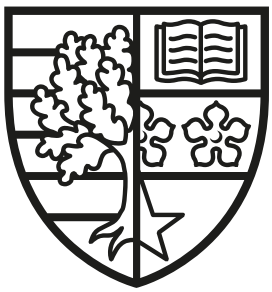
The Influence of Geometric Properties of Data Distributions on Artificial Neural Networks

Daniel Kienitz

Submitted for the degree of
Doctor of Philosophy

Heriot-Watt University



Department of Computer Science,
School of Mathematical and Computer Sciences.

February, 2024

Abstract

Due to recent advances in algorithmic methods, software and hardware, artificial neural networks have reached or even surpassed human-level performance in visual object recognition. Today, they are ubiquitous in real-life applications, such as autonomous vehicles, health care and various industrial settings. This remarkable achievement, however, is clouded by a significant deficit in robustness to distribution shifts, yielding concerns about their readiness to be deployed in safety-critical applications. At their core, artificial neural networks can be viewed as progressively disentangling complex distributions of images to make them suitable for linear separation. This perspective offers a methodology using geometric properties and methods to study their behaviour. In this thesis we use this methodology and study several open questions that lie in the intersection between neural networks' lack of robustness to distribution shifts and the geometric properties of data distributions and representations. First, we study the effect of the three main geometric properties, namely the intrinsic dimension, extrinsic dimension and entanglement, on the sample complexity. Complementary to previous works we show a strong interdependency between intrinsic dimension and entanglement, where the intrinsic dimension only affects the sample complexity if the entanglement of the distribution is high. Further, we show that the entanglement of label-specific distributions is the leading contributor to the sample complexity in general. In the second part, we investigate the geometric complexity of decision boundaries. We show that state-of-the-art robust neural networks learn geometrically more complex decision boundaries than standard ones which confirms a previously made hypothesis and, when combined with the results of the first part, at least partially explains the increased sample complexity of robust training. We also propose an upper bound on the perturbation magnitude over which provably a geometrically more complex decision boundary is required. Further, we show for real-world image benchmarks that our bound also restricts the introduction of label noise. In addition, we show that the commonly used nearest neighbour distance overestimates the robust radius of complex image distributions and that our bound is better suited to estimate the robust radius of these

distributions. In the final part, we compare several different state-of-the-art robust training paradigms and show that dimensionality reduction of their hidden representations is a common mechanism shared amongst them, despite fundamentally different approaches to robust training. We demonstrate that part of this dimensionality reduction is due to sharing of features between semantically similar classes. In summary, we show in this thesis that studying neural networks through the lens of geometric properties yields practical insights into their sample complexity and generalisation behaviour as well as the mechanisms that result in representations that are robust to distributions shifts.

To my mother and my brother.

Acknowledgements

I like to thank my first supervisor Ekaterina Komendantskaya for supporting my application for a PhD scholarship and for giving me the freedom to pursue the topic I was interested in during my studies. Additionally, I am grateful for her support and time in all organizational as well as research related matters. I also like to thank my second supervisor Michael Lones for his help with the work that is presented in this thesis. Further, I like to thank Christian Sämann for offering a different technical perspective on the work in this thesis which has been a valuable help.

Inclusion of Published Works Form

Please note you are only required to complete this form if your thesis contains published works. If this is the case, please include this form within your thesis before submission.

Declaration

This thesis contains one or more multi-author published works. I hereby declare that the contributions of each author to these publications is as follows:

Citation details	D. Kienitz, E. Komendantskaya and M. Lones. The Effect of Manifold Entanglement and Intrinsic Dimensionality on Learning. 36th AAAI Conference on Artificial Intelligence 2022.
D. Kienitz	Proposed the idea, carried out the literature review, conceived and performed the experiments, collected and analysed the data and wrote the paper
E. Komendantskaya	Provided feedback for structure, writing style and suggestions for technical improvements of the paper
M. Lones	Provided feedback for the paper

Citation details	D. Kienitz, E. Komendantskaya and M. Lones. Comparing Complexities of Decision Boundaries for Robust Training: A Universal Approach. Asian Conference on Computer Vision 2022.
D. Kienitz	Proposed the idea, carried out the literature review, conceived and performed the experiments, collected and analysed the data and wrote the paper
E. Komendantskaya	Provided feedback for structure, writing style and suggestions for technical improvements of the paper
M. Lones	Provided feedback for the paper

Please included additional citations as required.

Contents

Chapter 1

Introduction

1.1 The Difficulty of Object Recognition

A fundamental task in human' daily life is the visual recognition of encountered objects in a given scene. Crossing the street, for example, is a process that involves visually distinguishing between several moving and stationary objects. To make these distinctions, humans must learn a *function* that assigns the objects present in the perceived scene to a collection of *labels*, such as *car* or *pedestrian*, for example. This function uses a set of learned object patterns, such as their shape, that are *discriminative features* with respect to different objects.

In real-life no two encountered scenes are the same. Changes in lighting, background or viewing angle do not change the label of the same object but can result in a large number of variations [1] (see Figure 1.1, page 2 for illustration). It is commonly assumed that all different representations of the same object concentrate in the vicinity of a so-called *object-* or *image manifold* (see [2]). Here, a manifold (Definition 1, page 11) refers to a smooth non-linear subspace with an intrinsic dimension that is lower than its ambient space's dimension. The manifold assumption implies that most scene variations encountered in practice are *label-preserving*, so not semantically meaningful, and that distributions of natural images can rather be described by only a few variables. Hence, the features learned by the function should be sufficiently abstract to describe all different representations of the same object and only capture semantically meaningful changes. In other words, we require that the function processes information in a way that it is robust to small changes in a

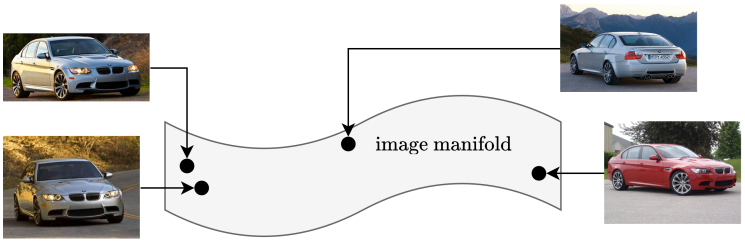


Figure 1.1: Changes in viewing angle, background, lighting and colour are examples of label-preserving transformations, i.e transformations that do not change the object's label. These variations can result in a highly complex geometry as indicated by the drawing of a curved image manifold. Image manifolds of different objects can be entangled, i.e. not linearly separable. The images have been obtained from the *BMW-10* dataset [3].

scene and generalises to entirely different scenes. This process is referred to as *visual object recognition* or *image classification*.

1.1.1 A Geometric Perspective on Object Recognition

In human and other primate brains the information processing for visual object recognition takes place along the *ventral visual stream* [4, 5]. Neurons along this path are hierarchically organised where every stage processes the information received from the previous one [6]. It reaches from the brain areas *V1* to *V2*, then to *V4* and to the *inferotemporal cortex*. In the inferotemporal cortex, image manifolds are separable with simple linear classifiers [1, 7]. However, perceived image manifolds are usually, due to object clutter, highly entangled, and thus linear separation of objects in the original space cannot be achieved. Hence, one task performed by the ventral visual stream is the successive untangling of image manifolds along its hierarchy [1, 8].

Artificial neural networks used for image classification function similarly. They consist of multiple hierarchically organised computations that process the information from the previous step. Their last layer is usually a linear read-out function, so they produce linearly separable representations in their last layer while also receiving non-linearly separable image manifolds as inputs. Thus, artificial neural networks also untangle image manifolds along their hierarchy.

As such the information processing in biologically and artificial neural networks can be studied from a geometric perspective and the aforementioned robustness

requirement can be viewed from a geometric point of view. Scene changes are generally movements along the geodesic, so the surface, of the manifold and certain small changes that are usually not encountered in real-life can constitute movements away from the manifold. In Chapter 2 (page 9) we define these changes more formally and refer to them as *natural* and *synthetic perturbations*, respectively. Thus, the geometric perspective offers a methodology to study representations obtained by image classification algorithms and to connect them to properties, such as their robustness to scene changes.

