical inconsistency from the perspective of general topology between the VAE modeling and the UAT. We addressed this discrepancy utilizing the compactness of the mapped image to the latent space. In order to enforce the compactness, we devised a provable method for controlling the bounds of the resulting compact. The experimental evidence revealed that constraining the limits of the factors of variation is beneficial for outlier detection. In particular, we discovered that outlier inputs tend to be mapped to the latent continuity holes. By devising a special score for the hole indicator, we conducted several experiments aimed at their detection. Utilizing this score, we achieved promising results in unsupervised outlier detection based on the latent representation. Specifically, the suggested method and score demonstrated the most robust performance across all the used benchmarks and datasets.

4.7. SUPPLEMENTARY MATERIAL

4.7.1. PRESERVATION OF COMPACTNESS UNDER CONTINUOUS MAPPING

Lemma 4.7.1. *Let $f : \mathcal{X} \to \mathcal{Y}$ be a continuous mapping from a topological space $\mathcal{X}$ to a topological space $\mathcal{Y}$. If $\mathcal{X}$ is compact then its image $f[\mathcal{X}]$ is also compact.*

Proof. [2] Let $\mathcal{C} = \{U_i\}_{i \in I}$ be any open covering of $f[\mathcal{X}]$ in $\mathcal{Y}$. Then: $f[\mathcal{X}] \subseteq \bigcup_{i \in I} U_i$

Now let's take the inverse of both its sizes:

$$\mathcal{X} \subseteq f^{-1}\left(\bigcup_{i \in I} U_i\right) \tag{4.10}$$

$$\mathcal{X} \subseteq \bigcup_{i \in I} f^{-1}(U_i) \tag{4.11}$$

Since f is continuous and U_i is open in $\mathcal{Y}$ for all $i \in I$ we have that $f^{-1}(U_i)$ is open in $\mathcal{X}$ for all $i \in I$. From above, we see that then $\{f^{-1}(U_i)\}_{i \in I}$ is an open cover of $\mathcal{X}$. Since $\mathcal{X}$ is compact, this open cover has a finite subcover, say $\{f^{-1}(U_{i_1}), f^{-1}(U_{i_2}), \ldots, f^{-1}(U_{i_n})\}$ where $i_n \in I$ where:

$$\mathcal{X} \subseteq \bigcup_{k=1}^{n} f^{-1}(U_{i_k}) \tag{4.12}$$

Taking the image of both sides above and we have that:

$$f[\mathcal{X}] \subseteq f\left(\bigcup_{k=1}^{n} f^{-1}(U_{i_k})\right) \tag{4.13}$$

$$f[\mathcal{X}] \subseteq \bigcup_{k=1}^{n} f(f^{-1}(U_{i_k})) \tag{4.14}$$

$$f[\mathcal{X}] \subseteq \bigcup_{k=1}^{n} U_{i_k} \tag{4.15}$$

Thus $\mathcal{C}* = \{U_{i_1}, U_{i_2}, \ldots, U_{i_n}\}$ is a finite subcover of $\mathcal{C}$. Hence $f[\mathcal{X}]$ is compact in $\mathcal{Y}$. □

4.7.2. SPHERE IS COMPACT

Lemma 4.7.2. *Let $\mathcal{S}^n := \{\mathbf{x} \in \mathbb{R}^{n+1} : ||\mathbf{x}|| = 1\}$ be a hypersphere with radius $r = 1$ centered at $\mathbf{0}$ and embedded in $\mathbb{R}^{n+1}$ then $\mathcal{S}^n$ is compact.*

[2]Adapted from: http://mathonline.wikidot.com/preservation-of-compactness-under-continuous-maps

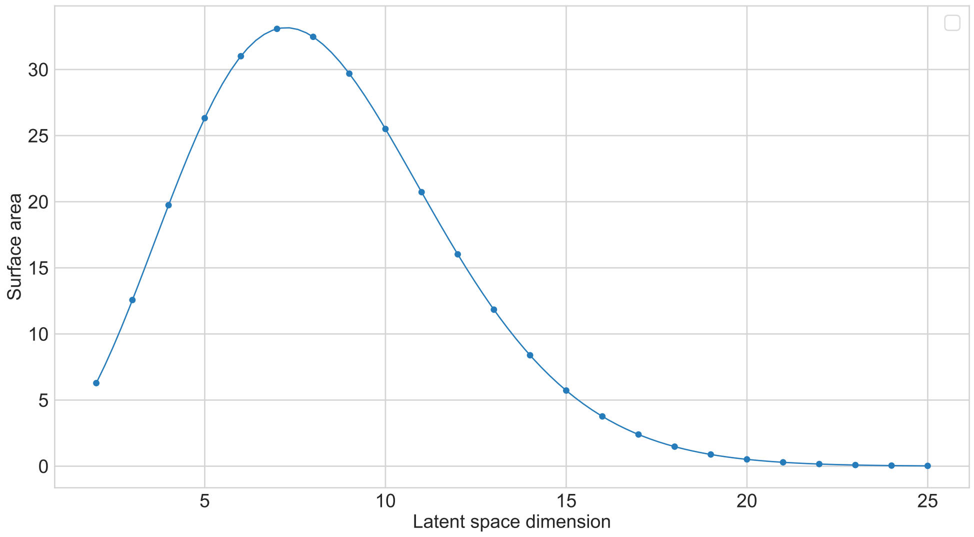


Figure 4.4: The problem of surface area collapse.

Proof. First note that $\mathcal{S}^n$ is obviously bounded. Next, observe that $||\mathbf{x}|| = \sum x^2$ which represents a continuous mapping whose inverse is a closed set: $\{1\}$, therefore the $\mathcal{S}^n$ is closed. It follows that the $\mathcal{S}^n$ is both closed and bounded, hence by Heine-Borel theorem it is compact. $\square$

4.7.3. SURFACE AREA COLLAPSE OF THE SPHERE

As can be observed from the Figure 4.4 the surface area grows up to approximately seven dimensions and after that it goes down completely collapsing in cases with greater than twenty dimensions. This issue makes it infeasible to use compact hyperspherical latent space in high-dimensional configurations.

4.7.4. DNN ARCHITECTURES USED

For MNIST and FashionMNIST datasets with a single channel we used the following architectures for baseline experiments.

For CIFAR10 dataset with three channels we used the following architectures with additional padding = 1 and no bias for every convolutional layer. Latent dimensionality = 70.

For all architectures, we used ReLU as a non-linearity in the case of classical VAE. For Lipschitz encoder we used GroupSort. In addition, all pixels of the images have been normalized to the [0,1] range for each channel for both the training and testing phases. For HVAE, we used

Table 4.3: Encoder CNN for MNIST and FashionMNIST

Operation	Kernel	Strides	Feature Maps
Convolution	3 x 3	1 x 1	32
Convolution	3 x 3	1 x 1	16
Max pooling 2D	2 x 2	2 x 2	—
Linear for $\boldsymbol{\mu}$	—	—	10
Linear for $\log \boldsymbol{\sigma}$	—	—	10

Table 4.4: Decoder CNN for MNIST and FashionMNIST

Operation	Kernel	Strides	Feature Maps
Linear for sampled $\boldsymbol{z}$	—	—	2306
Upsampling nearest 2D	—	—	—
Max pooling 2D	2 x 2	2 x 2	—
Transposed Convolution	3 x 3	1 x 1	32
Transposed Convolution	3 x 3	1 x 1	1

Table 4.5: Encoder CNN for SVHN and CIFAR10

Operation	Kernel	Strides	Feature Maps
Convolution	3 x 3	1 x 1	16
Convolution	3 x 3	2 x 2	32
Convolution	3 x 3	1 x 1	32
Convolution	3 x 3	2 x 2	16
Linear	—	—	512
Linear for $\boldsymbol{\mu}$	—	—	70
Linear for $\log \boldsymbol{\sigma}$	—	—	70

Table 4.6: Decoder CNN for SVNH and CIFAR10

Operation	Kernel	Strides	Feature Maps
Linear for sampled $\boldsymbol{z}$	—	—	512
Linear	—	—	1024
Transposed Convolution	3 x 3	2 x 2	32
Transposed Convolution	3 x 3	1 x 1	32
Transposed Convolution	3 x 3	2 x 2	16
Transposed Convolution	3 x 3	1 x 1	3

the same architectures as in the original implementation [3], i.e., two hidden linear layers for the encoder with the dimensionality 256 and 128 correspondingly, and two hidden linear layers for the decoder with di-

[3]We used the official implementation available at https://github.com/nicola-decao/s-vae-pytorch

mensionality 128 and 256. For Lipschitz VAE we also used two hidden linear layers for both encoder and decoder with doubled dimensionality for each corresponding hidden layer.

4.7.5. FORWARD PASS OF THE LIPSCHITZ CONSTANT ENFORCING

Algorithm 1: Ensuring Lipschitz constant in a DNN mapping

```
LInfBallProjection
    Input    : $\mathbf{y} \in \mathbb{R}^N$
    Output   : $\mathbf{x} \in \mathbb{R}^N$
    Sort y into u: $u_1 \geq \ldots \geq u_N$
    Set $K := \max_{1 \leq k \leq N} \{k | (\sum_{r=1}^{k} u_r - 1)/k < u_k\}$
    Set $\tau := (\sum_{k}^{K} u_k - 1)/K$
    for $n = 1, \ldots, N$ do
        Set $x_n := \max_{y_n - \tau, 0}$
Input    : Data point x
Result   : Network output $\mathbf{h}_L$
Requires: Lipschitz constant $M$
Forward pass
    $\mathbf{h}_0 \leftarrow \mathbf{x}$
    for $l = 1, \ldots, L$ do
        $\mathbf{W}_l \leftarrow$ LInfBallProjection($\mathbf{W}_l$)
        pre-activation $\leftarrow M^{\frac{1}{L}} \mathbf{W}_l \mathbf{h}_{l-1}$
        $\mathbf{h}_l \leftarrow$ GroupSort(pre-activation)
```

4.7.6. FURTHER EXPERIMENTS WITH HYPERSPHERICAL VAE

As can be observed in Table 4.7 the most robust scores are hole indicators that achieve the most consistent results across all used datasets.

4.7.7. SCORES

STDS OF LLS

Recall that *importance sampling* is used to estimate the marginal likelihood of the input under the trained VAE, namely:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) \simeq \frac{1}{N} \sum_{i=1}^{N} \frac{p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}_{(i)})}{q_{\boldsymbol{\phi}}(\mathbf{z}_{(i)}|\mathbf{x})}, \quad \text{where} \quad \mathbf{z}_{(i)} \sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \tag{4.16}$$

where $\boldsymbol{\phi}$ represents the variational parameters of the encoder responsible for the variational approximation of the posterior $q_{\boldsymbol{\phi}}$ over the latent

Table 4.7: Scoring values (means and 99.9% confidence interval) for toy experiments with $\mathcal{S}^2$ for MNIST vs. held-out and Fashion-MNIST. The held-out outliers are all digits except 0's and 1's. And with $\mathcal{S}^3$ for Fashion-MNIST vs. MNIST. Note that Vanilla VAEs in the experiments are equipped with the same low dimensional latent space as the surface of the corresponding HVAE.

	MNIST held-out			MNIST vs. Fashion-MNIST			Fashion-MNIST vs. MNIST		
	ROC AUC↑	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**
				Vanilla VAE					
Log likelihood	96.84 (±0.07)	98.50 (±0.04)	4.43 (±0.27)	99.85 (±0.02)	99.86 (±0.01)	0.00 (±0)	45.13 (±0.1)	43.75 (±0.05)	75.60 (±0.27)
Input complexity	42.98 (±0.86)	45.28 (±0.52)	81.82 (±0)	18.27 (±2.12)	37.18 (±0.8)	100 (±0)	94.96 (±1.18)	95.57 (±1.12)	10.91 (±5.68)
Typicality test	96.84 (±0.05)	98.50 (±0.04)	4.24 (±0.25)	99.86 (±0.01)	99.87 (±0.01)	0.00 (±0)	45.16 (±0.1)	43.76 (±0.06)	75.60 (±0.35)
				$\mathcal{S}$-VAE					
Log likelihood	97.07 (±0.05)	98.62 (±0.06)	4.34 (±0.24)	99.85 (±0.02)	99.87 (±0.01)	0.01 (±0.01)	45.25 (±0.07)	44.45 (±0.05)	76.21 (±0.26)
Input complexity	41.74 (±1.11)	44.67 (±0.44)	80.00 (±5.68)	17.54 (±2.45)	37.02 (±0.83)	100 (±0)	94.79 (±1.63)	95.45 (±1.39)	12.73 (±7.57)
Typicality test	97.04 (±0.05)	98.59 (±0.05)	4.34 (±0.25)	99.86 (±0.02)	99.87 (±0.02)	0.00 (±0)	45.25 (±0.08)	44.45 (±0.09)	76.17 (±0.24)
Hole indicator (ours)	**89.05 (±0.25)**	**99.38 (±0.02)**	**16.1 (±0.72)**	**94.54 (±0.09)**	**99.01 (±0.02)**	**5.60 (±0.2)**	**87.37 (±0.16)**	**88.86 (±0.15)**	**19.25 (±0.46)**

The most robust scores are in bold. The highest values are in gray.

* 0's in FPR80 are possible since it is a value for false-positive rate at 80% of true-positive rate

variable $\mathbf{z}$, and $\boldsymbol{\theta}$ stands fr the generative parameters of the decoder responsible for the parametrization of the likelihood of the input $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$. Hence, it is possible to compute the sample standard deviation of the marginal likelihood under *importance sampling* by computing the sample standard deviation of the terms within the given sum. This constitutes the essence of the Stds of LLs score.

HOLE INDICATOR SCORE

For this score we sample the approximated posterior $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ with several latent codes $\mathbf{z}$ under a particular input $\mathbf{x}$ and compute the sample standard deviation of the log-likelihoods $\log p(\mathbf{x}|\mathbf{z})$:

$$\Sigma_{\mathbf{z}}[\mathbf{x}] = \sqrt{\frac{1}{N-1}\sum_{\mathbf{z}}\left(\log p(\mathbf{x}|\mathbf{z}) - \overline{\log p(\mathbf{x}|\mathbf{z})}\right)^2} \tag{4.17}$$

TYPICALITY

The test for typicality treats all input sequences as inliers if their entropy is sufficiently close to the entropy of the model, i.e., if the following holds for small ϵ then the given input is in-distribution:

$$\left| -\log p\left(\mathbf{x}^*\right) - \sum_{\mathbf{x}\in\mathcal{D}} \log p(\mathbf{x}) \right| \leq \epsilon \tag{4.18}$$

This score is applied to one-element sequences in our work since it is the most realistic scenario in practical applications of outlier detection.