1.1.2 Motivation

Artificial neural networks have become ubiquitous in object recognition tasks in which they have achieved near- or super-human performance [9, 10]. However, whereas human object recognition is relatively robust to scene changes [11], artificial neural networks show a significant lack of robustness to small and large changes in images (see Section 2.2, page 18). These changes encompass synthetic ones that do not occur in real-life scenarios as well as natural ones that are commonly encountered in real-life tasks. Despite the substantial attention this topic has received in the form of *robust training* methods (see Section 2.4, page 35), progress has been slow and a considerable robustness gap between humans and neural networks persists (see Section 2.4.11, page 44). As neural networks are frequently used in safety critical applications, understanding and mitigating this robustness gap is of significant practical importance.

1.1.3 Research Questions and Contributions

The starting point of this thesis is that object recognition in both biological and artificial neural networks can be viewed from a geometric perspective. We take this perspective and tackle the robustness problem in object recognition with artificial neural networks. Over the last years some studies took the same approach. They study the geometric properties of images and their representations and connect them to the learning behaviour, the robustness and the generalisation performance of artificial neural networks and offer interesting insights into those (see Section 2.5, page 46). In this thesis we identify and answer three main open questions in the

intersection between the literature on robustness and geometric properties of data and its representations. In what follows we state these questions and briefly outline their proposed answers. Then, in Section 1.2 (page 8) we give an overview of the thesis' structure.

Manifold Geometry and Learning

As mentioned above, robustness can be viewed as invariance to movements away from or along the image manifold. Thus, requiring a classifier to be robust, means changing the geometry of the original image manifold by either adding samples in its vicinity or novel samples from the data distribution.

Generally, image manifolds in classification settings can be described by three key geometric properties: the *intrinsic dimension* (Definition 2, page 11), the *extrinsic dimension* (Definition 3, page 11) and the *entanglement* (Definition 4, page 12). These properties describe different aspects of the data distribution's complexity and requiring robustness of a classier increases this complexity. We illustrate these properties in Figure 2.1 (page 10) and formally define them in Section 2.1 (page 10).

Intuitively, the intrinsic dimension describes the number of identity-preserving changes the objects are subject to whereas the extrinsic dimension is simply the number of pixels and colour channels. The entanglement is the number of connected hyperplanes that are required to separate different class manifolds.

It has been observed that training neural networks to be robust requires a significantly larger number of data. In other words, robust training has a larger *sample complexity* (Definition 10, page 32) than *standard training* (see Section 2.3.2, page 32). In classical statistical learning theory it has been theoretically established that a classifier's sample complexity is positively dependent on the intrinsic dimension and the entanglement but independent of the extrinsic dimension [12, 13]. However, the proposed mathematical relationship is, due to its complexity, too cumbersome to work with. Further, neural networks do not always follow the predictions made by classical statistical learning theory (e.g. [14, 15]). Hence, the following question is still open.

Research question 1: *What is the connection between the sample complexity of neural networks and the geometric properties of data distributions and how do*

different properties depend on each other?

In Chapter 3 (page 56) we answer this question and make the following contributions.

- The entanglement is the leading contributor to the sample complexity of deep neural networks. Thus, the more complex the optimal decision boundary, the more samples are required to approximate it.
- The intrinsic dimensionality's influence on the sample complexity strongly depends on the level of entanglement. The intrinsic dimension only marginally influences the sample complexity when the data distribution is easy separable. Thus, for low entanglement distributions, neural networks behave similarly to support vector classifiers.

In contrast to previous work [16] we also consider the entanglement and thus all three geometric properties in one study and provide important interdependencies between them. Further, we study both toy and real-world image benchmarks as well as *fully-connected* and *convolutional* (see [2]) neural networks and find consistent results. Thus, this study offers a baseline to investigate the geometric complexity of decision boundaries that have recently gained interest in the literature (see Chapter 3, page 56).

Complexity of Decision Boundaries

The geometric complexity, i.e. the entanglement, of a decision boundary is the number of connected hyperplanes needed to approximate it. A common hypothesis in the robustness literature is that robust neural networks have geometrically more complex decision boundaries (see Section 4.1, page 82). However, determining this number for non-linear classifiers such as neural networks is still an open problem and thus yields the following open question.

Research question 2: *Do state-of-the-art robust neural networks learn geometrically more complex decision boundaries than non-robust ones and under what circumstances do they do so?*

In Chapter 4 (page 82) we propose a way to circumvent the difficulties of studying neural networks' decision boundaries and make the following contributions.

- State-of-the-art robust neural networks learn geometrically more complex decision boundaries than non-robust models. This observation confirms previous

hypotheses and partially explains the larger sample complexity of robust training.

- We propose an upper bound on the perturbation magnitude in image space over which provably a geometrically more complex decision boundary is required.
- Perturbation magnitudes larger than the aforementioned upper bound introduce label noise which impairs robustness and generalisation performance and thus partially explains the accuracy-robustness tradeoff.
- For common image benchmarks the minimum nearest neighbour distance [17] overestimates the robust radius of a distribution and our proposed bound is a more suitable measure.

In Chapter 4 (page 82) we investigate the decision boundary of several state-of-the-art-robust neural networks to ensure consistent results between them and to find commonalities that very different approaches to robust training might share. Then, in Chapter 5 (page 112) we consider again a subset of recently proposed robust training methods and study other commonalities between them. This time, however, with respect to the intrinsic dimension of the hidden representations.

Mechanisms of Robust Training

Due to the severity of the robustness gap between humans and neural networks, a substantial number of robust training methods have been proposed (see Section 2.4, page 35). These methods have largely been compared by their train or inference time or naturally by their robust accuracy. However, the mechanisms that underlie their performance are still largely unknown. Therefore, we consider the following open question.

Research question 3: *What common mechanisms underlie state-of-the-art robust training mechanisms and why do they exhibit them?*

In Chapter 5 (page 112) we compare several state-of-the-art robust training methods with respect to the geometry of their hidden representations and make the following contributions.

- A common property of robust representations is lower intrinsic dimensionality. This observation holds regardless of the specific method used to obtain them

and even holds for training with the addition of Gaussian noise.

- One reason for this lower dimensionality is partial sharing of features between semantically similar classes.
- This feature sharing is not the result of spatial disentanglement of dissimilar classes but of learning common features for semantically similar classes.

In Chapter 5 (page 112) we show that one common mechanism that distinguishes robust and non-robust representations is dimensionality reduction by learning shared features across classes. Thus, despite learning representations that yield more complex decision boundaries (see Chapter 4, page 82), they also display lower intrinsic dimension and are thus simpler according to this measure. This finding again shows that jointly studying all three geometric properties with respect to the sample complexity and robustness yields valuable insights into the behaviour and properties of neural network classifiers.

1.1.4 Publications

The contributions presented in Chapter 3 (page 56) and Chapter 4 (page 82) have been published in the following peer-reviewed publications, respectively.

- D. Kienitz, E. Komendantskaya and M. Lones. The Effect of Manifold Entanglement and Intrinsic Dimensionality on Learning. *36th AAAI Conference on Artificial Intelligence 2022*. [Link to paper.]
- D. Kienitz, E. Komendantskaya and M. Lones. Comparing Complexities of Decision Boundaries for Robust Training: A Universal Approach. *Asian Conference on Computer Vision 2022*. [Link to paper.]

Further, the author of this thesis contributed to the two following peer-reviewed publications. As these results are orthogonal to the ones presented in this thesis, they have not been included.