INPUT COMPLEXITY

First, we compute the complexity estimate $L(\mathbf{x})$ by compressing the input $\mathbf{x}$ with JPEG2000. The result represents a string of bits: $C(\mathbf{x})$. After that we apply the normalization of the length of the resulting string by dimensionality d:

$$L(\mathbf{x}) = \frac{|C(\mathbf{x})|}{d}.$$

Subsequently the input complexity score is calculated in the following way (in bits per dimension):

$$S(\mathbf{x}) = -\log p(\mathbf{x}) - L(\mathbf{x}) \tag{4.19}$$

The higher the S score, the more indicative it is that the current input is the outlier.

4.7.8. COMPACTNESS ABLATION

Since the placement of the outliers within the unconstrained compact space with Vanilla VAEs is basically arbitrary, it can be the case that some outliers will still be successfully detected via hole indicator when these outliers are mapped within the same space as the inliers. Hence, in order to make an appropriate ablation study only for the compactness, we conducted the following experiments. We gradually increase the pixel intensity of the images from one to higher values by multiplying it with a scalar. We calculate the hole indicator for each intensity step for both Lipschitz VAE and Vanilla VAE. The corresponding results can be observed in the Table 4.8.

Table 4.8: Ablation of compactness with hole indicator.

	1x	**3x**	**5x**	**7x**	**9x**	**11x**	**13x**	**15x**
Vanilla VAE	100.0	100.0	100.0	99.69	53.40	0.00	0.00	0.03
Lipschitz VAE	100.0	100.0	100.0	99.48	99.36	95.32	99.48	95.70

As can be seen from the obtained values, there is a clear transition from the detectable outliers vs. non-detectable ones through the latent holes in the case of Vanilla VAE, and no degradation of the results in the case with the Lipschitz VAEs.

REFERENCES

[1] M. Glazunov and A. Zarras. "Vacant Holes for Unsupervised Detection of the Outliers in Compact Latent Representation". In: *Uncertainty in Artificial Intelligence (UAI)*. 2023.

[2] E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, and B. Lakshminarayanan. "Do deep generative models know what they don't know?" In: *arXiv preprint arXiv:1810.09136* (2018).

[3] A. v. d. Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu. "Conditional Image Generation with PixelCNN Decoders". In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. NIPS'16. Barcelona, Spain: Curran Associates Inc., 2016, pp. 4797–4805. isbn: 9781510838819.

[4] L. Dinh, J. Sohl-Dickstein, and S. Bengio. "Density estimation using Real NVP". In: *International Conference on Learning Representations*. 2017. url: https://openreview.net/forum?id=HkpbnH9lx.

[5] D. P. Kingma and M. Welling. *Auto-Encoding Variational Bayes*. 2013. arXiv: 1312.6114 [stat.ML].

[6] D. J. Rezende, S. Mohamed, and D. Wierstra. "Stochastic backpropagation and approximate inference in deep generative models". In: *International conference on machine learning*. PMLR. 2014, pp. 1278–1286.

[7] C. L. Lan and L. Dinh. "Perfect density models cannot guarantee anomaly detection". In: *ArXiv* abs/2012.03808 (2020).

[8] D. Hendrycks, M. Mazeika, and T. Dietterich. "Deep anomaly detection with outlier exposure". In: *arXiv preprint arXiv:1812.04606* (2018).

[9] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. Depristo, J. Dillon, and B. Lakshminarayanan. "Likelihood ratios for out-of-distribution detection". In: *Advances in neural information processing systems* 32 (2019).

[10] E. Daxberger and J. M. Hernández-Lobato. "Bayesian variational autoencoders for unsupervised out-of-distribution detection". In: *arXiv preprint arXiv:1912.05651* (2019).

[11] M. Glazunov and A. Zarras. “Do Bayesian Variational Autoencoders Know What They Don’t Know?” In: *The 38th Conference on Uncertainty in Artificial Intelligence*. 2022. url: https://openreview.net/forum?id=SSr4JOIs5l5.

[12] H. Choi, E. Jang, and A. A. Alemi. *WAIC, but Why? Generative Ensembles for Robust Anomaly Detection*. 2019. arXiv: 1810.01392 [stat.ML].

[13] E. Nalisnick, A. Matsukawa, Y. W. Teh, and B. Lakshminarayanan. “Detecting out-of-distribution inputs to deep generative models using typicality”. In: *arXiv preprint arXiv:1906.02994* (2019).

[14] J. Serrà, D. Álvarez, V. Gómez, O. Slizovskaia, J. F. Núñez, and J. Luque. “Input complexity and out-of-distribution detection with likelihood-based generative models”. In: *arXiv preprint arXiv:1909.11480* (2019).

[15] D. Hernández-Lobato, T. D. Bui, Y. Li, J. M. Hernández-Lobato, and R. E. Turner. “Importance weighted autoencoders with random neural network parameters”. In: *Workshop on Bayesian Deep Learning, NIPS*. Vol. 2016. 2016.

[16] R. Schirrmeister, Y. Zhou, T. Ball, and D. Zhang. “Understanding Anomaly Detection with Deep Invertible Networks through Hierarchies of Distributions and Features”. In: *ArXiv* abs/2006.10848 (June 2020).

[17] Z. Xiao, Q. Yan, and Y. Amit. *Likelihood Regret: An Out-of-Distribution Detection Score For Variational Auto-encoder*. 2020. arXiv: 2003.02977 [cs.LG].

[18] G. Cybenko. “Approximation by Superpositions of a Sigmoidal Function”. In: *Math. Control Signals Syst.* 2.4 (1989), pp. 303–314.

[19] K. Hornik. “Approximation Capabilities of Multilayer Feedforward Networks”. In: *Neural Netw.* 4.2 (1991), pp. 251–257.

[20] A. Pinkus. “Approximation Theory of the MLP Model in Neural Networks”. In: *Acta Numer.* 8 (1999), pp. 143–195.

[21] D. P. Kingma and M. Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

[22] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun. “What is the best multi-stage architecture for object recognition?” In: *2009 IEEE 12th International Conference on Computer Vision*. 2009, pp. 2146–2153. doi: 10.1109/ICCV.2009.5459469.

[23] Y. Yoshida and T. Miyato. *Spectral Norm Regularization for Improving the Generalizability of Deep Learning*. 2017. doi: 10.48550/ARXIV.1705.10941. url: https://arxiv.org/abs/1705.10941.

[24] M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier. *Parseval Networks: Improving Robustness to Adversarial Examples*. 2017. doi: 10.48550/ARXIV.1704.08847. url: https://arxiv.org/abs/1704.08847.

[25] T. Huster, C.-Y. J. Chiang, and R. Chadha. *Limitations of the Lipschitz constant as a defense against adversarial examples*. 2018. doi: 10.48550/ARXIV.1807.09705. url: https://arxiv.org/abs/1807.09705.

[26] C. Anil, J. Lucas, and R. Grosse. *Sorting out Lipschitz function approximation*. 2018. doi: 10.48550/ARXIV.1811.05381. url: https://arxiv.org/abs/1811.05381.

[27] M. Hein and M. Andriushchenko. "Formal Guarantees on the Robustness of a Classifier against Adversarial Manipulation". In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Curran Associates, Inc., 2017, pp. 2263–2273.

[28] Y. Tsuzuku, I. Sato, and M. Sugiyama. "Lipschitz-Margin Training: Scalable Certification of Perturbation Invariance for Deep Neural Networks". In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. NIPS'18. Montréal, Canada: Curran Associates Inc., 2018, pp. 6542–6551.

[29] Y.-Y. Yang, C. Rashtchian, H. Zhang, R. Salakhutdinov, and K. Chaudhuri. *A Closer Look at Accuracy vs. Robustness*. 2020. doi: 10.48550/ARXIV.2003.02460. url: https://arxiv.org/abs/2003.02460.

[30] B. Barrett, A. Camuto, M. Willetts, and T. Rainforth. "Certifiably robust variational autoencoders". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2022, pp. 3663–3683.

[31] P. Kirichenko, P. Izmailov, and A. G. Wilson. *Why Normalizing Flows Fail to Detect Out-of-Distribution Data*. 2020. arXiv: 2006.08545 [stat.ML].

[32] T. R. Davidson, L. Falorsi, N. De Cao, T. Kipf, and J. M. Tomczak. "Hyperspherical variational auto-encoders". In: *arXiv preprint arXiv:1804.00891* (2018).

[33] L. Falorsi, P. de Haan, T. R. Davidson, N. De Cao, M. Weiler, P. Forré, and T. S. Cohen. "Explorations in homeomorphic variational autoencoding". In: *TADGM Workshop @ ICML*. 2018.

[34] P. Xu, J. C. K. Cheung, and Y. Cao. "On variational learning of controllable representations for text without supervision". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 10534–10543.

[35] R. Li, X. Peng, and C. Lin. *On the Latent Holes of VAEs for Text Generation*. 2021. doi: 10.48550/ARXIV.2110.03318. url: https://arxiv.org/abs/2110.03318.

[36] J. Davis and M. Goadrich. “The relationship between Precision-Recall and ROC curves”. In: *Proceedings of the 23rd international conference on Machine learning*. 2006, pp. 233–240.

[37] Y. LeCun and C. Cortes. *MNIST handwritten digit database*. http://yann.lecun.com/exdb/mnist/. 2010.

[38] H. Xiao, K. Rasul, and R. Vollgraf. “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms”. In: *arXiv preprint arXiv:1708.07747* (2017).

[39] T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, and D. Ha. “Deep learning for classical japanese literature”. In: *arXiv preprint arXiv:1812.01718* (2018).

[40] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. “Weight uncertainty in neural network”. In: *International conference on machine learning*. PMLR. 2015, pp. 1613–1622.

5 ENHANCING ROBUSTNESS OF DEEP LEARNING VIA UNIﬁED LATENT REPRESENTATION

Adversarial examples and Out-of-Distribution (OoD) instances constitute major problematic inputs for Deep Neural Network (DNN). DNNs tend to be overconfident with their predictions assigning a different category with a high probability. In this work, we suggest a combined solution to both issues based on Variational Autoencoder (VAE). First, we put under scrutiny the recent successful results in detecting OoDs utilizing Bayesian epistemic uncertainty estimation over weights of VAEs. We obtain comparable results without using Bayesian inference over weights utilizing a standard procedure of importance sampling with the classical formulation of VAEs. Second, we dissect the marginal likelihood approximation analyzing the main source of variation responsible for distinguishing inliers versus outliers, and establish a link with the recent promising results in the detection of outliers using latent holes. Finally, we identify similarities between the latent representations of adversarial examples and OoD inputs, which allows for devising separate methods for their detection and mitigation taking into consideration their differences in the input space. The suggested approach enables to pre-train a model under particular input data that acts as a filter achieving two major goals: defending the DNN classifier against the potential attack and filtering out the OoDs. Once pre-trained, VAE can be plugged as a filter into any DNN image classifier of arbitrary architecture trained on the same data inputs without the need for its retraining or accessing the layers and weights of the DNN.

This chapter is based on the journal article to be submitted to IEEE Transactions on Neural Networks and Learning Systems

5.1. INTRODUCTION

Deep Neural Network (DNN) started being universally applied to tasks ranging from classification to anomaly detection. However, the thorough theoretical foundation of deep learning is still lacking. It results in a limited understanding of how deep neural networks generalize. Such a situation led to the discovery of the following facts:

- There is a possibility to mislead the DNN classification with specifically forged inputs that, while preserving the semantics from the point of view of human observers, result in a wrong classification category by a DNN, i.e., the adversarial examples [1–3].
- The inability of the DNN to infer the fact that the provided input doesn't adhere to the data distribution they have been previously trained on, i.e., the overconfidence of DNN predictions with OoD inputs [4–6].

The discriminative nature of the supervised image DNN classifiers implies learning a mapping from the input pixel space to the target labels. This mapping is usually considered to represent a categorical distribution over labels y given the particular input $\mathbf{x}$: $p(y|\mathbf{x})$. However, in practice, the categorical distribution is based on the softmax activation function. As it has been recently formally proved, the softmax does not provide the desirable properties of categorical distribution and operates in a way similar to the k-means clustering, i.e., it partitions the transformed input space into several cones where every cone represents a different category [7]. It may explain why DNNs struggle with both adversarial examples and OoD: on one hand, it is feasible to find an adversarial direction from one category to another while attempting to preserve as little modification to the input as possible, especially for the examples that lie far from the cluster centroids, and on another hand when a new unseen data input arrives, the k-means clustering would necessarily cluster it into one of the categories resulting in overconfident predictions of OoD as in-distribution examples.

Recently, many different solutions have been proposed to address either the issue with adversarial examples [8][9][10][11][12] or the issue with OoD inputs [13][14][15][16][17][18]. The works that provide a solution to both of the problems simultaneously within the same framework are few [19][20]. These works are based on learning the DNN class-conditional weight uncertainties which imply access to the model architecture, its weights, and output categories. Such an approach is closely interlinked with the DNN model under the protection and it also introduces the unnecessary inductive bias by class conditioning. It makes the suggested methods non-modularizable and non-transferrable in a plug-and-play manner to other DNN architectures that require the same functionality of protection against adversarial attacks or OoD detection and that have been trained on the same input data.

Conversely, we apply an unsupervised Deep Generative Modeling (DGM) to tackle both of the problems, i.e., instead of learning discriminative mapping $p(y|\mathbf{x})$ and subsequently attempting to estimate the uncertainty of the weights under different inputs, DGM allows learning the approximation of a true distribution over the training data: $p(\mathbf{x})$ which in theory should assign a low density to the OoD and adversarial inputs. However, recent research revealed that such estimations are prone to errors, often providing higher likelihood values to both OoD and adversarial examples than to in-distribution data [6].

To overcome this problem, we apply two recently suggested methods based on model parameter sensitivity analysis.

1. We use a *Bayesian* DGM, namely, VAE, that learns the weights uncertainty during training yielding the following posterior distribution: $p(\boldsymbol{\theta}|\mathcal{D})$ for the training data $\mathcal{D}$ over the model weights $\boldsymbol{\theta}$. It allows us to get an ensemble of the approximations of a true data distribution where each sample from the posterior $\boldsymbol{\theta} \sim p(\boldsymbol{\theta}|\mathcal{D})$ gives a separate instance of the model in the ensemble. Based on sampling from the posterior distribution, we estimate the likelihood of the input instance, however, instead of the usual calculation of the expected likelihood, we calculate the recently suggested scores of variance of the likelihoods between the different instance models in the ensemble [17]. The high degree of model/epistemic uncertainty is captured by the high values of the variance score.

2. We use recently suggested scores based on detecting if the corresponding latent code in the hole indicator [18]. We apply a single instance of classic VAE. Moreover, we enforce both compactness and continuity constraints on the latent representation and the corresponding encoder map.

We suggest a single DGM based on VAE to detect simultaneously both the OoD and adversarial inputs. We make the following contributions:

1. We explicitly disentangle Bayesian inference applied to parameters and Bayesian inference applied to latent codes, illustrating effective techniques for assessing the VAE's sensitivity to variations in its latent representation.

2. We identify similarities in locations between the latent representations of adversarial examples and OoD inputs. As a result, we identify that adversarial examples can be successfully detected using a hole indicator, which successfully works in the case of transferability from discriminative models to generative, and in the case of attacking generative models.

3. We introduce an efficient algorithm for active defense against weak adversarial examples that allow for identification if the generative

model is under attack, distinguishing them from both OoD inputs and discriminative adversarial attacks.

4. Our implementation of VAE represents a separate filtering layer that is pre-trained on any given dataset and that models $p(\mathbf{x})$ independently of any other model so it can be plugged into any DNN trained on the same data that requires OoD and adversarial examples detection; the filtering is done based on the values computed with a hole indicator.

We empirically evaluate the suggested approach based on the several datasets achieving promising results.

5.2. PROBLEM STATEMENT

There are two types of problematic inputs that we deal with in this chapter: adversarial examples and OoDs.

5.2.1. ADVERSARIAL ATTACKS

There are two different perspectives on adversarial examples that give rise to two different definitions: one from the perspective of the generalization properties of the DNN and the other from the attacker's perspective.

From the generalization perspective, an adversarial example [1] is a technique in which the input for the DNN image classier is intentionally modified to look almost the same as the original image to the human eye. Yet, it is perceived as something completely different by DNN. DNNs incorrectly classify such adversarial examples from the human perspective. On the other hand, the attacker perspective does not necessarily demand the part that relates to the imperceptibility of the difference [21]. On the contrary, if the miscreants want their attack's outcome to succeed, they should not constrain themselves to the superfluous imperceptibility demands. In this chapter, we evaluate the imperceptible examples. In addition, we analyze both alternatives from the perspective of their internal representation.

Furthermore, we conduct the experiments with both the adversarial examples generated for the discriminative model under attack and the adversarial examples generated to attack our defending VAE filter.

Specifically, for the former case, we consider three types of such attacks: Fast Gradient Sign Method (FGSM (FGSM) [2], Carlini-Wagner (CW) attack [3], and Jacobian-based Saliency Map Attack (JSMA) [22]. For the latter case, we evaluate attacks on the encoder in the same vein as in [23]. All inputs are normalized into the [0,1] range.

FAST GRADIENT SIGN METHOD

The FGSM attacks DNNs by leveraging their learning process based on gradients [2]. FGSM can be described by the following formula:

$$\mathbf{x}' = \mathbf{x} + \lambda \cdot \mathrm{sgn}(\nabla_{\mathbf{x}} \ell(h_{\boldsymbol{\theta}}(\mathbf{x}), y_s)), \mathbf{x}' \in [0,1]^n$$

Here $\nabla_{\mathbf{x}}\ell$ is the gradient of the loss function with respect to the original input pixel vector $\mathbf{x}$, y_s is the true or source label for $\mathbf{x}$, and $\boldsymbol{\theta}$ stands for the model parameters that are constant.

Gradient w.r.t. $\mathbf{x}$ is easier to calculate with backpropagation than for $\boldsymbol{\theta}$ which allows the fast generation of adversarial examples. FGSM exploits gradient ascent to increase the loss. Subsequently, the sign applies a max-norm constraint on the gradient value, and λ represents a small magnitude of the step in the direction of increasing the loss. It represents the untargeted type of adversarial attacks.

FGSM can be converted into a targeted attack by means of substituting the source label with a target one y_t and doing gradient descent instead of ascent, namely:

$$\mathbf{x}' = \mathbf{x} - \lambda \cdot \mathrm{sgn}(\nabla_{\mathbf{x}} \ell(h_{\boldsymbol{\theta}}(\mathbf{x}), y_t)), \mathbf{x}' \in [0,1]^n$$

However, due to the fact that FGSM is designed to be fast rather than optimal, it is not necessarily guaranteed to produce the targeted adversarial examples of minimal perturbations.

CARLINI-WAGNER

Carlini-Wagner (CW) attack [3] aims at optimality in contrast with FGSM, i.e., it attempts to generate as little noise as possible in order to succeed in the attack. It poses the following optimization objective:

$$\text{minimize } ||\varepsilon||_p \ \text{ subj. to } h_{\boldsymbol{\theta}}(\mathbf{x}+\varepsilon) = y_t, \quad \mathbf{x}+\varepsilon \in [0,1]^n$$

where, $\mathbf{x} \in [0,1]^n$ represents an image, $\varepsilon \in [0,1]^n$ is added noise to the image and y_t is a target class label of the image under attack. The noise level is calculated in terms of L_p norms. Authors consider several norms, in this work we concentrate on L_2-norm.

It represents a targeted attack with very strong properties, till the moment, it is one of the strongest known adversarial attacks.

JACOBIAN-BASED SALIENCY MAP ATTACK

JSMA [22] leverages the saliency maps to devise an adversarial input. Namely, it computes the forward derivative of the whole DNN (Jacobian) w.r.t. the input, and based on this derivative, it constructs the saliency map. Large absolute values of the saliency map reveal the features with a significant impact on the final output. The JSMA takes the maximum absolute value and perturbs it by a hyperparameter θ and repeats the process. The stopping criteria are either a successful attack with misclassification or reaching the total perturbation threshold of Υ.

ATTACK ON ENCODER

This attack aims at maximization of the symmetric KL-divergence between the latent code of the reference input and the latent code of the reference input with the added perturbation:

$$\varepsilon = \arg\max_{\|\varepsilon\|_p \leq \delta} \mathrm{SKL}\left[q(\mathbf{z}|\mathbf{x}+\varepsilon), q(\mathbf{z}|\mathbf{x})\right] \tag{5.1}$$

where SKL is the symmetric KL-divergence, δ is the maximum amount of noise, and $q(\mathbf{z}|\mathbf{x})$ is the encoder under attack. The resulting adversarial perturbation is denoted as ε.

5.2.2. TRANSFERABILITY

It has been discovered that different architectures of DNNs trained to tackle the same classification problem on similar datasets tend to have similar fairly piece-wise linear decision boundaries that separate categories in the input data domain [2]. This property is called transferability.

Transferability is especially dangerous since it allows to devise an adversarial attack that universally targets all DNNs with a similar final objective in a black-box manner [24].

Moreover, since we utilize a generative approach we explore if there is transferability from the adversarial examples generated for a discriminative model to a generative one.

5.2.3. OUT-OF-DISTRIBUTION

Deploying a successful classifier requires from the system the ability to detect input data that are statistically anomalous or significantly different from those used in training. This is especially important for DNN classifiers since DNNs with the softmax classifier tend to produce overconfident predictions even for such Out-of-Distribution (OoD) inputs [19]. The lack of reliability of DNN classifiers when faced with OoDs was recently addressed by various methods [5][15][25].

According to recent research, the softmax activation function does not model a categorical distribution but represents a k-means clustering [7]. That is why it seems logical to seek another approach. We decided to consider using unsupervised DGMs for that purpose. In our case, we apply the same VAE model to detect the OoDs based on the sensitivity analysis [17, 18].

5.3. METHODS

We employ an approach, which is based on the sensitivity analysis of the model parameter with respect to the different inputs. Namely, we test the level of stability of our model when dealing with OoDs versus IDs. There are two possible ways to achieve this goal. The first one is to utilize epistemic uncertainty estimation that would allow us to sample model parameters to be subsequently used for sensitivity analysis. The second one is to employ the learned posterior distribution over the latent codes in VAE and sample posterior for different latent codes. This approach does not change the parameters of DNNs used in the model, however, it allows conducting sensitivity analysis with respect to different sampled hypotheses from the latent posterior.

5.3.1. EPISTEMIC UNCERTAINTY IN OOD AND IN ADVERSARIAL ATTACKS

It has been shown that DGMs do not produce valid estimations of $p(\mathbf{x})$ when it comes to distinguishing between OoD and in-distribution [6]. Most of the results reveal DGMs being overconfident when dealing with OoD data. Another work dedicated to adversarial defense [26] showed that it is possible to statistically differentiate between adversarial vs non-adversarial input data using DGMs. In this work, we first rely on the estimation of the weight uncertainty to address this issue utilizing Bayesian and, in particular, variational inference.

5.3.2. BAYESIAN INFERENCE

The general idea of Bayesian inference is to estimate the posterior distribution of an unknown parameter given the data and the prior distribution:

$$p(\boldsymbol{\theta}|\mathbf{x}) = \frac{p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{x})} = \frac{p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}} \tag{5.2}$$

where the nominator of the fraction consists of the known prior distribution $p(\boldsymbol{\theta})$ over the parameter and a known or possible-to-calculate likelihood of the observed data $p(\mathbf{x}|\boldsymbol{\theta})$.

In cases when we deal with high dimensional distributions and our posterior and prior distributions are non-conjugate, one cannot compute a closed-form expression to the problem due to the intractability of the marginal integral in the denominator. In such situations, there are two main methods used to estimate the posterior:

- Markov chain Monte Carlo (MCMC)
- variational inference

MCMC represents a class of algorithms based on sampling, one prominent example of which is Metropolis-Hastings [27][28]. This algorithm estimates the posterior utilizing guided sampling from this intractable distribution ignoring the denominator and sampling from the unnormalized nominator.

Variational inference, on the other hand, estimates the posterior not by sampling but by converting the problem of posterior estimation to the optimization problem.

5.3.3. VARIATIONAL INFERENCE

Since the posterior is intractable, one can choose a function from the family of tractable probability distributions that is closest to the intractable posterior. This goal may be achieved by minimizing the KL-divergence between the distribution $q(\boldsymbol{\theta})$ from the tractable family $\mathcal{Q}$ and intractable posterior distribution $p(\boldsymbol{\theta}|\mathbf{x})$:

$$\underset{q(\boldsymbol{\theta})\in\mathcal{Q}}{\arg\min}\,\mathrm{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathbf{x})) \tag{5.3}$$

The only problem is the fact that $p(\boldsymbol{\theta}|\mathbf{x})$ is indeed intractable so the given objective doesn't provide an immediate criterion for divergence minimization.

However, one can obtain an optimization objective by the following reformulation:

$$\begin{gathered}
\log(p(\mathbf{x})) = \int q(\boldsymbol{\theta})\log(p(\mathbf{x}))d\boldsymbol{\theta} = \\
\int q(\boldsymbol{\theta})\log(\frac{p(\mathbf{x},\boldsymbol{\theta})}{p(\boldsymbol{\theta}|\mathbf{x})})d\boldsymbol{\theta} = \\
\int q(\boldsymbol{\theta})\log(\frac{p(\mathbf{x},\boldsymbol{\theta})q(\boldsymbol{\theta})}{p(\boldsymbol{\theta}|\mathbf{x})q(\boldsymbol{\theta})})d\boldsymbol{\theta} = \\
\int q(\boldsymbol{\theta})\log(\frac{p(\mathbf{x},\boldsymbol{\theta})}{q(\boldsymbol{\theta})})d\boldsymbol{\theta} + \int q(\boldsymbol{\theta})\log(\frac{q(\boldsymbol{\theta})}{p(\boldsymbol{\theta}|\mathbf{x})})d\boldsymbol{\theta} = \\
\mathcal{L}(q(\boldsymbol{\theta})) + \mathrm{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathbf{x}))
\end{gathered}$$

So the log-marginal is reformulated as a sum of KL-divergence that we would like to minimize and some other term $\mathcal{L}(q(\boldsymbol{\theta}))$ that depends only on our tractable $q(\boldsymbol{\theta})$ distribution. This term is called evidence lower bound.

EVIDENCE LOWER BOUND (ELBO)

The Evidence Lower Bound (ELBO) is an optimization objective, commonly used for variational methods. As it has been shown in the previous section: starting from the *marginal log-likelihood*: $\log(p(\mathbf{x}))$, the variational lower bound $\mathcal{L}(q(\boldsymbol{\theta}))$ can be derived [29] [30]:

$$\log(p(\mathbf{x})) = \mathcal{L}(q(\boldsymbol{\theta})) + \mathrm{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathbf{x})) \tag{5.4}$$

with

$$\mathcal{L}(q(\boldsymbol{\theta})) = \int q(\boldsymbol{\theta}) \log(\frac{p(\mathbf{x},\boldsymbol{\theta})}{q(\boldsymbol{\theta})}) d\boldsymbol{\theta} \tag{5.5}$$

The KL-Divergence is used to evaluate the dissimilarity of the approximated distribution $q(\boldsymbol{\theta})$ w.r.t to the true distribution $p(\boldsymbol{\theta}|\mathbf{x})$. KL is always a positive number and is closer to zero when the distributions are similar. With this property, the above equation can be formulated with the following inequality that defines $\mathcal{L}(q(\boldsymbol{\theta}))$ as ELBO for the marginal log-likelihood $\log(p(\mathbf{x}))$.

$$\log(p(\mathbf{x})) \geq \mathcal{L}(q(\boldsymbol{\theta})) \tag{5.6}$$

The *gap* between the ELBO and the marginal log-likelihood is called *tightness* of the bound, which decreases when the true posterior distribution is approximated well (hence smaller KL).

This leads us to the formulation of the main optimization objective in the variational inference:

$$\underset{q(\boldsymbol{\theta})\in\mathcal{Q}}{\arg\max}\, \mathcal{L}(q(\boldsymbol{\theta})) \tag{5.7}$$

As it can be seen, ELBO allows us to approximate the intractable posterior just using the known information: the prior and the likelihood since $p(\mathbf{x},\boldsymbol{\theta}) = p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})$ without the need to know anything about the true intractable posterior.