- E. Komendantskaya, W. Kokke, and D. Kienitz. Continuous Verification of Machine Learning: A Declarative Programming Approach. In Proceedings of the 22nd International Symposium on Principles and Practice of Declarative Programming. PPDP '20. New York, NY, USA 2020. Association for Computing Machinery.
- W. Kokke, E. Komendantskaya, D. Kienitz, R. Atkey, and D. Aspinall. Neu-

ral networks, secure by construction: An exploration of refinement types. In Programming Languages and Systems: 18th Asian Symposium, APLAS 2020, Fukuoka, Japan, November 30–December 2, 2020, Proceedings 18 67. Springer2020.

1.2 Overview of the Thesis

In Chapter 2 (page 9) we begin with introducing background and the related literature. As mentioned above, the primary interest of this thesis is to study the robustness of artificial neural networks to distributions shifts through the lens of their geometric properties. To that end, we first give an overview of the robustness literature. Over the last years, this area of machine learning has grown rapidly and now encompasses several thousands of papers that deal with the failure types of neural networks (Section 2.2, page 18), the potential reasons for these failures (Section 2.3, page 30) as well as ways to mitigate them (Section 2.4, page 35). Due to the scope of this literature our introduction focuses on breadth and we refer the reader to multiple different survey papers that introduce the particular subject in that area in depth. Following the overview of the robustness literature, is an *exhaustive* overview of the works that study the geometric properties of datasets with respect to the learning process and the robustness of artificial neural networks. In Chapter 3 (page 56), Chapter 4 (page 82) and Chapter 5 (page 112) we present the contributions of this thesis. We close with the conclusions in Chapter 6 (page 133), discuss limitations of our work and propose some possible directions for future work.

Chapter 2

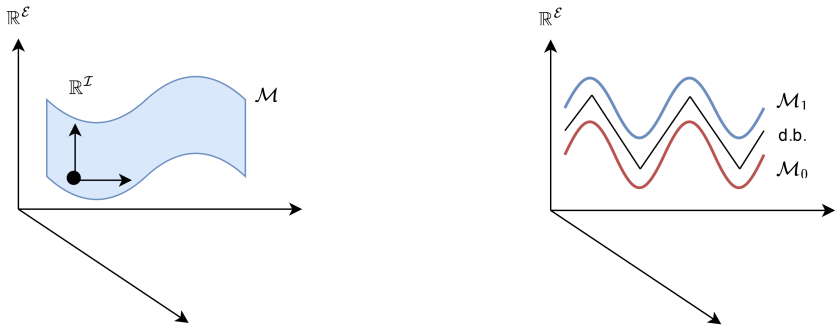
Background and Literature Review

In this chapter we review the literature that is related to our contributions which we present in Chapter 3 (page 56), Chapter 4 (page 82) and Chapter 5 (page 112). The two main areas of relevant research are those that study the robustness of deep neural networks to distribution shifts and those that study the geometric properties of data representations learned by deep neural networks. The literature review is structured accordingly.

First, in Section 2.1 (page 10), we describe the background of this thesis and introduce notations and definitions used throughout it. In Section 2.2 (page 18) we introduce works that investigate the robustness of deep neural networks under a variety of distributions shifts. Despite their strong generalisation performance, deep neural networks display a significant lack of robustness to these shifts and in Section 2.3 (page 30) we overview the main hypothesis introduced in the literature in order to explain this deficiency. Then, in Section 2.4 (page 35) we introduce common training methods that were proposed to alleviate this lack of robustness. Given the number of papers that study robustness of neural networks[1], we focus on those works that are the most relevant to our contributions and give a general overview over the entire area.

The findings in the robustness literature are the main motivation for the contri-

[1] *Adversarial examples* (Definition 7, page 20) are a special kind of distribution shift and as of 2023 over $6,000$ papers have been published on *arXiv* on that topic alone [18].



(a) Manifold dimensionality. (b) Entanglement.

Figure 2.1: Illustration of the three key geometric properties of manifolds. (a) The intrinsic dimension $\mathcal{I}$ is the dimension of the manifold $\mathcal{M}$ (see Definition 2, page 11). The extrinsic dimension $\mathcal{E}$ is the dimension of the ambient space that $\mathcal{M}$ is embedded in (see Definition 3, page 11). Here $\mathcal{I} = 2$ and $\mathcal{E} = 3$. (b) The entanglement Σ is the number of connected hyperplanes required to approximate the decision boundary (d.b.) separating the manifolds $\mathcal{M}_0$ and $\mathcal{M}_1$ (see Definition 4, page 12). Here $\Sigma = 5$.

butions presented in this thesis and underlie all of them. Based on this literature, we investigate certain aspects of neural network robustness through the lens of the geometry of their representations. To this end, we give an *exhaustive* overview of works published up to mid 2023 that study any of the key geometric properties of data distributions or representations with respect to the robustness or sample complexity of neural networks in Section 2.5 (page 46).

2.1 Background

In this section we provide the background definitions underlying this thesis. As we primarily focus on applications to supervised image classification, we restrict this section to this topic.

2.1.1 Supervised Classification

An image dataset $X \in \mathbb{R}^{l \times \mathcal{E}}$ consists of $l \in \mathbb{N}_{>0}$ images where each image is written as $x \in \mathbb{R}^{1 \times \mathcal{E}}$. In this thesis we only consider classification scenarios in which every image is associated with exactly one label y. The variable $\mathcal{E}$ describes the dataset's variables, so in the case of images the number of pixels and colour channels. We assume that there exists a joint distribution $P(x, y)$, which implies that samples that

share a common label also share a common set of features. It is frequently assumed in machine learning that distributions of natural images $P(x|y)$ concentrate near a label-specific manifold (see [2]).

Definition 1. ***Manifold***

A $\mathcal{I}$-manifold $\mathcal{M}$ is a topological space where the neighbourhood of every sample is homeomorphic to an open subset of $\mathcal{I}$-dimensional Euclidean space.

In the mathematics literature the term manifold is generic and refers to a wide variety of different structures. As distributions of natural data usually concentrate near low-dimensional subspaces [16] and movements along these subspaces are label-preserving transformations [2], their structure can naturally be described by a manifold. Thus, in this thesis we use the term manifold loosely to refer to distributions whose structure can be characterised by the following three geometric properties. We also use the terms manifold and image manifold interchangeably.

The first key characteristic of a manifold is its intrinsic dimension which is defined as follows.

Definition 2. ***Intrinsic dimension of a manifold***

The variable $\mathcal{I}$ in Definition 1 (page 11) is referred to as the intrinsic dimension of the associated manifold $\mathcal{M}$. It describes the dimension of the manifold's tangent space.

A tangent space is the multi-dimensional generalisation of a tangent line or a tangent plane for 1-manifolds or 2-manifolds, respectively. Thus, it describes the degrees of freedom for movements along the manifold' surface. The intrinsic dimension can be viewed as the number variables that are required to describe the distribution that concentrates near it. Movements along the manifold's geodesic correspond to label-preserving transformations, such as changes in background, lighting, colour and perspective (see Figure 1.1, page 2 for illustration). Manifolds are embedded within an ambient space. The extrinsic dimension $\mathcal{E}$ of an $\mathcal{I}$-manifold is defined as follows.

Definition 3. ***Extrinsic dimension of a manifold***

The extrinsic dimension of a manifold describes the dimensionality of its ambient space, i.e. the space it is embedded within.

For natural images, the size of the ambient space is simply the product of the number of pixels and colour channels, i.e. $\mathcal{E} = |\text{pixels}| \cdot |\text{channels}|$.

Further, in classification settings every class is associated with at least one manifold. In these scenarios the third key geometric property is the following.

Definition 4. ***Entanglement of manifolds***

Given two or more image manifolds, the entanglement refers to the minimum number of connected hyperplanes that are required to perfectly separate them.