OPTIMIZING ELBO

There are two main approaches to ELBO optimization:

- the first one is to allocate the optimal $q(\boldsymbol{\theta})$ by means of setting the factorized constraint on the family of our distributions $\mathcal{Q}$, namely, we consider only those distributions that adhere to the following factorization:

$$q(\boldsymbol{\theta}) = \prod_{j=1}^{m} q_j(\boldsymbol{\theta}_j), \boldsymbol{\theta} = [\boldsymbol{\theta}_1, \cdot, \boldsymbol{\theta}_m] \tag{5.8}$$

so we assume that all parameters are independent from each other. Here we optimize the ELBO functional by means of the calculus of variations and coordinate ascent optimization method, it is the so-called mean-field approximation approach;

- the second one is making a parametric approximation, i.e., we consider only those distributions that adhere to the space of some parametric family:

$$q(\boldsymbol{\theta}) = q(\boldsymbol{\theta}|\lambda) \tag{5.9}$$

The difference between the two approaches lies in the scaling properties of the underlying optimization procedures: the second approach converts the problem of functional optimization to the parametric optimization where the optimal parameter λ has to be found. It can be implemented by means of mini-batch optimization algorithms such as stochastic gradient ascent which scales up very efficiently to the large volume of data. The second approach is also called amortized variational inference.

5.3.4. VARIATIONAL AUTOENCODER

Variational Autoencoder (VAE) represents an amortized variational inference approach. VAE consists of two main parts: encoder and decoder.

First of all, the unknown parameters $\boldsymbol{\Theta}$ in our previously introduced Bayesian inference sense in VAE represent the latent space. VAE is a non-fully observable model approach with latent random variable [31]. The latent space follows some predefined simple prior distribution as before, e.g., standard Normal: $p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbb{I})$. This latent space could be easily sampled after training for the subsequent generation of the samples that are coming from the same distribution as the input data.

Secondly, encoder is again represented by the neural network but now it is used for our posterior density parametrization, namely, we want to approximate our true posterior $p(\mathbf{z}|\mathbf{x})$ by some parametric family of distributions $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x};\lambda)$:

$$q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x};\lambda) \approx p(\mathbf{z}|\mathbf{x}) \tag{5.10}$$

where we optimize for the parameters by means of the neural network, for example, if we restrict our posterior family of distributions to be Gaussians, the encoder will learn the following parametrization:

$$(\boldsymbol{\mu}, \boldsymbol{\sigma}^2) = \text{EncoderDNN}_{\boldsymbol{\phi}}(\mathbf{x}) \tag{5.11}$$

$$q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x};\lambda) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, diag(\boldsymbol{\sigma}^2)) \tag{5.12}$$

Please note that the encoder tries to approximate the posterior by learning the appropriate parameters for this posterior (in our example, λ is represented by $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ for Gaussian), however, since encoder is in itself a neural network, it has its own parameters $\boldsymbol{\phi}$, these are so-called variational parameters and they include the weights and biases of the neural network.

Finally, the decoder is responsible for the mapping of the latent space back to the data space. It is also approximated by a separate neural network that makes this mapping possible. Here, the decoder uses the fact that if there is a random variable Z with one distribution, it is possible to create a separate random variable $X = g(\mathbf{z})$ with a totally different distribution [32].

The optimization objective is the same: ELBO which allows joint optimization with respect to both variational parameters $\boldsymbol{\phi}$ of the encoder and generation parameters $\boldsymbol{\theta}$ of the decoder:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}[\log(p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})) - \log(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}))]$$

VAEs are trained in an unsupervised manner from data and are widely used for generative purposes.

5.3.5. ESTIMATION OF THE MARGINAL LIKELIHOOD

As suggested in [33], as soon as the VAE is trained, it is possible to estimate the likelihood of the input under the generative model using *importance sampling* w.r.t to the approximated posterior, namely:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) \simeq \frac{1}{N}\sum_{i=1}^{N}\frac{p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}_{(i)})}{q_{\boldsymbol{\phi}}(\mathbf{z}_{(i)}|\mathbf{x})}, \quad \text{where} \quad \mathbf{z}_{(i)} \sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \tag{5.13}$$

However, as it has been discovered in [6], we cannot rely directly on the likelihood estimations produced by a single DGM. This discovery is not surprising taking into consideration the fact that DGMs obtain the optimal parameters $\boldsymbol{\theta}^*$ under the Maximum Likelihood Estimation (MLE) for the $p(\mathcal{D}|\boldsymbol{\theta})$, where $\mathcal{D}$ represents the training data, resulting in a point estimate. Due to the fact that in modern DNNs $|\boldsymbol{\theta}| \gg |\mathcal{D}|$, it is possible that there may be several models $\boldsymbol{\theta}$ that generated $\mathcal{D}$. Hence, it is impossible to estimate the epistemic uncertainty with a point estimate which results in the inability of the model to provide the robust estimation of the likelihood for OoD and adversarial examples.

WEIGHT UNCERTAINTY: BAYES BY BACKPROPAGATION

Since we use variational inference to approximate our VAE posterior based on the assumption of the model with latent variables, we have chosen to apply the same variational approach to the weight uncertainty estimation instead of a point MLE estimate. Namely, we approximate posterior distribution of the DGM parameters given the training data $p(\boldsymbol{\theta}|\mathcal{D})$ based on the method suggested in [34]. This method initially was applied to the supervised learning, however, nothing prevents us from using it in the unsupervised setting. The ELBO objective is formulated in the following way:

$$\mathcal{L}_{\boldsymbol{\theta}}(\mathcal{D},\lambda) = \int q(\boldsymbol{\theta}|\lambda) log(\frac{p(\boldsymbol{\theta})p(\mathcal{D}|\boldsymbol{\theta})}{q(\boldsymbol{\theta}|\lambda)}) d\boldsymbol{\theta} \tag{5.14}$$

The approximation of the negative ELBO is obtained by:

$$-\widehat{\mathcal{L}_{\boldsymbol{\theta}}}(\mathcal{D},\lambda) = \frac{1}{N}\sum_{i=1}^{N}\left[\log q(\boldsymbol{\theta}^{(i)}|\lambda) - \log p(\boldsymbol{\theta}^{(i)}) - \log p(\mathcal{D}|\boldsymbol{\theta}^{(i)})\right] \tag{5.15}$$

where $\boldsymbol{\theta}^{(i)}$ is sampled from the posterior $q(\boldsymbol{\theta}^{(i)}|\lambda)$.

We assume a diagonal Gaussian distribution for the variational posterior with parameters μ and σ. In order to make σ to be always non-negative we apply the same reparametrization as it was suggested in [34], namely $\sigma = \log(1+\exp(\rho))$, yielding the following posterior parameters $\lambda = (\mu, \rho)$.

For the prior we also use the suggested scale mixture of two Gaussians:

$$p(\boldsymbol{\theta}) = \pi\mathcal{N}(\boldsymbol{\theta}|0,\sigma_1^2) + (1-\pi)\mathcal{N}(\boldsymbol{\theta}|0,\sigma_2^2), \quad \text{where} \quad \pi = 0.5 \tag{5.16}$$

By adding weight uncertainty to the VAE, we are implementing *a Bayesian* VAE.

SCORES USED FOR PROBLEMATIC INPUTS DETECTION

After we approximated the variational posterior over the weights, the usual practice is to estimate the expected likelihood, the exact form of which can be formulated like this:

$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{\theta} \tag{5.17}$$

The unbiased estimate of which can be obtained in the following way:

$$\mathbb{E}_{p(\boldsymbol{\theta}|\mathcal{D})}[p(\mathbf{x}|\boldsymbol{\theta})] \simeq \frac{1}{N}\sum_{i=1}^{N} p(\mathbf{x}|\boldsymbol{\theta}_i); \quad \text{where} \quad \boldsymbol{\theta} \sim p(\boldsymbol{\theta}|\mathcal{D}) \tag{5.18}$$

And $p(\mathbf{x}|\boldsymbol{\theta}_i)$ is computed by importance sampling as in (5.13). As soon as the expected likelihood is estimated, one can apply some threshold that would distinguish if the considered input adheres to the in-distribution sample or not.

In this work, however, we aim at the estimation of the model parameter sensitivity. Hence, we calculate the sample standard deviation of the marginal log-likelihoods returned by the models within the ensemble [17]:

$$\Sigma_{\boldsymbol{\Theta}}[\mathbf{x}] = \sqrt{\frac{1}{N-1}\sum_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} (\log p(\mathbf{x}|\boldsymbol{\theta}) - \overline{\log p(\mathbf{x}|\boldsymbol{\theta})})^2} \tag{5.19}$$

It measures the variation within the log-likelihoods, so if there is a different level of sensitivity between the inliers and problematic inputs, then the standard deviation will capture this difference: the higher the value, the more uncertainty there is between the models about a particular input.

Furthermore, in the case of a single VAE, we instead apply the hole indicator [18]. For this score we sample the approximated posterior $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ with several latent codes $\mathbf{z}$ under a particular input $\mathbf{x}$ and compute the sample standard deviation of the log-likelihoods $\log p(\mathbf{x}|\mathbf{z})$:

$$\Sigma_{\mathbf{z}}[\mathbf{x}] = \sqrt{\frac{1}{N-1}\sum_{\mathbf{z}} \left(\log p(\mathbf{x}|\mathbf{z}) - \overline{\log p(\mathbf{x}|\mathbf{z})}\right)^2} \tag{5.20}$$

The higher the score, the farther the input is from the IDs.

Both of these scores allow for measuring the level of stability of the model with respect to different parameters. We can detect our problematic inputs based on the difference in this stability level.

SCORE FOR DISTINGUISHING BETWEEN ADVERSARIAL AND OOD INPUTS

Note that utilizing the same score for both outliers and adversarial examples does not allow us to distinguish between them. To address this issue, we devise a simple algorithm that allows for such a distinction.

Leveraging the intuition that adversarial examples also tend to land on the latent holes, it makes it possible to utilize the recently introduced approach for utilizing Hamiltonian Monte Carlo (HMC) to reevaluate the current latent code [23]. If the generated image of the reevaluated latent code from the region close to the mean of the posterior is similar to the one that has been provided as the input, then it is highly likely to assume that there is an ongoing attack on the DNN. This similarity is based on Multi-Scale Structural Similarity (MSSSIM). This method represents an active defense approach.

Algorithm 2: Active Defense Algorithm

```
H
Input: $\mathbf{x}$, $M$, $\boldsymbol{\theta}$, $\varnothing(\cdot)$, $h^{\text{enc}}(\cdot)$, $h^{\text{dec}}(\cdot)$, hmc$(\cdot)$, MSSSIM$(\cdot,\cdot)$
Output: Decision on whether $\mathbf{x}$ is an attack, an outlier or an inlier
// Get a reconstruction and a latent code
$\mathbf{z} \leftarrow h^{\text{enc}}(\mathbf{x})$
$\mathbf{x}' \leftarrow h^{\text{dec}}(\mathbf{z})$
// check if $\mathbf{z}$ is in the hole
if $\varnothing(\mathbf{z})$ then
    // Run active defense with $M$ steps
    for $i = 1$ to $M$ do
        // One step of HMC:
        $\mathbf{z} \leftarrow \text{hmc}(\mathbf{z})$
    $\mathbf{x}_{\text{hmc}} \leftarrow h^{\text{dec}}(\mathbf{z})$
    $\gamma_{\text{hmc}} \leftarrow \text{MSSSIM}(\mathbf{x}, \mathbf{x}_{\text{hmc}})$
    $\gamma_{\text{no\_hmc}} \leftarrow \text{MSSSIM}(\mathbf{x}, \mathbf{x}')$
    // Check MSSSIM gain with a threshold $\boldsymbol{\theta}$
    if $|\gamma_{\text{hmc}} - \gamma_{\text{no\_hmc}}| > \boldsymbol{\theta}$ then
        return "Attack"
    else
        return "Outlier"
else
    return "Inlier"
```

The starting point is to identify if the corresponding latent code for the current input is located in the hole utilizing a hole indicator. If it is not in the hole then we can immediately classify it as in-distribution input. Otherwise, the distinguishing between OoD and adversarial attack is implemented on the basis of the restored latent code via HMC. The insight is that the resulting distance in the input space should be much closer for the adversarial inputs than for the OoDs (see Algorithm 3).

As a result, we implement the robust VAE model against both outliers and adversarial examples with two level of defense allowing to identify if we are being attacked or not.

ENFORCED CONTROLLED CONTINUITY AND COMPACTNESS BY LIPSCHITZ CONTINUITY

To increase robustness even further, we enforce a predefined Lipschitz constant on the encoder map of the VAE. First, it reduces the ability of the attacker to gain substantial benefits while generating adversarial examples with VAEs that possess encoding maps with great Lipschitz constants. Second, it allows the control of the properties of compactness of the mapped image to the latent space which has recently been demonstrated as being beneficial for outlier detections [18] utilizing latent hole. To that end, we employ the GroupSort activation function and compute the corresponding Lipschitz constant following the protocol described in [18].

5.3.6. DISENTANGLING THE VARIATION AND THE BAYESIAN INFERENCE.

We identify the source of the variation observed with Bayesian VAEs. The general procedure of the marginal likelihood estimation follows these steps:

1. Sampling the weights from the estimated posterior: $\boldsymbol{\theta} \sim p(\boldsymbol{\theta}|\mathcal{D})$.
2. Estimating the marginal likelihood for separate sampled models (Equation 4.16).
3. Computing a single value based on the separate estimated marginal likelihoods.

As it can be seen, there are two possible sources of variation, namely, variation from *Step 1* and variation from *Step 2*. It was hypothesized in [13, 17] that the *Bayesian* inference over the DNN parameters is responsible for the observed variation of the results. In our work, we test if it is indeed the case by eliminating the first step and estimating the variation in the case of a simple classical VAE; namely, instead of sampling from $p(\boldsymbol{\theta}|\mathcal{D})$, we use a single VAE model that is used for marginal likelihood estimation several times. In such a case, all the variation comes only from the importance sampling. We apply the same scores as for the Bayesian VAEs to identify if the variation still persists for the classical VAEs.

DISSECTING THE SOURCE OF VARIATION.

By taking the log of both sides of the Equation 4.16 and by factoring the joint probability $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})$, we can obtain the following equation for the importance sampling:

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}) \simeq \frac{1}{N}\sum_{i=1}^{N}\Big[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}_{(i)}) + \log p(\mathbf{z}_{(i)}) - \log q_{\boldsymbol{\phi}}(\mathbf{z}_{(i)}|\mathbf{x})\Big] \tag{5.21}$$

All the scores that we have considered so far are measuring the variation of the left-hand side. To better understand where the variation comes from, we also consider the separate constituents of the right-hand side; namely, we measure standard deviations of all three terms separately, which allows us to identify the most uncertain term in case of OoD detection.