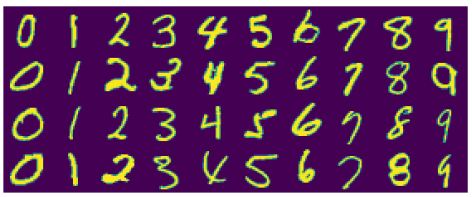
Assuming that image distributions can be separated by *connected* hyperplanes is a reasonable assumption as humans assign distinct labels to semantically different classes. This yields a separation of the associated image manifolds that can be achieved with connected hyperplanes. However, real-world image datasets used in practice might contain labels noise or ambiguous examples. In this case finding a set of connected hyperplanes that separate class manifolds perfectly, that is with no wrongly labelled samples, is not possible. In good quality datasets, however, label noise and ambiguous images should be rare occurrences. Thus, although the definition of does not hold perfectly, it is only violated for a finite set of examples.

In Figure 2.1 (page 10) we illustrate the three aforementioned geometric properties of data distributions. With these three properties an image classification problem can be characterised from a geometric perspective. These properties have been subject to some recent works in machine learning and have been connected to their generalisation performance and learning behaviour (see Section 2.5, page 46). In this thesis we revisit them and study their influence on artificial neural networks. We consider three open questions in the literature which we introduced in Section 1.1.3 (page 3) whose answers yield novel insights into their generalisation performance and learning behaviour.

Data and Distribution Manifold

In machine learning the data distribution $P(x|y)$ from which we draw the available samples X with label y concentrates near a manifold. We distinguish between the *data manifold* and the *distribution manifold*.

The *data manifold* is the space near which the available train samples from X concentrate with label y. The *distribution manifold*, on the other hand, is the

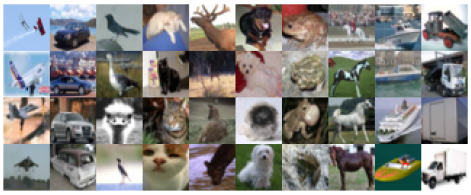


(a) MNIST.

(b) FASHION.

Figure 2.2: Examples Images of the MNIST and FASHION datasets.

(a) SVHN.



(b) CIFAR-10.

Figure 2.3: Example images of the SVHN and CIFAR-10 datasets.

space near which all samples $x \sim P(x)$ of label y concentrate. Thus, by these definitions, there exist samples that are not part of the data manifold but are part of the distribution manifold, but not vice versa. Distinguishing between data and distribution manifold allows us to define robustness to naturally occurring image changes as generalisation to samples that are part of the distribution manifold but are not part of the train samples X that constitute the data manifold.

Image Datasets

In this thesis we use four common image datasets. These datasets range from a simple ones which are used mostly for illustration purposes in the literature, to a challenging real-world datasets. All datasets consist of a comparably small number of samples l so experiments can be run on a single off-the-shelf graphics processing unit.

The *MNIST* [19] dataset consists of $60,000$ grey-scale train images and $10,000$ grey-scale test images of the handwritten numbers one to ten (see Figure 2.2a, page 13 for example images). It is considered to be simple as it is almost[2] linearly separable and can be solved by only relying one a few pixels for classification without learning any semantic features that humans might use.

[2]Training a *linear support vector classifier* [20] with the one-vs-rest method on MNIST results in 91.36% test accuracy. In comparison, on CIFAR-10 only 20.68% test accuracy is achieved.

The *FASHION* [21] is designed to be drop-in replacement for the MNIST dataset. It consists of $60,000$ grey-scale train images and $10,000$ grey-scale test images spread uniformly over 10 different clothing items (see Figure 2.2b, page 13 for example images). It is considered to be more challenging than MNIST, however, compared to to other standard computer vision datasets it is still considered to be a simple dataset used mostly for illustration purposes in the literature.

We additionally utilise the *SVHN* [22] dataset. It consists of $73,257$ coloured train images and $26,032$ coloured test images of the house numbers one to nine, where the number of images per class vary (see Figure 2.3a, page 13 for example images). This dataset can be considered to be of moderate difficulty as it is not linearly separable and the images display different types of noise.

Finally, we use the *CIFAR-10* [23] dataset. It consists of $50,000$ coloured train images and $10,000$ coloured test images of 10 commonly encountered objects such as cars, horses and ships (see Figure 2.3b, page 13 for example images). As SVHN, CIFAR-10 is a real-world dataset. It is ubiquitous in the computer vision literature and is considered to be one of the most challenging datasets among the ones with a smaller number of samples.

Image Classifiers

The goal of any classification function $f : \mathbb{R}^{1\times\mathcal{E}} \to \mathbb{R}^{y}$ is to assign any given input $x \in \mathbb{R}^{1\times\mathcal{E}}$ a label y. We assume the existence of a classification function that can assign every sample of the data distribution its ground truth label. We refer to this classifier as the *Oracle classifier* and define it as follows.

Definition 5. ***Oracle classifier***
For a given distribution $P(x)$ and labels y, an oracle $\mathcal{O} : \mathbb{R}^{1\times\mathcal{E}} \to \mathbb{R}^{y}$ assigns each $x \sim P(x)$ its ground truth label y, i.e. it represents the ground truth distribution $p(y|x)$. In other words, $\mathcal{O}$ is the perfect classifier that correctly classifies every sample in $P(x)$ and thus every sample in a dataset X, with entries $x \sim P(x)$.

For natural data it is reasonable to assume that small changes to the input do not result in large output changes, i.e. if $x \approx \tilde{x}$, then $f(x) \approx f(\tilde{x})$, which is sometimes referred to as the *Lipschitz-smoothness prior* [24]. In other words, altering an image such that no class change is recognizable by the oracle (Definition 5, page 14) should

not produce an image of a different class and therefore the label predictions should be consistent. This assumption of local smoothness yields the following important property of classification functions.

Definition 6. ***δ-robustness of a classifier***
Let $f : \mathbb{R}^{1\times\mathcal{E}} \rightarrow \mathbb{R}^y$ be a classifier that maps a sample $x \in \mathbb{R}^{1\times\mathcal{E}}$ to a possible label y. We say that f is δ-robust in some p-norm (Definition 2.2, page 15) if and only if

$$\forall x, \tilde{x} \in P(x) : \ argmax\, f(\tilde{x}) = argmax\, f(x), \ where\, ||\tilde{x} - x||_p \leq \delta \tag{2.1}$$

Thus, every possible norm-bounded perturbation $\tilde{x}$ of x is classified with the same label as x.

p-norm and semantic similarity

The p-norm of a vector $x \in \mathbb{R}^{1\times\mathcal{E}}$ is defined as

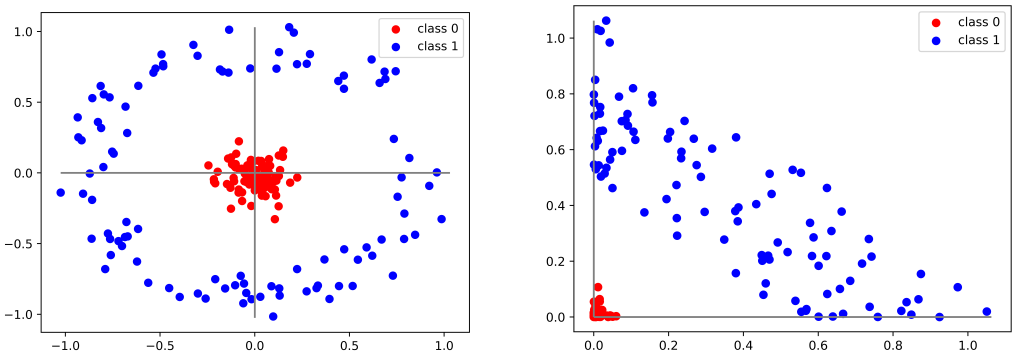
$$||x||_p = \Big(\sum_{i=1}^{\mathcal{E}} |x_i|^p\Big)^{\frac{1}{p}} \tag{2.2}$$

Generally, the p-norm between the difference of two images is a not a sufficient proxy for their semantic similarity. For example, depending on the choice of the p-norm, even changes with large magnitudes can introduce barely visible changes. Nevertheless, p-norm are frequently employed in the literature if the perturbations magnitudes δ are small (see Chapter 4, page 82). One reason for their widespread usage is that learning a distance metric that captures semantic similarity between images is the goal of classification. Hence, the availability of such a metric would imply that the classification task is solved. Thus, using some p-norm is the best proxy for semantic similarity that does not require solving the classification task.