5.3.7. ANALYZING LATENT REPRESENTATION

Our experiments confirm that adversarial examples can be identified using the same scores that were successfully applied to outliers. It implies that adversarial examples occupy latent holes similar to the OoDs. The difference is that it is possible to control the strength of the adversarial attack. Hence, we can visualize the dynamics of the attack strength with respect to the learned data representation in the latent space. To that end, we employ the learning procedure suggested in [35] to mold the latent data manifold into a mixture of Gaussians, the so-called Variational Deep Embeddings (VADEs). Such an approach allows us to calculate distances to the centroids of the learned clusters that can be visually inspected. In addition, no Lipschitz constraints are used for these experiments, so no restraints are applied for the adversarial locations.

5.4. EXPERIMENTS AND RESULTS

Our experiments have been conducted on several datasets widely used for validation of OoD and adversarial attacks, namely: MNIST[36], FashionMNIST[37], SVHN[38] and CIFAR10[39].

First, we estimated the impact of the number of dimensions of the latent space on the loss function. The dimensionality is closely connected with the dataset the model is trained on. MNIST and FashionMNIST results reveal that there is no need to go over 10 latent dimensions since the loss function didn't significantly decrease after that value. For SVHN, we experimented with the number of latent dimensions up to 50, and the most optimal results have been achieved with dimensionality = 20.

For our tests, we used two different architectures: for grayscale images, we applied a multilayer perceptron for both the encoder and decoder with two fully connected hidden layers. For RGBs images, we applied a Convolutional Neural Network (CNN) with two convolutional layers of 32 and 64 filters. For epistemic uncertainty estimation, all layers that contain parameters have been enhanced with the BBB, namely, convolutional 2D, fully connected, and convolutional 2D transpose. All the rest, such as reshape and flatten, are used with their default implementations as provided by the Tensorflow Keras [40] framework. For a single VAE, we used the same architectures without the BBB. Moreover, the continuity of the encoder map is controlled via the specifically predefined Lipschitz constant calculated in the same way as in [18] for the cases where the hole indicator is used.

All models have been trained for 1000 epochs. For the evaluation of the inputs, we sampled 100 different models for our ensemble. Since we have a doubly stochastic nature of the results, one due to the sampling from latent posterior and the second one due to the sampling from the weights posterior, we ran the experiments 10 times each and averaged the final results.

For our implementation of BBB we noticed that random Normal initializer of the DNNs weights suggested as a prior in the original paper [34] resulted in very slow convergence. So, to speed up the process, we also experimented with the following parameters: random Normal initializer with 0 mean and 0.1 standard deviation for μ and constant initializer for $\rho = -3$, which improved the training speed.

The metrics that we used to validate both OoD are the area under Receiver operating characteristic (ROC) curve (ROC AUC), the area under the precision-recall curve (AUPRC), and the false-positive rate at 80% of true-positive rate (FPR80). We used two OoD benchmarks: (i) MNIST as in-distribution vs FashionMNIST as OoD and (ii) CIFAR10 as in-distribution vs SVHN as OoD. As it can be observed from the results of Stds of LLs in Table 5.1, they are comparable with the state-of-the-art in the field [13, 17].

Table 5.1: OoD detection results with Bayesian VAE based on Stds of LLs

Metric	**MNIST vs FashionMNIST**	**CIFAR10 vs SVHN**
ROC AUC↑	99.76	90.88
AUPRC↑	99.77	89.64
FPR80↓	0.00	11.72

Subsequently, we performed experiments utilizing a single classical VAE testing if the previously observed variation persists. The obtained results clearly demonstrate that variation that comes from the importance sampling is sufficient for the detection of the OoD inputs (see Table 5.2). It allows us to disentangle the variation from the Bayesian inference over the weights and directly use latent posterior sampling with a classical VAE.

Table 5.2: OoD detection results with classical VAE based on Stds of LLs

Metric	**MNIST vs FashionMNIST**	**CIFAR10 vs SVHN**
ROC AUC↑	99.81	93.07
AUPRC↑	99.82	91.23
FPR80↓	0.00	11.36

In addition, we calculate the sample standard deviation of the separate terms on the right-hand side of Equation 5.21. The obtained values reveal the fact that most of the observed variance results from the likelihood term $\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}_{(i)})$ that is parameterized by the decoder DNN. The boxplots of the standard deviations for all three terms (in the case of the classical VAE trained on the Fashion-MNIST dataset and tested on the MNIST as OoD) are plotted in Figure 5.1. As it can be seen, the variance obtained by the variational inference over the latent variable $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ does not result in high values as one may have expected, which denotes that most of the responsibility for the variation is laid on the decoder which is more sensitive to the OoD inputs versus IDs. Such sensitivity has been observed for all of the considered datasets and models, which strongly supports the usage of the recently introduced hole indicator [18] for the OoD input detection.

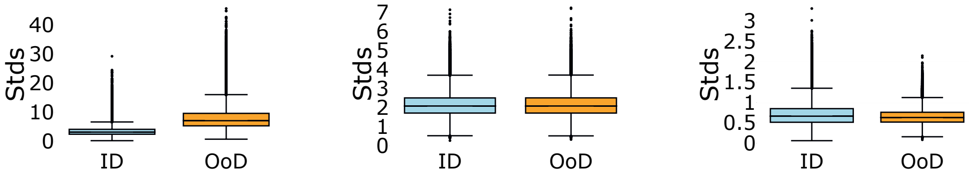


Figure 5.1: Standard deviations of the separate components of the ELBO *within* the importance samples for Fashion-MNIST as in-distribution (blue) vs MNIST as out-of-distribution (orange). **Left:** variation of the log-likelihood of the decoder $\log p(\mathbf{x}|\boldsymbol{z})$ **Middle:** variation of the encoder $\log q(\boldsymbol{z}|\mathbf{x})$. **Right:** variation of the latent prior $\log p(\boldsymbol{z})$.

For the generation of the adversarial inputs, we used the Cleverhans framework [41]. We use the default discriminative DNN architecture for our victim classifier provided within this framework. We benchmark our model on three common attacks: FGSM, CW and JSMA (see Section 5.2.1 for more details). For FGSM, we used $\epsilon = 3$, for CW we used attack under L_2-norm, and we applied 128 attack iterations with 0.2 learning rate, and, finally, for JSMA we used $\theta = 1$ and $\gamma = 0.1$. For CW and JSMA we generated targeted attacks per each of 10 categories available in MNIST and FashionMNIST, for SVHN we applied an untargeted attack. In the case of FGSM, all inputs implemented an untargeted attack.

Consequently, as Bayesian inference over DNN weights proves to be unnecessary, we employ a single VAE model, evaluating the results using a hole indicator (refer to Eq. 5.20).

Results of the experiments convincingly demonstrate that there is indeed transferability between discriminative and generative models. The adversarial examples generated for the classifier can be detected by the VAE, which is trained on the same dataset in an unsupervised manner (see Tables 5.3 - 5.5). It is reproduced across a wide range of adversarial attacks and datasets.

Table 5.3: Discriminative adversarial results: MNIST

Metric	**MNIST vs FGSM**	**MNIST vs CW**	**MNIST vs JSMA**
ROC AUC↑	100.00	92.24	93.01
AUPRC↑	100.00	92.55	90.13
FPR80↓	0.00	11.72	10.16

Table 5.4: Discriminative adversarial results: FashionMNIST

Metric	**FMNIST vs FGSM**	**FMNIST vs CW**	**FMNIST vs JSMA**
ROC AUC↑	96.49	95.01	83.97
AUPRC↑	96.52	92.36	78.38
FPR80↓	5.47	10.94	16.66

Table 5.5: Discriminative adversarial results: SVHN

Metric	**SVHN vs FGSM**	**SVHN vs CW**	**SVHN vs JSMA**
ROC AUC↑	86.74	77.35	82.75
AUPRC↑	77.76	71.40	78.19
FPR80↓	24.13	56.21	17.28

It is especially remarkable that they also tend to land to the holes in the VAE latent representation since they are detected based on the results of the hole indicator. Such a phenomenon may be explained by the similarity of internal representation within DNNs that are trained on the same datasets.

As can be observed from the results, the best values are achieved for the FGSM adversarial inputs, which result in a higher standard deviation of the log-likelihoods, leading to better detection. It seems not surprising, taking into consideration that FGSM does not aim at an optimal attack but the fastest one. CW, on the contrary, represents the least uncertainty, which also can be explained by the fact that this attack exploits the optimization procedure with the appropriate objective of as few modifications as possible to the input. JSMA is located somewhere in-between FGSM and CW.

We proceed to evaluate the robustness of the proposed VAE filter by subjecting it to adversarial attacks designed explicitly for this model. We put under test a single VAE. The model is enforced with a controlled continuity on the encoder map taking into consideration appropriate properties of compactness of the latent image.

Table 5.6: Generative adversarial examples

Metric	**Lipschitz MNIST: MNIST vs Adversarial**	**Lipschitz FMNIST: FMNIST vs Adversarial**	**Lipschitz MNIST heldout: MNIST 01 vs Adversarial**
ROC AUC↑	97.89	93.40	99.98
AUPRC↑	98.70	94.51	99.98
FPR80↓	9.06	9.10	0.00

As it can be seen from Table 5.6, the hole indicator successfully detects attacks on VAEs. It allows using only one score to detect both outliers and adversarial examples, including discriminative and generative ones.

Following this, we apply our algorithm based on active defense to distinguish between the outliers and both types of adversarial examples. Since the major value responsible for this distinguishing is based on MSS-SIM gain, we register the corresponding values in Tables 5.7 and 5.8. It can be observed that generative adversarial examples can be easily discerned from the rest categories of problematic inputs. However, there is no possibility to delimit outlier and discriminative adversarial attacks relying only on the MSSSIM gain.

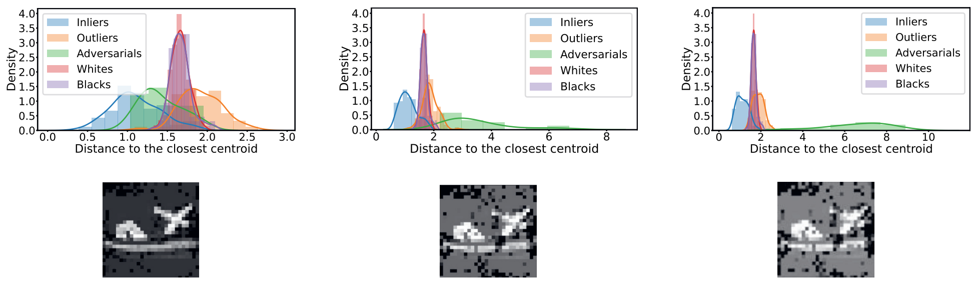
Table 5.7: MNIST: Multi-Scale Structural Similarity (MSSSIM)

	No HMC	**HMC**	**MSSSIM Gain**
	Discriminative Adversarial Examples		
MNIST FGSM $\varepsilon = 0.1$	0.43	0.34	0.09
MNIST FGSM $\varepsilon = 0.3$	0.26	0.27	0.01
MNIST CW	0.18	0.20	0.02
	Generative Adversarial Examples		
MNIST $\varepsilon = 0.1$	0.43	0.85	**0.42**
MNIST $\varepsilon = 0.2$	0.30	0.67	**0.37**
MNIST $\varepsilon = 0.3$	0.25	0.64	**0.39**
	Outliers		
MNIST vs FMNIST	0.03	0.09	0.06
MNIST vs KMNIST	0.21	0.16	0.05
MNIST vs All White	0.03	0.10	0.07

Table 5.8: FMNIST: Multi-Scale Structural Similarity (MSSSIM)

	No HMC	HMC	MSSSIM Gain
	Discriminative Adversarial Examples		
FMNIST FGSM $\varepsilon = 0.1$	0.28	0.17	0.11
FMNIST FGSM $\varepsilon = 0.3$	0.19	0.24	0.05
FMNIST CW	0.33	0.26	0.07
	Generative Adversarial Examples		
FMNIST $\varepsilon = 0.1$	0.41	0.60	**0.19**
FMNIST $\varepsilon = 0.2$	0.25	0.45	**0.20**
FMNIST $\varepsilon = 0.3$	0.19	0.38	**0.19**
	Outliers		
FMNIST vs MNIST	0.18	0.23	0.05
FMNIST vs KMNIST	0.20	0.19	0.01
FMNIST vs All White	0.21	0.17	0.04

Finally, we visualize how the different levels of attack strength influence the location of adversarial latent codes within the learned data representation. This location is calculated with respect to the closest centroid of the cluster to the corresponding adversarial latent code. As can be observed in Figure 5.2, the stronger the attack, the farther the corresponding latent codes drift away from the inlier manifold. Note that a weak adversarial attack is akin to the near-OoD instance, and a strong attack is akin to the far-OoD input.



(a) $\varepsilon = 0.1$ (b) $\varepsilon = 0.3$ (c) $\varepsilon = 0.5$

Figure 5.2: **From left to right**: The strength of FGSM attack, expressed by the magnitude of perturbations. **Top:** Distances to the closest centroid within the latent manifold for various categories of inputs. **Bottom:** Examples of a particular Fashion-MNIST instance that undergoes the corresponding strength of an attack.

5.5. DISCUSSION

The hole indicator confirms that transferability extends from discriminative to generative models, indicating a similar learned representation between these two approaches. However, even though adversarial examples from the discriminative model end up in the latent holes of the VAE, the active defense through HMC cannot return to the regions with high probability. This suggests that despite some commonalities, significant differences still exist between discriminative and generative settings.

Adversarial attacks on the VAE's latent space can be effectively distinguished from OoD inputs using active defense strategies. Furthermore, the internal latent representations of near- and far-OoD instances are similar to those of weak and strong adversarial attacks, respectively.

Contrary to common belief, Bayesian inference over DNN parameters is not essential for sensitivity analysis. We observe different levels of model stability with respect to inliers versus outliers, which is related to the differences in the variances of log-likelihoods, revealing a connection with the recently introduced score of the hole indicator.

5.6. CONCLUSION

We explore two common types of problematic inputs in DNN classifiers: OoDs and adversarial attacks. Our proposed solution uses a variational autoencoder (VAE) to address both problems simultaneously. We initially evaluate the effectiveness of using Bayesian estimation of epistemic uncertainty from VAE weights to detect OoD inputs and discover that comparable results can be achieved by importance sampling with classical VAE formulations without resorting to Bayesian inference over weights. This result indicates that latent codes possess all the necessary information for measuring a model's sensitivity. Furthermore, we introduce a simple algorithm that distinguishes generative adversarial examples from both outliers and discriminative adversarial attacks using active defense. It enables identifying if the VAE model is currently being under attack. In addition, this algorithm allows for detecting both types of adversarial attacks: one is based on the imperceptible perturbations of the input image to the classifier, and it is based on the transferability of the adversarial examples from discriminative to generative models, while another is based on the attacks aimed at the encoder of the VAE. Finally, our approach allows a VAE model to be pretrained on specific datasets so that it functions as a filter, serving the purpose of protecting the DNN classifier from potential attacks and OoD inputs. This pre-trained VAE can be easily integrated as a filtering engine with any DNN image classifier, regardless of its architecture, trained on the same dataset, eliminating the need for further training or modification of internal DNN configurations.

5.7. SUPPLEMENTARY MATERIALS

5.7.1. CLASSICAL VAE'S OVERCONFIDENCE

As it was demonstrated by Nalisnick et al. in [6], all of the DGMs suffer from the overconfidence while trying to estimate the density of the out-of-distribution data assigning a higher density to the OoD inputs in comparison with ID data. We observed such an overconfidence during our experiments as well. A couple of examples of the overconfidence of the classical VAEs in our experimental setup can be seen in Figure 5.3.

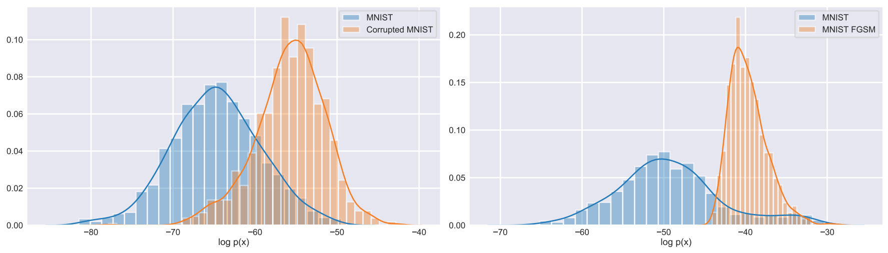


Figure 5.3: **Left:** Log-likelihoods for MNIST as in-distribution (blue) vs Corrupted MNIST as out-of-distribution (orange). **Right:** Log-likelihoods for MNIST as in-distribution (blue) vs MNIST FGSM attacks as out-of-distribution (orange).

5.7.2. Bayesian VAES VARIATION SCORING FOR THE REST DATASETS

We ran out experiments also for MNIST as in-distrubtion vs Fashion-MNIST as OoD and for SVHN as in-distribution and CIFAR-10 as OoD. The results can be seen in Table 5.9 and Table 5.10.

Table 5.9: Scoring values across all types of *Bayesian* VAEs trained on MNIST data and tested on Fashion-MNIST as OoD

	MNIST vs. Fashion-MNIST								
	BBB			**SGHMC**			**SWAG**		
	ROC AUC↑	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**
Expected LL	99.98	99.98	0.00	99.93	99.92	0.04	**96.83**	**96.20**	**5.18**
WAIC	99.99	99.99	0.00	**99.94**	**99.94**	0.02	80.37	76.25	33.56
Disagreement score	98.95	99.01	0.23	97.32	97.70	1.37	94.88	93.97	8.99
Entropy (ours)	99.42	99.47	0.02	98.50	98.75	0.29	95.72	95.20	8.37
Stds of LLs (ours)	**99.99**	**99.99**	**0.00**	99.91	99.91	**0.00**	80.37	82.78	39.12

5.7.3. HAMILTONIAN MONTE CARLO ALGORITHM

We employ the same approach as suggested in [23].

In the Hamiltonian Monte Carlo (HMC) framework, the target distribution is given by the product of $p(\mathbf{x}|\mathbf{z})$ and $p(\mathbf{z})$. The Hamiltonian rep-

Table 5.10: Scoring values across all types of *Bayesian* VAEs trained on SVHN data and tested on CIFAR-10 as OoD

	SVHN vs. CIFAR-10								
	BBB			**SGHMC**			**SWAG**		
	ROC AUC↑	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**
Expected LL	58.65	61.79	77.72	57.09	60.56	80.18	58.98	62.06	76.52
WAIC	64.46	66.01	68.39	62.17	64.38	72.45	62.84	68.42	75.25
Disagreement score	85.20	88.35	30.26	85.31	88.52	28.66	77.58	80.36	45.60
Entropy (ours)	87.80	90.63	20.77	87.89	90.76	19.91	**80.01**	**83.24**	**41.58**
Stds of LLs (ours)	**93.29**	**91.51**	**10.99**	**94.70**	**93.95**	**8.67**	59.31	53.36	61.78

resents the energy of the combined distribution of **z** and the auxiliary variable **p**, defined as follows:

$$H(\mathbf{z},\mathbf{p}) = U(\mathbf{z}) + K(\mathbf{p}),$$

where

$$U(\mathbf{z}) = -\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) - \log p(\mathbf{z}),$$

and

$$K(\mathbf{p}) = -\frac{1}{2}\mathbf{p}^T\mathbf{p}.$$

For the corresponding pseudocode for restoring the latent code, please see following Aglorithm 3.

Algorithm 3: A single iteration of HMC

```
Input: $\mathbf{z}, \eta, L$
// Sample the auxiliary variable
$\mathbf{p} \sim \mathcal{N}(\mathbf{0}, \mathbb{I})$
$\mathbf{z}^{(0)} := \mathbf{z}, \mathbf{p}^{(0)} := \mathbf{p}$
// Make $L$ steps of leapfrog
for $l = 1$ to $L$ do
    $\mathbf{p}^{(l)} := \mathbf{p}^{(l-1)} - \frac{\eta}{2}\nabla_{\mathbf{z}} U(\mathbf{z}^{(l-1)})$
    $\mathbf{z}^{(l)} := \mathbf{z}^{(l-1)} + \eta\nabla_{\mathbf{p}} K(\mathbf{p}^{(l)})$
    $\mathbf{p}^{(l)} := \mathbf{p}^{(l)} - \frac{\eta}{2}\nabla_{\mathbf{z}} U(\mathbf{z}^{(l)})$
// Accept new point with probability $\alpha$
$\alpha := \min\left(1, \exp\left(-H(\mathbf{z}^{(L)}, \mathbf{p}^{(L)}) + H(\mathbf{z}^{(0)}, \mathbf{p}^{(0)})\right)\right)$
$\mathbf{z} := \begin{cases} \mathbf{z}^{(L)} & \text{with probability } \alpha, \\ \mathbf{z}^{(0)} & \text{otherwise.} \end{cases}$
return $\mathbf{z}$
```

5.7.4. CLASSICAL VAES VARIATION SCORING FOR THE REST DATASETS

Table 5.11: Scoring values for the classical VAEs trained on MNIST and Fashion-MNIST data

	Classical VAE					
	MNIST vs. Fashion-MNIST			**Fashion-MNIST vs. MNIST**		
	ROC AUC↑	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**
Expected LL	**99.97**	**99.97**	**0.00**	46.72	51.54	92.57
WAIC	99.96	99.96	0.00	64.07	64.43	66.98
Disagreement score	97.86	98.09	1.11	96.83	97.56	0.84
Entropy (ours)	98.67	98.84	0.38	98.18	98.63	**0.08**
Stds of LLs (ours)	99.81	99.82	0.00	**99.68**	**99.64**	0.36

REFERENCES

[1] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. *Intriguing properties of neural networks*. 2013. arXiv: 1312.6199 [cs.CV].

[2] I. J. Goodfellow, J. Shlens, and C. Szegedy. *Explaining and Harnessing Adversarial Examples*. 2014. arXiv: 1412.6572 [stat.ML].

[3] N. Carlini and D. Wagner. *Towards Evaluating the Robustness of Neural Networks*. 2016. arXiv: 1608.04644 [cs.CR].

[4] A. Nguyen, J. Yosinski, and J. Clune. "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images". In: *IEEE conference on computer vision and pattern recognition*. 2015, pp. 427–436.

[5] D. Hendrycks and K. Gimpel. "A baseline for detecting misclassified and out-of-distribution examples in neural networks". In: *arXiv preprint arXiv:1610.02136* (2016).

[6] E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, and B. Lakshminarayanan. "Do deep generative models know what they don't know?" In: *arXiv preprint arXiv:1810.09136* (2018).

[7] S. Hess, W. Duivesteijn, and D. Mocanu. *Softmax-based Classification is k-means Clustering: Formal Proof, Consequences for Adversarial Attacks, and Improvement through Centroid Based Tailoring*. 2020. arXiv: 2001.01987 [cs.LG].

[8] H. Zhang and J. Wang. "Defense against adversarial attacks using feature scattering-based adversarial training". In: *Advances in Neural Information Processing Systems*. 2019, pp. 1831–1841.

[9] S. Hu, T. Yu, C. Guo, W.-L. Chao, and K. Q. Weinberger. "A new defense against adversarial images: Turning a weakness into a strength". In: *Advances in Neural Information Processing Systems*. 2019, pp. 1635–1646.

[10] P. Samangouei, M. Kabkab, and R. Chellappa. "Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models". In: *CoRR* abs/1805.06605 (2018). arXiv: 1805.06605. url: http://arxiv.org/abs/1805.06605.

[11] D. Meng and H. Chen. "MagNet: a Two-Pronged Defense against Adversarial Examples". In: *CoRR* abs/1705.09064 (2017). arXiv: 1705.09064. url: http://arxiv.org/abs/1705.09064.

[12] U. Hwang, J. Park, H. Jang, S. Yoon, and N. I. Cho. *PuVAE: A Variational Autoencoder to Purify Adversarial Examples*. 2019. arXiv: 1903.00585 [cs.LG].

[13] E. Daxberger and J. M. Hernández-Lobato. "Bayesian variational autoencoders for unsupervised out-of-distribution detection". In: *arXiv preprint arXiv:1912.05651* (2019).

[14] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. Depristo, J. Dillon, and B. Lakshminarayanan. "Likelihood ratios for out-of-distribution detection". In: *Advances in neural information processing systems* 32 (2019).

[15] D. Hendrycks, M. Mazeika, and T. Dietterich. "Deep anomaly detection with outlier exposure". In: *arXiv preprint arXiv:1812.04606* (2018).

[16] K. Lee, H. Lee, K. Lee, and J. Shin. *Training Confidence-calibrated Classifiers for Detecting Out-of-Distribution Samples*. 2017. arXiv: 1711.09325 [stat.ML].

[17] M. Glazunov and A. Zarras. "Do Bayesian Variational Autoencoders Know What They Don't Know?" In: *Conference on Uncertainty in Artificial Intelligence (UAI)*. 2022.

[18] M. Glazunov and A. Zarras. "Vacant Holes for Unsupervised Detection of the Outliers in Compact Latent Representation". In: *Uncertainty in Artificial Intelligence (UAI)*. 2023.

[19] K. Lee, K. Lee, H. Lee, and J. Shin. "A simple unified framework for detecting out-of-distribution samples and adversarial attacks". In: *Advances in neural information processing systems* 31 (2018).

[20] N. A. Ahuja, I. Ndiour, T. Kalyanpur, and O. Tickoo. *Probabilistic Modeling of Deep Features for Out-of-Distribution and Adversarial Detection*. 2019. arXiv: 1909.11786 [stat.ML].

[21] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli. "Evasion Attacks Against Machine Learning at Test Time". In: *Machine Learning and Knowledge Discovery in Databases*. 2013.

[22] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. "The Limitations of Deep Learning in Adversarial Settings". In: *2016 IEEE European Symposium on Security and Privacy (EuroS P)*. 2016, pp. 372–387.

[23] A. Kuzina, M. Welling, and J. M. Tomczak. "Alleviating adversarial attacks on variational autoencoders with MCMC". In: *Proceedings of the 36th International Conference on Neural Information Processing Systems*. NIPS 22. New Orleans, LA, USA: Curran Associates Inc., 2024. isbn: 9781713871088.

[24] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. *Practical Black-Box Attacks against Machine Learning*. 2016. arXiv: 1602.02697 [cs.CR].

[25] S. Liang, Y. Li, and R. Srikant. “Enhancing the reliability of out-of-distribution image detection in neural networks”. In: *arXiv preprint arXiv:1706.02690* (2017).

[26] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman. *PixelDefend: Leveraging Generative Models to Understand and Defend against Adversarial Examples*. 2017. arXiv: 1710.10766 [cs.LG].

[27] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. “Equation of State Calculations by Fast Computing Machines”. In: *The Journal of Chemical Physics* 21.6 (1953), pp. 1087–1092.

[28] W. K. Hastings. “Monte Carlo Sampling Methods Using Markov Chains and Their Applications”. In: *Biometrika* 57.1 (1970), pp. 97–109. issn: 00063444. url: http://www.jstor.org/stable/2334940.

[29] J. Heinbockel. *Introduction to the Variational Calculus*. Trafford Publishing, 2007. isbn: 9781425103521. url: https://books.google.fr/books?id=Ty9zAgAACAAJ.

[30] D. P. Kingma and M. Welling. *Auto-Encoding Variational Bayes*. 2013. arXiv: 1312.6114 [stat.ML].

[31] C. Bishop. “Latent Variable Models”. English. In: *Learning in Graphical Models*. Ed. by M. Jordan. Adaptive Computation and Machine Learning. MIT Press, Jan. 1999, pp. 371–403. isbn: 9780262600323.

[32] C. Doersch. *Tutorial on Variational Autoencoders*. 2016. arXiv: 1606.05908 [stat.ML].

[33] D. J. Rezende, S. Mohamed, and D. Wierstra. “Stochastic backpropagation and approximate inference in deep generative models”. In: *International conference on machine learning*. PMLR. 2014, pp. 1278–1286.

[34] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. “Weight uncertainty in neural network”. In: *International conference on machine learning*. PMLR. 2015, pp. 1613–1622.

[35] Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou. “Variational Deep Embedding: An Unsupervised and Generative Approach to Clustering”. In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*. 2017, pp. 1965–1972. doi: 10.24963/ijcai.2017/273.