The Kernel Trick

As noted, the fundamental task of biological and computer vision can be viewed as the disentanglement of manifolds[3]. Thus, image distributions that are not linearly

[3]Stephenson et al. [25] study the representational geometry of neural networks trained for speech recognition. They find analogous results in this case, namely that neural networks disentangle auditory manifolds consisting of word classes, speaker classes or other semantic concepts.



(a) Before kernel-trick.

(b) After kernel-trick.

Figure 2.4: Illustration of the kernel trick. (a) The original data is not linearly separable. (b) After applying the function $k(x) = [(x^{(0)})^2, (x^{(1)})^2]$ to the data, the samples are linearly separable.

separable in their original ambient space are processed in such way that their representations at the end of the hierarchy are usable for linear classifiers. In this section we illustrate such processing for a non-linearly separable dataset that is suitable for a linear classifier after using the so-called *kernel-trick*.

In Figure 2.4 (page 16) we illustrate the kernel trick with a simple example. The goal is to find a mapping $k : \mathbb{R}^2 \to \mathbb{R}^2$ that transforms the original dataset in Figure 2.4a (page 16) such that it is linearly separable. It is easy the see that choosing $k(x) = [(x^{(0)})^2, (x^{(1)})^2]$ results in a linearly separable dataset as displayed in Figure 2.4b (page 16).

Thus, we can recast the procedure employed by biological and artificial neural networks as the learning of a kernel function k that transforms the image distributions such that they are separable by a linear classifier. In the following section we introduce the mathematical operations performed by artificial neural networks more formally.

2.1.2 Artificial Neural Networks

From here on, we only consider artificial neural networks which we refer to plainly as neural networks. A neural network refers to a function that consists of *neurons* as its basic building blocks. The mathematical operation carried out by a neuron f_i is usually written as

$$f_i(x) = \sigma(Wx^T + b) \tag{2.3}$$

where Wx^T refers to the matrix multiplication between a the set of learnable weights $W \in \mathbb{R}^{\mathcal{E}\times\mathcal{E}'}$ and inputs $x \in \mathbb{R}^{1\times\mathcal{E}'}$ and $\sigma : \mathbb{R} \to \mathbb{R}$ is an activation function that is point-wise applied to every element in $f_i(x) \in \mathbb{R}^{1\times\mathcal{E}}$. A neural network f is the composition of L such neurons:

$$f(x) = f_L(f_{L-1}(f_{...}(f_1(x)))) \tag{2.4}$$

We refer to $f_i(x)$ as a *hidden representation* of x.

The computational model of a neuron was provided by Rosenblatt [26] and the idea of stacking multiple neurons for hierarchical processing of information was first provided by Fukushima et al. [27–29]. Originally, the parameters θ of f, were trained by an unsupervised learning rule. Later, *back propagation* [19] was used to minimise a loss function $\mathcal{L}(f_L(x), y; \theta)$ between the prediction $f_L(x)$ and the label y by optimising the function's parameters θ. For a full overview of neural networks and their history we refer the reader to the survey by Schmidhuber [30].

Custom Architectures

In this thesis, when using custom architectures we only use the rectified linear unit $\sigma(x) = \max(0, x)$ (ReLU) ([28, 31, 32]) as an activation function which is a common choice in deep learning. Further, if the matrix multiplication wx in Equation 2.3 (page 16) is the *convolutional operation* we refer to the model as a *convolutional neural network*, as opposed to a *fully-connected neural network* when the matrix multiplication is the *dot-product* (see [2]).

Conclusion

This concludes the background section in which we briefly described neural networks applied to image datasets and their geometric properties. Further, we gave an intuitive example for the processing of data to make them linearly separable. In what follows we introduce the related literature of this thesis.

Table 2.1: Taxonomy of distribution (covariate) shifts used in this literature review and example studies.

Synthetic perturbations (Sec. 2.2.1, page 19)		**Natural perturbations** (Sec. 2.2.2, page 23)		
Small-norm	Large-norm	Object/Image	Background	Other
• Adversarial examples: [33, 34]	• Unrecognisable images: [35]	• Geometric: [36–40] • Noise: [41] • Blur: [41]	• [42–45]	• Collection: [46, 47] • Time: [48]

2.2 The Lack of Robustness of Neural Networks

A large body of work has shown that despite their strong generalisation performance and robustness to label noise [49] neural networks lack robustness to a wide variety of *distribution shifts*. This lack of robustness to distributions shifts hinders the adoption of neural networks to safety critical applications and is a somewhat surprising property as the human visual system is robust against previously unseen distortions [11].

Distribution shifts are usually separated into *label shift*, *concept shift* and *covariate shift*. When the joint feature-label distribution $p(x, y)$ is decomposed into

$$\begin{aligned} P(x, y) &= P(x|y)\ P(y) \\ &= P(y|x)\ P(x) \end{aligned} \tag{2.5}$$

then these types of distribution shifts can be characterised as follows. Label shift refers to the case in which $P(y)$ changes but $P(x|y)$ remains constant, concept shift refers to the case in which $P(y|x)$ changes but $P(x)$ remains constant and covariate shift refers to the case in which $P(x)$ changes but $P(y|x)$ remains constant.

In the context of machine learning, and especially deep learning, covariate shifts are the most widely studied type of distribution shift. They are frequently encountered in real-world tasks and are generally the most challenging to deal with. In this thesis we only consider covariate shifts and we use this term interchangeably with the term distribution shift since it is the data set that changes. In what follows we describe the literature on the vulnerability of deep neural network classifiers to this type of distribution shift. We focus on supervised object recognition and briefly introduce robustness deficits in other tasks and algorithms in Section 2.2.4 (page 27).

There are multiple taxonomies of distribution shifts employed in the literature

(see for example Koh et al. [50]). Here, we choose to divide them broadly into *synthetic* and *natural* distribution shifts. Synthetic distribution shifts are those where the resulting images are not part of the data distribution any more and natural shifts are those where they still are. In other words, synthetic perturbations cannot be encountered in real-life situations whereas natural perturbations can be. The most common form of synthetic distribution shifts are classical *adversarial examples* [33] (Definition 7, page 20) which are crafted for a specific model and are usually not encountered in real-life mage datasets. Natural distribution shifts are for example changes in lighting or background or the introduction of blur. In Table 2.1 (page 18) we display some important papers using this taxonomy.

Using the argumentation of Stutz et al. [51], synthetic perturbations result in samples that lie off the distribution manifold (see Section 2.1.1, page 10) whereas samples altered by natural perturbations still lie on it. Thus, robustness to natural perturbations can be considered as standard generalisation to the entire data distribution and not just the test set. We further divide synthetic and natural distribution shifts into *small-norm* and *large-norm* shifts which loosely correlates with their visibility to human observers (see Figure 5.1, page 116 and Table 5.1, page 117). It is commonly assumed for both synthetic and natural distribution shifts that the applied perturbations do not change the ground truth label, so an oracle (Definition 5, page 14) does not recognise a class change.

In Section 2.2.1 (page 19) we describe the most common form of synthetic perturbations and introduce algorithms for crafting it that are also used later in this thesis. Then, in Section 2.2.2 (page 23) we describe natural perturbations. In Section 2.2.3 (page 26) we discuss that the distinction between small-norm and large-norm perturbations is required as empirically there appears to be a tradeoff between these two. We focus primarily on applications to image datasets in object recognition settings but briefly mention related works in other areas in Section 2.2.4 (page 27).

2.2.1 Robustness to Synthetic Perturbations

Szegedy et al. [33] investigate neural network classifiers for image datasets. They find that applying small-norm, for humans (almost) imperceptible, perturbations to samples is sufficient to change the predicted label. The authors define such samples

(a) (b) (c)

Figure 2.5: Example of a synthetic perturbation in the form of an adversarial example (Definition 7, page 20) for a neural network which achieves 82.52% test accuracy on the *CIFAR-10* [23] dataset. (a) Original (benign) sample from the CIFAR-10 dataset. The neural network correctly predicts the class *airplane* (label: 0) with 1.0 confidence. (b) For the adversarial example the neural network falsely predicts the class *deer* (label: 4) again with 1.0 confidence. (c) The applied perturbation is created by PGD (Definition 9, page 21) and is visualised by summing over each pixel's colour channel. Values of the colour channels are in the interval of real numbers $[0, 1]$.

as *adversarial examples*. In Figure 2.5 (page 20) we display an adversarial example and its corresponding original image along with the applied perturbation. While the applied perturbation is only barely visible to humans and does not change the ground truth label (*airplane*), the neural network predicts the wrong one (*deer*) with perfect confidence.

Definition 7. ***Adversarial example*** *[33]*

Let $x \in \mathbb{R}^{1\times\mathcal{E}}$ be a sample from the dataset $X \in \mathbb{R}^{l\times\mathcal{E}}$, and let f be a trained classifier. An adversarial example $x^{adv} := ||x + \epsilon||_p$ in some p-norm for some sample x can be found by solving the following optimisation problem

$$\begin{aligned} &min\ ||\epsilon||_p\ s.t. \\ &f(x+\epsilon) \neq f(x)\ ,\ \mathcal{O}(x+\epsilon) = \mathcal{O}(x) \end{aligned} \tag{2.6}$$

Thus, an adversarial example changes the predicted label of the classifier while keeping the applied perturbation as small as possible as to not change the prediction of the oracle $\mathcal{O}$.

An example of large-norm synthetic perturbations was offered early by Nguyen et al. [35]. Here the authors show that neural networks confidently predict class labels on inputs that do not resemble natural images at all.

Algorithms for Adversarial Attacks

Over the last years a large number of algorithms have been proposed to efficiently solve the optimisation problem in Definition 7 (page 20). For a complete overview of adversarial attack algorithms we refer the reader to the survey by Serban et al. [52]. In most cases these applied perturbations are imperceptible to humans [33] and can even consist of changing just one pixel in an image [53].

In this thesis we only utilise the *fast gradient sign method* [54] (FGSM) and *projected gradient descent* [55] (PGD) and describe them below. These two methods have been shown to be computationally efficient and hard to defend against even with state-of-the art defence methods [56]. FGSM and PGD are also the backbone of many *adversarial training methods* that we describe in Section 2.4 (page 35).

Definition 8. ***Fast gradient sign method (FGSM)*** *[54]*
Given a classifier f with loss function $\mathcal{L}$ and a sample $x \in \mathbb{R}^{1\times\mathcal{E}}$ with associated label y, FGSM computes its adversarial example x^{adv} as

$$x^{adv} \coloneqq x + \epsilon \; sign(\nabla_x \mathcal{L}(x, y)) \tag{2.7}$$

where $\nabla_x \mathcal{L}$ is gradient of the loss-function with respect to the input. We write FGSM-ϵ to denote the application of the FGSM algorithm with magnitude ϵ.

Definition 9. ***Projected gradient descent method (PGD)*** *[55]*
Given a classifier f with loss function $\mathcal{L}$ and a sample $x \in \mathbb{R}^{1\times\mathcal{E}}$ with associated label y, PGD computes its adversarial example x^{adv} in iteration $t \in \mathbb{N}_+$ as

$$\begin{aligned} x_{t+1} &\coloneqq \Pi[x_t + \epsilon \; sign(\nabla_x \mathcal{L}(x_t, y))] \\ with\ x^0 &\coloneqq x \end{aligned} \tag{2.8}$$

where $\nabla_x \mathcal{L}$ is gradient of the loss-function with respect to the input and Π is a projection operator mapping the resulting sample onto a user-defined domain. We write PGD-ϵ-t to denote the application of the PGD algorithm with magnitude ϵ and t iterations.

In practice, FGSM and PGD often find perturbations that are imperceptible to humans, however they are not guaranteed to find the minimum-norm solution to

the optimisation problem in Definition 7 (page 20). Dezfooli et al. [57] propose *DeepFool*, an algorithm that efficiently finds adversarial examples whose applied perturbation is close to the minimum-norm solution. For a detailed description of DeepFool we refer the reader to the original paper. We utilise DeepFool in Chapter 4 (page 82) to show that state-of-the-art robust models learn functions whose margin (i.e. the distance to the decision boundary) is close to the maximum possible.

The above algorithms assume access to the target model's parameters and are usually called *white-box attacks*. In contrast *black-box attacks* are those that do not require access to the model's parameters. One example of a black-box attack is the method presented by Alzantot et al. [58] who utilise a *genetic algorithm* (see Luke et al. [59] for an overview) to generate imperceptible adversarial examples for neural networks on many commonly used real-world image benchmarks. We do not employ any black-box attacks in this thesis and thus refer the reader to the survey of Serban et al. [52] for a complete overview of adversarial attack algorithms.

Adversarial Subspaces

Adversarial examples as defined above are crafted for a specific neural network. Szegedy et al. [33] recognise that with high probability adversarial examples transfer to other neural networks with different number of hidden layers, activation functions or trained on a different subset of the train data. Fawzi et al. [60] study the learned decision boundaries of neural networks and show that adversarial examples occur where the decision boundary is curved. The directions along which the boundary is curved can be shared between models and samples, yielding *universal adversarial examples* that transfer between different models. Tramer et al. [61] study these universal adversarial examples and show that they lie within relatively large continuous subspaces shared between many input samples. These spaces are usually referred to as *adversarial subspaces*. Gu et al. [62] further highlight that adversarial subspaces are continuous by showing that perturbing an adversarial examples with additional random noise is not effective in turning the adversarial example benign again. Ma et al. [63] estimate the local intrinsic dimension of these adversarial subspaces and find that they are significantly larger than the intrinsic dimension of the data manifold near which benign samples concentrate.

Invariance-based Adversarial Examples

According to Definition 7 (page 20), an adversarial example changes the classification decision of the target model but not that of an oracle, i.e. the ground truth label does not change. Thus, these adversarial examples can be referred to as *sensitivity-based* adversarial examples, as they exploit overly sensitive features [64]. Jacobsen et al. [34, 65] find that neural networks are also susceptible to the opposite case. *Invariance-based* adversarial examples do not change the prediction of the target model but do change the classification decision of the oracle, so the ground truth label. Tramer et al. [66] compare sensitivity-based and invariance-based adversarial examples and find that increased robustness to one of them results in reduced robustness to the other one. They further hypothesise that invariance-based adversarial examples might be caused by the presence of overly robust and predictive features that the models rely on for classification.

In this thesis we only utilise sensitivity-based adversarial examples, so those described in Definition 7 (page 20).

2.2.2 Robustness to Natural Perturbations

We define natural perturbations as those that can be encountered in real-life scenarios, so are part of the data distribution but may not be part of the training data set. We further divided natural perturbations into three different groups. The first group encompasses transformations that can be easily modelled mathematically, such as rigid and non-rigid transformations. These are for example geometric transformations like rotations. The second group consists of perturbations that only affect the background but not the object to be recognised. Finally, the last group encompasses those perturbations that are hard to model mathematically and are the result of a different dataset collection process.

Perturbations to the Object

Geometric transformations, such as rotation, scaling or translation, do not change the ground truth label of natural objects and are frequently encountered in real-life situations. Therefore, they can be considered as being part of the distribution

manifold (see Section 2.1.1, page 10). The human visual system is invariant to geometric transformations [1] and ideally image classifiers should be invariant to these transformations, too [67]. While the convolutional operation is translation invariant, rotation invariance is not an inductive bias of convolutional neural networks. Thus, some works studied the robustness of convolutional neural networks to geometric transformations.