[36] Y. LeCun and C. Cortes. *MNIST handwritten digit database*. http://yann.lecun.com/exdb/mnist/. 2010.

[37] H. Xiao, K. Rasul, and R. Vollgraf. “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms”. In: *arXiv preprint arXiv:1708.07747* (2017).

[38] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. “Reading Digits in Natural Images with Unsupervised Feature Learning”. In: *Advances in neural information processing systems (NIPS)*. 2011.

[39] A. Krizhevsky, V. Nair, and G. Hinton. *Cifar-10 (canadian institute for advanced research)*. http://www.cs.toronto.edu/kriz/cifar.html. 2010.

[40] F. Chollet *et al. Keras*. https://keras.io. 2015.

[41] N. Papernot, F. Faghri, N. Carlini, I. Goodfellow, R. Feinman, A. Kurakin, C. Xie, Y. Sharma, T. Brown, A. Roy, A. Matyasko, V. Behzadan, K. Hambardzumyan, Z. Zhang, Y.-L. Juang, Z. Li, R. Sheatsley, A. Garg, J. Uesato, W. Gierke, Y. Dong, D. Berthelot, P. Hendricks, J. Rauber, and R. Long. “Technical Report on the CleverHans v2.1.0 Adversarial Examples Library”. In: *arXiv preprint arXiv:1610.00768* (2018).

6 CONCLUDING REMARKS

DNNs utilized in safety-critical tasks are highly susceptible to problematic inputs, like OoD inputs, or adversarial attacks. A core part of this dissertation is dedicated to understanding and addressing the issue of model robustness against these problematic inputs. We began by exploring the concept of learnability and generalization within machine learning, particularly by examining the generalization of models to non-identically distributed samples and the formal challenges it poses. Recognizing the impossibility of OoD learnability in general and taking into consideration that good generalization to inliers is indicative of outliers detection, we identified the notion of stability of the learning algorithm as a key ingredient for good generalization with respect to i.i.d. data. Based on this insight, we formulated a dual problem of stability with respect to the parameters of the model.

As for our modeling assumptions, we choose the DGM approach due to its potential to learn the inlier data distribution. It is beneficial from the perspective of features learned, contrasting it with discriminative modeling primarily interested in boundaries between the categories. In particular, we focused on VAEs since they allow for the setting of a desirable inductive bias from various perspectives ranging from information bottleneck compression to PAC-Bayes learnability.

Consequently, we concentrated on two key aspects: epistemic uncertainty estimation and tailored inductive bias to address the issue of detection of problematic inputs. We employed these tools as operational means to either estimate the degree of sensitivity of a model regarding inliers versus outliers or to improve model stability. Both of the considered aspects allow for the effective handling of evaluated problematic inputs.

Finally, we formulated a novel approach to improving the robustness of deep learning models based on their latent representation. The major revelation is that both problematic inputs tend to gravitate toward the vacant holes in the latent space that could be easily identified. As a result, we devised a defensive method addressing both OoD and adversarial input mitigation and, in many practical scenarios, detection of ongoing adversarial attacks.

In this thesis, we explored various aspects of deep learning, focusing on understanding and addressing vulnerabilities that can compromise the performance and reliability of DNNs. Considering the growing prevalence of DNNs in critical applications, it is necessary to ensure that these models are resilient to potential threats. The following sections discuss the basic issues of robustness, sensitivity, and latent representation of problematic inputs in DNNs. Each section is structured around the key research questions of our study, starting with the fundamental aspect of deep learning robustness.

6.1. ROBUSTNESS IN DEEP LEARNING

Robustness in deep learning is a critical area of research, especially considering the growing use of DNNs in safety-critical applications. The reliability of these models can be significantly reduced by various types of problematic inputs, such as adversarial attacks and outliers, which can lead to model misbehavior or unauthorized data extraction. In this section, we present our findings related to the categorization of these problematic factors and corresponding defenses in the domain of deep learning pertaining to RQ1:

RQ1: *How can we categorize problematic inputs in the context of attacking and defensive methods in deep learning?*

In Chapter 2, we explored in detail the resilience of DNNs to various threats, emphasizing the need for a better understanding of their vulnerabilities to ensure their reliability in safety-critical applications. Our investigation included a comprehensive analysis of attack vectors targeting DNNs. We identified two major categories within the diverse spectrum of malicious activities related to DNN models which we dubbed as functionality-oriented and privacy-oriented attacks. The former category seeks to manipulate the model's behavior, whereas the latter attempts to extract any private data related to a particular model.

Based on this coarse categorization, it becomes apparent that both adversarial examples and outliers fall under the category of functionality-oriented attacks. This category also includes poisonous inputs. The difference between malicious inputs considered in this thesis and poisonous inputs is that the latter aims to manipulate the behavior of the model or violate the integrity of the data during the training stage, while the former interferes with the inference stage. Privacy-oriented attacks, including model theft and membership discovery, aim to unauthorizedly extract sensitive information related to the model's architecture or the data on which it was trained.

In addition, our study emphasizes the potential adversarial aspect of outliers by categorizing them within the CIA triad along with other threats.

As for defensive mechanisms against problematic inputs, there are two prominent approaches: adversarial training to mitigate adversarial attacks and outlier exposure training to enhance DNN robustness against outliers. Despite the difference in details, they are based on the same underlying principle: to incorporate knowledge about these problematic inputs into the training phase. It can be done either directly by adding new instances into the training set, as is the case with outlier exposure, or via modifying an objective function, as is the case with adversarial training.

LIMITATIONS AND FUTURE WORK

The identified subset of potential vulnerabilities is limited to the inference stage. Although it was a deliberate decision due to the high criticality of the exposure of DNNs to errors during testing in the wild, it still represents a limitation. Nevertheless, the attacks related to the training stage could be curated much more efficiently. They may involve additional preprocessing steps with a potential subsequent automatic or manual verification. Such a privilege is not available during inference when the model is already deployed in the wild.

For future work, it is critical to develop a theoretical framework for DNNs that will not only improve our understanding of their generalization and optimization but also contribute to creating more robust architectures. In addition, a promising direction is the exploration of new defense strategies, especially those that exploit the interaction between data and model properties. Such an approach could be beneficial for detecting and neutralizing potential threats. Additionally, increasing the transparency and interpretability of DNNs can play a critical role in identifying and mitigating vulnerabilities, and strengthening their defenses against current and emerging threats.

6.2. SENSITIVITY OVER HYPOTHESES

Detecting problematic inputs in deep learning models is an important task that has serious implications for the reliability and robustness of these models, especially in safety-critical applications. One promising way to address this problem is to estimate epistemic uncertainty, which can provide insight into a model's confidence in its predictions. In this section, we explore the potential of using epistemic uncertainty to answer RQ2:

RQ2: *Is detecting problematic inputs utilizing epistemic uncertainty estimation possible?*

To answer this question, we implemented three distinct Bayesian approaches to infer a posterior over VAE parameters. The key observation is that our assumption about different degrees of sensitivity of parameters of VAEs with respect to inliers versus outliers holds true in practice. Namely, the stability of the model is lower with respect to OoDs. For that reason, we devised two corresponding scores based on information entropy and sample standard deviation with respect to the parameters of the model. Both of these scores demonstrated state-of-the-art results in detecting outliers.

Another key takeaway comes from comparing our scores with other scores devised for the ensembles. The WAIC score is of particular interest since it also involves a term responsible for variance of the likelihoods. Surprisingly, we identified a significant difference between our scores and WAIC in performance: while our scores demonstrated state-of-the-art performance along all considered datasets, WAIC, on the other hand, achieved good results on simple datasets like MNIST and failed on more complex datasets like CIFAR-10. Training models on more sophisticated data results in a wider range of likelihoods being attributed to the inputs; as a consequence, the difference between the models within an ensemble may disappear and would be hidden behind the input likelihood variability. This phenomenon is entirely overlooked by metrics like WAIC. Conversely, our proposed metrics focus on evaluating the variance observed within the ensemble itself. This approach enables the detection of even minimal variations in the sensitivity capturing the model's instability.

LIMITATIONS AND FUTURE WORK

There are three distinct limitations to this approach. First, Bayesian approaches, especially BBB and SGHMC, require significantly longer training time to converge compared to conventional VAEs. This may limit their practical applications for large-scale solutions or when rapid model development is strictly required.

Second, the relative complexity of the suggested methods since the implementation of Bayesian methods for estimating epistemic uncertainty is more complex than traditional methods which may potentially prevent their widespread adoption without further simplification or optimization.

Finally, there is a distinct speed-accuracy trade-off. Although SWAG provides faster training convergence time, it generally provides lower OoD detection performance compared to other Bayesian methods tested.

For future work, in Chapter 4, we elaborated on a similar approach based on the latent codes of VAE. It significantly increases training performance. We also dived deeper into the reasons behind this sensitivity.

6.3. LATENT REPRESENTATIONS OF PROBLEMATIC INPUTS

Understanding hidden representations of problematic inputs, such as OoD inputs and adversarial examples, is necassary to improving the robustness of deep learning models. Latent space analysis provides insight into how these inputs differ from normal inputs and offers a path for developing effective detection mechanisms. This section addresses research question RQ3 by exploring how problematic inputs can be interpreted through their internal hidden representations, and using this understanding to improve the robustness of the model:

RQ3: *How problematic inputs can be interpreted from the perspective of their internal latent representation?*

First, based on the sensitivity of VAE to different inputs, we have discovered that both OoDs and adversarial examples tend to gravitate to latent holes. These holes are low-density regions in the aggregated posterior of the latent space. We devised a special score to detect these holes based on two main ideas: the compressive nature of the latent representation and the fact that low-density regions occupy space between distinct inlier posteriors corresponding to the input space's different instances. Sampling from such a posterior would result in higher variability of the likelihoods for problematic inputs in comparison with inliers, providing a simple and robust way of their detection.

Second, we enforced controllable compactness on the mapped image of the encoder in the latent space in order to both limit the potential variability of the mapping and squeeze the available holes within the manifold available for the inliers. It achieved greater detection precision and provably regularized the continuity of the mapping.

Third, we analyzed the behavior of the two distinct categories of adversarial examples: discriminative and generative. Both of these categories can be detected through their latent representation using a hole indicator. Moreover, the attacks that are based on the imperceptible perturbation of the image tend not to drift far away from the original reference latent posterior. Thus it can be successfully distinguished from both the outliers and the strong adversarial attacks by utilizing HMC and comparing MSSSIM gain between the adversarial and restored images. Additionally, the modification of VAE objective to incorporate the mixture of Gaussians in the latent manifold instead of a standard Normal prior to getting the centroids of learned clusters in the latent space allows visualization of the location of latent codes of adversarial attacks depending on the strength. We detect that maximum damage adversarial examples tend to gravitate away from the learned manifold. Their distinguishing from the outlier is an open problem since they are similar to the far-OoD inputs.

LIMITATIONS AND FUTURE WORK

The main limitation concerns the strong cases of adversarial attacks, i.e., when a latent point moved away sufficiently far from being restored with HMC. These cases currently represent an unsolved problem and can be analyzed in future works. Another drawback relates to the speed performance of the algorithm based on active defense. Since the HMC needs time to restore the posterior to the regions with high probability, it may be unfit for real-time problematic inputs distinguishing into separate categories.

6.4. RESULTS

In summary, we achieved the following results:

- We categorized outliers within the diverse spectrum of deep learning vulnerabilities, stressing the possibility of exploiting them in attacks in the same manner as untargeted adversarial examples to cause a potential malfunctioning of the model.
- We identified the stability of the variational autoencoder during inference that is measured via sensitivity over its parameters as a robust and effective indicator for outliers detection.
- We implemented three different Bayesian inference methods and estimated the parameters' variation within the ensemble, obtaining state-of-the-art results.
- We disentangled Bayesian inference over parameters and Bayesian inference over latent codes, demonstrating the effective ways of measuring the sensitivity of the model with respect to the latent representation.
- We discovered that the source of the successful detection by means of sensitivity comparison between inliers and problematic inputs lies in the fact that the latter tends to gravitate to vacant holes in the latent representation of the VAE.
- We employed the discovery of latent holes to dissect the variation of the log-likelihood, and based on that, we devised a new score to detect problematic inputs.

- We addressed a persistent theoretical flaw in the modeling assumptions of VAEs by enforcing compactness on the latent space in two different manners, which is validated as being advantageous for OoD and adversarial examples detection.
- We addressed the encoder mapping's variation range, employing the stricter continuity condition based on the Lipschitz constant. We devised a way to properly constrain the continuity taking into account the properties of the resulting image and the dimensionality of the corresponding latent space.
- We developed methods to discern generative adversarial examples from both outliers and discriminative adversarial attacks.
- We developed a filter solution based on the VAE trained on the same dataset as the classifier to be protected and enhance the robustness during the inference stage, allowing for the detection of the problematic inputs and also mitigation of the classical definition of the adversarial examples.

6.5. SOCIETAL RELEVANCE

DNNs play a critical role in a variety of applications that impact society. These range from diagnostic tools in medicine that can detect early signs of disease to manufacturing precision and safety and the reliability of autonomous cars. The social impact is profound. These technologies have the potential to save lives, improve efficiency, and reduce human error. However, this promise depends on the DNNs' ability to accurately interpret and respond to the diverse range of its potential inputs. This ability is currently undermined when it encounters OoD and adversarial inputs. Potential threats may undermine public trust. As society becomes increasingly dependent on DNNs, the integrity of these systems becomes very important. Lack of confidence in the accuracy and reliability of DNN results can delay the adoption of useful technologies and hinder progress.

Prior to the advances presented in this thesis, DNN were vulnerable to various types of problematic inputs, such as adversarial attacks and OoD data, which could compromise their functionality and reliability. These vulnerabilities pose significant risks in safety-critical applications, leading to potential failures in medical diagnostics, autonomous vehicles, and other applications where accuracy and reliability are paramount. The lack of robust mechanisms to identify and mitigate these factors has undermined confidence in these systems and slowed their integration into critical industries.

The results of this thesis directly address these vulnerabilities by developing robust detection capabilities for both OoD and adversarial inputs.

By deepening the understanding of how these problematic inputs interact with the latent space of VAEs and implementing advanced Bayesian approaches, this research proposes effective strategies for detecting and mitigating such inputs. This results in a significant increase in the reliability and stability of DNNs.