Lenc et al. [36] as well as Soatto et al. [37] find that convolutional neural networks are not necessarily invariant to geometric changes. To test the invariance of neural networks to geometric transformations more systematically, Fawzi et al. [38] introduce *ManiFool*, an algorithm that finds a geometric transformation that changes the prediction of the model. They report that models are vulnerable to small, and sometimes even imperceptible, geometric transformations. Later, Kanbak et al. [39] propose an algorithm to find label-changing geometric transformations that is scalable to larger networks and manifolds. In previous works, the geometric transformation acted on the entire image. In contrast, Alcorn et al. [40] utilise three-dimensional models of objects and change their position and orientation within the image. They also report signifiant robustness deficits to these sorts of transformations.

Hendrycks et al. [41] propose a benchmark dataset with common perturbations encountered in real-life scenarios such as different types of noises and blurs. They find that accuracy on in-distribution samples is significantly larger than accuracy on the proposed benchmark dataset[4].

Humans [4, 5] and other vertebrates [68] rely on the shape of objects for recognition. Geirhos et al. [69] find that this shape-bias is not present in convolutional neural networks when trained on *ImageNet* [70] dataset. Instead, convolutional neural networks heavily rely on texture clues. The authors introduce a novel test set with a texture-shape conflict and find that when trained on the original ImageNet dataset neural networks perform poorly on this novel test set. Hermann et al. [71] also argue that this texture-bias is a result of the provided data by showing that carefully crafted data augmentations can improve the shape-bias of neural networks.

[4]In this thesis we frequently use the benchmark by Hendrycks et al. [41] and in Figure 5.1 (page 116) we display some example images from it.

Perturbations to the Background

In the previous paragraph we introduced work in which either the entire image, including the object to be recognised, or only the object is altered. We define the background, so the context, of an object as the complement of that given object. Backgrounds can offer predictive clues for classifying the object [72]. Humans, for example, have been observed to rely on these predictive background features as their object recognition ability is hampered when familiar objects are placed in an incongruent context [73]. Convolutional neural networks are also vulnerable to the context changes, though to a greater degree than humans [73]. Nevertheless, natural objects are usually invariant to the context they are placed in, so background changes should ideally not alter the predictions of neural networks. Below, we highlight key works that established neural networks' over-reliance on object backgrounds.

Zhu et al. [42] show that removing the object from an image deteriorates the generalisation performance only marginally in contrast to humans. Beery et al. [43] introduce a dataset to measure the generalisation of neural networks for animal detection to different camera backgrounds and also find significant drops in performance when animals are photographed in front of a different background. Carter et al. [44] show that convolutional neural networks trained on CIFAR-10 and ImageNet base their decisions on 5% to 10% of the available pixels. In ImageNet, the border pixels are sufficient for training and testing. Thus, in common computer vision benchmarks the background provides strong and easily decodable signals for the label on which neural networks learn to rely. Xia et al. [45] further show that neural networks rely on background features to a greater extend than humans. The vulnerability to background changes is highly related to the existence of spurious correlations in common image benchmarks which we will discuss in Section 2.3.1 (page 30).

Related to the problem of recognising objects outside their natural context is the issue of *crowding*. Crowding refers to the scenario in which the ability for recognition deteriorates when another object displayed close to the original one without occluding it. Humans have been shown to be susceptible to crowding [74, 75] and Volokitin et al. [76] show that convolutional neural networks also suffer from performance drops in this scenario.

Perturbations to the Entire Image

Recht et al. [46] investigate generalisation to a distribution resulting from a new dataset collection process. They create new test sets for CIFAR-10 and ImageNet by mimicking the old dataset collection processes. They find significantly reduced test performances compared to the original test set but also a positive correlation between them. Thus, the authors hypothesise that neural networks do not overfit on the original test set but suffer the drop on the new test due to the distribution shift induced by the suboptimal repetition of the original data creation process. Hendrycks et al. [47] also gather a new test dataset set for ImageNet with the same set of labels but a different distribution of images and obtain similar findings.

Shankar et al. [48] investigate the robustness of neural networks to perturbations that result from changes between temporally near video frames. They find that both object localisation and classification performance deteriorates in this scenario.

2.2.3 Relationship between Synthetic and Natural Perturbations

Whether robustness to synthetic distribution shifts improves robustness to natural distribution shifts is unclear [77]. On the one hand, Taori et al. [78] note that robustness to shifts that can be modelled by functions such as adversarial perturbations or some sort of noise and blur perturbations does not transfer to natural dataset shifts and vice versa. But on the other hand, Hendrycks et al. [79] show that using artificial data augmentations can improve robustness on real-world distribution shifts.

Geirhos et al. [80] note that adversarial robustness is negatively correlated with robustness to large-norm perturbations. Thus, despite the fact that adversarial training induces some useful priors [81], like shape-bias [80], the robustness to different distribution shifts appear to be independent problems. In Chapter 5 (page 112) we find further empirical evidence that robustness to different types of distributions shift are negatively correlated.

Tradeoff between Accuracy and Robustness

In addition to the apparent tradeoff between robustness to small-norm adversarial perturbations and large-norm natural perturbations, several authors noted a tradeoff between adversarial robustness and generalisation performance [82]. This observation is usually referred to as the *accuracy-robustness tradeoff*.

Tsipras et al. [82] and Zhang et al. [83] argue that this tradeoff is theoretically inevitable, however, they use a data distribution for which no robust classifier exists, a scenario that is unlikely the case for real-world datasets as they are usually well-separated [17]. Instead, the higher sample complexity of robust training has been named as one reason for the observed tradeoff [84, 85]. Sanyal et al. [86] also show that robust training methods do not lead to the memorisation of atypical examples which leads to a drop in standard accuracy. Further, Rade et al. [87] show that adversarial training increases the margin along some discriminative directions which leads to a reduction in their predictive power. They propose to add intentionally wrongly labelled samples to counteract this effect and report improved accuracy and robustness.

In Chapter 4 (page 82) we show that state-of-the-art robust training methods indeed learn geometrically more complex decision boundaries which in combination with our results from Chapter 3 (page 56) proves that robust training has indeed a higher sample complexity. Further, in Chapter 4 (page 82) we show that robust neural networks classify extremely blurry images which introduces label noise and hurts standard accuracy.

2.2.4 Robustness Deficits in other Tasks and Algorithms

Previously, we introduced the literature dealing with neural network robustness in the context of object recognition. In this section we give a brief overview of the literature that investigates the robustness of neural networks in other tasks as well as the robustness of algorithms other than neural networks.

Other Computer Vision Tasks

Related to object recognition are the tasks of *semantic segmentation* (see Minaee et al. [88] for an overview) in which every image pixel is supposed to be classified and *object detection* (see Liu et al. [89] for an overview) where the location of an object in the image is supposed to be highlighted. Xie et al. [90] show that these tasks are also susceptible to adversarial attacks and that the perturbations applied here are also transferable between training sets and network architectures.

Elsayed et al. [91] show that adversarial perturbations can even be used to make neural networks perform other tasks than they were originally trained on. For example, the authors reprogram an ImageNet classifier to perform square counting in the given perturbed image.

Reinforcement Learning Tasks

As neural networks are frequently used as backbones of other algorithms (e.g. [92, 93]), their vulnerability might also affect the system they are part of. Huang et al. [94] demonstrate that in a *reinforcement learning* settings (see Montague et al. [95] for an overview) perturbing pixels of the input images for a policy network leads to the same output changes as in object recognition.

Natural Language Processing Tasks

Neural networks applied in natural language processing tasks have also been shown to be susceptible to adversarial attacks. For example, Qin et al. [96] show that it is possible to generate adversarial examples for speech recognition systems that are still correctly classified by humans. In this thesis we only consider applications in computer vision and refer the reader to the survey by Zhan et al. [97] for an overview of adversarial examples in natural language processing.