The social implications of these advances are significant. With improved threat detection and mitigation, DNNs can be deployed to security-critical applications with greater confidence. This means more reliable medical diagnoses, safer autonomous vehicles and generally more reliable AI systems across various sectors. The ability to accurately identify and respond to problematic input reduces the risk of system failures and increases public trust in AI technologies. This in turn speeds up the adoption of these technologies, leading to broader societal benefits such as improved health outcomes, increased production efficiency and safer transport systems.

By addressing these issues, we can safely use the power of DNNs. The future of deep learning depends not only on the experts who create it but on all of those who use it and who are influenced by it.

6.6. CONCLUSION

This thesis offers valuable insights and contributions into the field of DNN robustness, which are particularly relevant for safety-critical tasks. It focuses on notoriously problematic inputs for modern deep learning models, including OoD instances and adversarial attacks. Through a comprehensive analysis of learnability, generalization, and challenges inherent to OoD learnability, this dissertation identified the stability of learning algorithms as the key ingredient to both problematic input detection and also to improving model robustness. Focusing on DGM, in particular on VAE, this thesis presents a new approach based on the idea of measuring different degrees of sensitivity of a model with respect to different inputs. Moreover, this dissertation thoroughly examines and implements the desirable topological properties in VAE models from the perspective of better detection and mitigation of problematic inputs. The insights gained from studying the latent space of VAE, especially identifying vacant latent holes as regions where problematic inputs tend to gravitate, have led to the development of effective strategies for detecting and mitigating adversarial and OoD inputs.

This work acknowledges limitations, such as the impossibility of OoD learning in the discriminative case, and the need for further research into the cases of strong adversarial attacks. In addition, a suggested separate defensive filter solution may not always be feasible in practice due to time constraints. Future work may aim to improve the current state and discover new security strategies exploiting different DNN architectures.

The societal ramifications of this thesis are profound. DNNs play a crucial role in many applications that significantly impact society, from healthcare to autonomous driving. The reliability and safety of these systems depend on their ability to accurately process and respond to a wide range of input data. The focus of this thesis on developing robust DNNs for detecting and managing OoD and adversarial inputs is a step forward in ensuring the integrity and reliability of these systems.

In conclusion, this thesis advances our understanding of the reasons why DNNs and VAEs are susceptible to problematic inputs, and it also provides the means to significantly enhance robustness in ensuring the secure and efficient deployment of DNNs in real-world applications.

BIBLIOGRAPHY

[1] M. **Glazunov** and A. Zarras. “Who Guards the Guardians? On Robustness of Deep Neural Networks”. In: *Artificial Intelligence for Security: Enhancing Protection in a Changing World*. Ed. by T. Sipola, J. Alatalo, M. Wolfmayr, and T. Kokkonen. Cham: Springer Nature Switzerland, 2024, pp. 103–127. isbn: 978-3-031-57452-8.

[2] M. **Glazunov** and A. Zarras. “Vacant holes for unsupervised detection of the outliers in compact latent representation”. In: *Proceedings of the Thirty-Ninth Conference on Uncertainty in Artificial Intelligence*. Ed. by R. J. Evans and I. Shpitser. Vol. 216. Proceedings of Machine Learning Research. PMLR, 2023, pp. 701–711. url: https://proceedings.mlr.press/v216/glazunov23a.html.

[3] M. **Glazunov** and A. Zarras. “Do Bayesian variational autoencoders know what they don’t know?” In: *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*. Ed. by J. Cussens and K. Zhang. Vol. 180. Proceedings of Machine Learning Research. PMLR, 2022, pp. 718–727. url: https://proceedings.mlr.press/v180/glazunov22a.html.

[4] M. **Glazunov**, A. Aranda, and C. Galuzzi. “Optimal ECG Lead System for Automatic Myocardial Ischemia Detection”. In: *2021 Computing in Cardiology (CinC)*. Vol. 48. 2021, pp. 01–04.

[5] T. Bauer, E. Devrim, M. **Glazunov**, W. L. Jaramillo, B. Mohan, and G. Spanakis. “#MeTooMaastricht: Building a Chatbot to Assist Survivors of Sexual Harassment”. In: *Machine Learning and Knowledge Discovery in Databases*. Ed. by P. Cellier and K. Driessens. Cham: Springer International Publishing, 2020, pp. 503–521.

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to my supervisors, **Apostolis Zarras**, **Inald Lagendijk**, and **David Tax**. Thank you for your dedicated support, frequent feedback, and thought-provoking discussions, which have guided me through both my week-to-week tasks and the more significant milestones of this research. The unique perspectives each of you brings—ranging from practical applicability and robustness of methods to advanced statistical considerations—have repeatedly inspired me to refine my work and broaden my research horizons. Your encouragement to explore new ideas, combined with your openness and willingness to address every question I raise, has played a pivotal role in shaping me into a more confident and resilient researcher.

I would also like to extend my appreciation to the members of my dissertation committee: **Kurt Driessens**, **Sicco Verwer**, **Milan Petkovic**, **Matthijs Spaan**, and **Marcel Reinders**. Thank you for dedicating your time and expertise to reviewing my work and for attesting to the quality and rigor of this thesis. Your valuable feedback and participation have played a pivotal role in shaping the final outcome of my research.

My deepest thanks go to my parents, whose unwavering love and support have guided me throughout my life. **Mom**, that moment when I was eight years old and you took me to your office to show me my very first DOS game—Prince of Persia—was a turning point in my life. Through your example, you taught me the value of facing challenges with positivity, and your love for programming and natural talent in mathematics have continually inspired me. I am also incredibly grateful for your constant encouragement and belief in my abilities, which gave me the confidence to follow my ambitions. Thanks for everything! Sadly, **Dad** could only witness half of my Ph.D. journey before cancer took him from us. Even so, his guidance, unwavering support, and constant motivation to keep pushing forward will always stay with me. Thank you, Dad—you were the best and will always remain an example for me!

I am profoundly thankful to my wife and our son. **Olga**, your unconditional love and support have upheld me throughout this entire Ph.D. journey. I am especially thankful for the sacrifices you made—from long hours to missed family moments—that allowed me to focus on my research. Your unwavering belief in me was often the motivation I needed to keep going, and I thank you for standing by me through every challenge, celebrating every success, and reminding me of what truly mat-

ters beyond academia. **Jan-Willem**, you are only 2.5 years old, but I am grateful for your boundless energy, endless curiosity, and the many laughs you bring into our home. Whether it is playing on my working desk or proudly showing off your newfound skills, you never fail to remind me of what is most important at the end of the day. Thank you for your giggles, spontaneous dance moves, and for being the best little distraction a dad could ask for.

I would also like to express my gratitude to the rest of my family. My elder sister, **Katyusha**, you have consistently been ready to help me in countless situations, showing remarkable patience and resourcefulness whenever I faced a challenge. My brother-in-law, **Andrey**, you have always brought a calm and rational perspective. Discussing complex topics with you was always captivating, especially the clarity with which you shared your point of view. My deepest thanks also go to my mother-in-law, **Liudmila**, for your tremendous help with our newborn, which allowed me to concentrate on my thesis. Your collective support and guidance have been a steady source of strength throughout my academic journey.

I would also like to express my gratitude to my current and former colleagues from the cybersecurity research group. Although my work has largely been remote, and my in-person interactions with many of you have been brief, your support and encouragement have left a lasting impression. In particular, I want to thank **Rui Wang**, one of my closest collaborators, whose expertise and willingness to help have always been much appreciated throughout my work. I also fondly remember the thought-provoking discussions with **Bart Hermans** about BGP and our attempts to optimize it in a more robust way; your insights continue to inspire me. My initial brainstorming sessions with **Marwan Darwish Khabbaz**, before our paths diverged, were invaluable in shaping my early research direction. **Stefanos Koffas**, thank you for not only sharing your ideas for potential collaboration that, unfortunately, never bore fruit, but also for showing me a much-needed cycling route from the outskirts of Delft to the train station! **Marina Krcek**, your assistance right before my Go/NoGo meeting was a true lifesaver—I am especially grateful, given the tight timeline after transferring from another university. **Daniel and Jelle Vos**, thank you for discussing robust machine learning and blockchain prospects. I am deeply grateful to **Georgios Smaragdakis** and **Kaitai Liang** for their belief in me, allowing me to continue my research within the cybersecurity group. Lastly, a special thanks goes to **Sandra Wolff**, whose readiness to help and patience in answering my endless questions have been a constant source of support. Thank you for enhancing my path and making it truly fulfilling.

I would also like to extend my sincere appreciation to **Annette and Joep Koster**, my landlords and now cherished friends. When I first arrived in the Netherlands for my bachelor studies, you were the very first

Dutch people to welcome me into your home. Your kindness and support have accompanied me from undergraduate to Ph.D., and I have always felt incredibly fortunate to have such wonderful local friends. I am deeply grateful for your genuine hospitality, warm guidance, and for making me feel at home even when I was far from my own. Particularly for my Ph.D. defense, you were willing to help and participate, and Joep agreed to be one of my paranymphs! Thank you both for all of that; this dissertation would not have been possible without your steadfast backing.

I would also like to dedicate a separate heartfelt thank you to my best friend, **Philippe Debie**, with whom I began my academic journey during our bachelor studies. Our friendship has flourished ever since, and your positive outlook on life, unwavering scientific rigor, and critical thinking have always guided and inspired me to grow. I am immensely grateful for the countless discussions we have had—covering everything from everyday challenges to complex research questions—which have helped refine my thinking and spark new ideas. Thank you, Philippe, for your enduring encouragement, for reminding me that true curiosity and passion drive any meaningful pursuit, and for kindly agreeing to serve as my paranymph during my Ph.D. defense. Your support has truly meant more than words can express.

There are many more: **Evgeny Kynin**, **Roman Golubev**, **Roman Raikhlen**, **Petr Zaitsev**, **Alfonso Aranda**, **Jop Koster**, **Samantha van Hattum**, **Alex Rousskov**, **Stanislav Gordeev**—all of you helped in one way or another during my journey, and I am very grateful to you for that!

Finally, I am certain there are many others whose support, advice, and encouragement have contributed to this work, even if I have not mentioned each of you by name. I am truly grateful for your help and friendship, and your contributions have not gone unnoticed. Thank you for being part of my journey and making it an experience I will always cherish.

CURRICULUM VITæ

Mikhail Glazunov

02-11-1981	Born in Kirishi, Russia.

SUMMARY

15y experience in industry: 7y as a software engineer, 5y as a tech lead, 3y as a data scientist
4y experience in academia: 4y as a Ph.D. candidate

WORK EXPERIENCE

2021 - now	PhD. candidate Delft University of Technology, the Netherlands
2019 - 2020	Data Scientist Open Pos Oy, Finland
2016 - 2018	Junior Data Scientist Bakken Research Center of Medtronic, The Netherlands
2011 - 2015	Tech Lead Gilbarco Ltd., Russia
2007 - 2011	C++ Software Engineer Autotank Oy, Finland

EDUCATION

2021 - 2024	Ph.D. candidate Delft University of Technology, The Netherlands Supervisor: Prof. Apostolis Zarras Promotor: Prof. dr. ig. Inald Lagendijk Co-promotor: Dr. D.M.J. Tax
2020 - 2021	Ph.D. student Maastricht University, The Netherlands Supervisor: Prof. Apostolis Zarras
2018 - 2020	M.Sc. in Data Science for Decision Making Maastricht University, The Netherlands Supervisor: Prof. Apostolis Zarras
2003 - 2005	M.Sc. in Applied and Mathematical Linguistics Saint-Petersburg State University, Russia Supervisor: Prof. Irina Azarova

LIST OF PUBLICATIONS

BOOK CHAPTERS

1. M. **Glazunov** and A. Zarras. “Who Guards the Guardians? On Robustness of Deep Neural Networks”. In: *Artificial Intelligence for Security: Enhancing Protection in a Changing World*. Ed. by T. Sipola, J. Alatalo, M. Wolfmayr, and T. Kokkonen. Cham: Springer Nature Switzerland, 2024, pp. 103–127. isbn: 978-3-031-57452-8

CONFERENCES

4. M. **Glazunov** and A. Zarras. “Vacant holes for unsupervised detection of the outliers in compact latent representation”. In: *Proceedings of the Thirty-Ninth Conference on Uncertainty in Artificial Intelligence*. Ed. by R. J. Evans and I. Shpitser. Vol. 216. Proceedings of Machine Learning Research. PMLR, 2023, pp. 701–711. url: `https://proceedings.mlr.press/v216/glazunov23a.html`

3. M. **Glazunov** and A. Zarras. “Do Bayesian variational autoencoders know what they don’t know?” In: *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*. Ed. by J. Cussens and K. Zhang. Vol. 180. Proceedings of Machine Learning Research. PMLR, 2022, pp. 718–727. url: `https://proceedings.mlr.press/v180/glazunov22a.html`

2. M. **Glazunov**, A. Aranda, and C. Galuzzi. “Optimal ECG Lead System for Automatic Myocardial Ischemia Detection”. In: *2021 Computing in Cardiology (CinC)*. vol. 48. 2021, pp. 01–04

1. T. Bauer, E. Devrim, M. **Glazunov**, W. L. Jaramillo, B. Mohan, and G. Spanakis. “#MeTooMaastricht: Building a Chatbot to Assist Survivors of Sexual Harassment”. In: *Machine Learning and Knowledge Discovery in Databases*. Ed. by P. Cellier and K. Driessens. Cham: Springer International Publishing, 2020, pp. 503–521

As Deep Neural Networks (DNNs) continue to be deployed in safety-critical domains, two specific concerns — adversarial examples and Out-of-Distribution (OoD) data — pose significant threats to their reliability. This dissertation proposes novel methods to enhance the robustness of deep learning by detecting such inputs and mitigating their impact.

A central insight of this work is that algorithmic stability plays a crucial role in generalizing to in-distribution data. Motivated by this, we formulate a dual perspective on stability with respect to the hypotheses and explore whether this perspective facilitates the separation of problematic inputs under two main lenses: epistemic uncertainty estimation and the choice of an appropriate inductive bias. By grounding our approach in generative modeling with a latent variable based on an information bottleneck and, specifically, employing Variational Autoencoders (VAEs), we first leverage Bayesian inference over model parameters to estimate the model's uncertainty with respect to a particular input. Second, we investigate the required properties of both VAE maps and latent representations from a topological perspective. This reveals how OoD inputs predominantly map onto empty regions — or "holes" — in the latent manifold. Finally, we discover that adversarial examples likewise exhibit similar behavior. This finding is then used to craft new scoring functions that reliably distinguish between inliers, outliers, and adversarial attacks.

TU Delft

9 789465 104782