Other Algorithms

Compared to convolutional neural networks (CNNs), *Transformers* are a novel architecture that use the attention-mechanism [98] and not convolutional operations. Transformers are ubiquitous in natural language processing applications, but have recently also been used in computer vision [99]. Naturally, robustness comparisons

between transformers and classical CNNs have been conducted [100, 101] that find a greater robustness of Transformers compared to CNNs. However, Bai et al. [102] argue that these studies are flawed as they do not account for differences in architecture sizes, architecture configurations as well as training procedures. Contrary to previous works, these authors find that Transformers do not exhibit greater robustness to adversarial examples. Only against natural perturbations Transformers outperform CNNs.

Whereas the majority of previous works deals with feed-forward neural networks, Papernot et al. [103] study the robustness of recurrent neural networks. They find that these architectures also suffer from a lack of robustness to small-norm perturbations. Sengupta et al. [104] propose regularising the spectral norm of weight matrices of recurrent neural networks to alleviate the vulnerability to small-norm perturbations.

The previously mentioned works study robustness properties of neural networks trained with label supervision. Shi et al. [105] extend the robustness analysis to unsupervised learning objectives and algorithms. They find representations obtained by unsupervised training are less vulnerable to distribution shifts than supervised ones but still lack the robustness of humans.

Whereas all previous works study the robustness of some form of deep learning method, other works focus on classical algorithms such as the *k nearest-neighbours* classifier [106]. Papernot et al. [107] show that adversarial examples also exist for other algorithms such as *decision trees* (see Breiman et al. [108]) and *support vector machines* [20] and that they transfer between these models. Fazwi et al. [109] study robustness from a theoretical perspective and derive bounds on the robustness in p-norms for a given dataset that are model-agnostic.

2.2.5 Conclusions

Neural networks have been shown to be vulnerable to a wide variety of distribution shifts ranging from small-norm synthetic perturbations to natural perturbations that are encountered in real-life scenarios. Although some progress has been made in closing the robustness gap between humans and neural networks [80, 110], it remains significant. Since neural network are increasingly deployed in safety-critical

applications, their robustness deficits are of great practical importance. Further, the robustness gap between humans and neural networks as well as their different failure types [80] also highlight that current computer vision algorithms conduct object recognition significantly differently from humans. Thus, studying neural networks to better understand the human visual system is limited.

In the next section we introduce the literature that deals with finding the reasons for this observed lack in robustness.

2.3 Reasons For the Lack of Robustness of Neural Networks

In this section we introduce the main hypotheses made to explain the lack of robustness of neural network classifiers. Previous work showed that the robustness to natural distribution shifts can be mitigated comparatively easily by incorporating these perturbations into the train set [11]. Thus, the vulnerability to natural perturbations appears to be the result of a violation of the *independent-identically distributed assumption* on which neural networks heavily rely. However, the reason for the observed adversarial vulnerability is unclear and over the last years several, and sometimes competing, hypotheses have been made. In this section we introduce the most important ones.

2.3.1 Violation of the independent-identically distributed Assumption

Szegedy et al. [33] hypothesise that adversarial samples lie within 'low-dimensional pockets' of the data distribution and are thus not encountered during training. This hypothesis was later challenged. Stutz et al. [51] show that adversarial examples actually lie off the distribution manifold. The authors train a *VAE-GAN* [111, 112] (a combination of a *variational autoencoder* [113] (VAE) and a *generative adversarial network* [114] (GAN)) to learn a mapping onto the manifold of the train samples. For MNIST and EMNIST [115] the authors show that the distance between benign and adversarial samples in the VAE-GAN's latent space is higher than for other

samples. Further, samples that are loss-maximising for a classifier but constrained to be on the learned manifold correspond to transformations of the original samples. Later, Ma et al. [63] show that the local intrinsic dimension of adversarial samples is significantly higher than those of benign samples and those that have been perturbed by non-adversarial, such as Gaussian, noise. Thus, they conclude that adversarial samples indeed lie off the distribution manifold and within adversarial subspaces.

As already noted above, we focus on the hypothesis claiming that adversarial examples do not lie on the distribution manifold of the same class and therefore violate the *independent-identically-distributed* (i.i.d.) assumption. Although the violation of this assumptions is certainly not the sole reason for the lack of adversarial robustness it has been reasonably well established and it also valid for robustness against natural perturbations. Nevertheless, as mentioned in Section 2.2.3 (page 26), the relationship between robustness to synthetic and natural perturbations is not straightforward. Therefore, a unifying hypothesis explaining both modes of failure is currently unavailable.

Spurious Correlations

One reason for the lack of robustness and the vulnerability to distribution shifts of neural networks is their reliance on *spurious correlations* in the dataset. Spurious correlations refer to correlations between a non-semantic feature in the input image and its ground-truth label. An example of a spurious feature is provided by Singla et al. [116]for the ImageNet dataset who show that the presence of ‘fingers‘ in an image co-occurs with the label ‘band aid‘ and that neural networks overfit on these. Hence, spurious correlations can break the i.i.d.-assumption if neural networks over-rely on these for their predictions.

Ilyas et al. [64] show that commonly used image benchmarks contain highly-predictive yet brittle features. Neural networks trained on these datasets learn to rely on them and, therefore, are susceptible to small-norm changes of these features. Joe et al. [117] show that these features can be disentangled from the robust ones using a VAE. Several authors provide a related finding for a significant *simplicity bias* of these models [118–121]. Neural networks mostly base their prediction on the simplest feature in the data and ignore more ones that are also predictive of the

label. This remains true even if the complex feature is more predictive of the label than the simple one, and thus simplicity bias can hurt robustness and generalisation. Further, if the simple and the complex feature contradict each other the network still produces high confidence scores[5]. The authors find extreme simplicity bias for fully-connected, convolutional and sequential networks. As a result neural networks are prone to rely on superficial statistical regularities that are not aligned with human perception [122–124]. Singla et al. [116] show that neural networks mostly rely on spurious features for predictions and do not differentiate between core and spurious features during learning.

2.3.2 Training Deficiencies

In the previous section we introduced hypotheses that investigate properties of the dataset and their interaction with the learning process of neural networks as potential sources of robustness deficits. In this section we introduce those works that investigate properties of the classifier itself.

Sample Complexity

In statistical learning theory [125] the goal is to find a classifier f that minimises the risk $R(f) \coloneqq \mathbb{E}_{x \in P(x)}[\mathcal{L}(y, f(x))]$ over the distribution $P(x)$ where $\mathcal{L}$ is a suitable loss function. By definition, the oracle classifier (Definition 5, page 14) is defined as the classifier with the minimum possible risk which represents the conditional distribution $P(y|x)$. In the statistical learning literature the oracle classifier is also commonly referred to as the *Bayes-classifier* f_{Bayes}. Since $P(y|x)$ is generally unknown, the goal of learning is to find f that approximates f_{Bayes}. The sample complexity of a hypothesis class containing f is defined as follows.

Definition 10. ***Sample complexity***
The sample complexity *of a hypothesis class containing the classifier f is the number of train samples necessary to ensure a* probably approximately correct *(PAC) solution, so a solution such that $|R(f) - R(f_{Bayes})| < \gamma$ holds with probability $1 - \eta$ for $\gamma, \eta \in \mathbb{R}$.*

[5]In neural networks the value of the last layer's softmax function is usually taken as the *confidence* of the prediction as it returns values in $(0, 1)$.

In the literature on statistical learning theory *empirical risk minimisation* (ERM) refers to the procedure of finding a classifier f which minimises the empirical risk (loss) over the train samples.

Several studies argue that adversarial training has a larger sample complexity than standard training. Schmidt et al. [126] provide bounds on the sample complexity when the data distribution is a mixture of Gaussians and show increased sample complexity of adversarial training that holds regardless of the training algorithm and the model family. Bhagoji et al. [127] study the sample complexity for distributions that are a mixture of two Gaussians with an approach from optimal transport. Dobriban et al. [128] extended prior analyses to mixtures of three Gaussians in 2- and ∞-norm and Dan et al. [129] derived general results for the case of two-mixture Gaussians for all norms. More recently, Bhattacharjee et al. [130] studied the sample complexity of robust classification for linearly separable datasets. They showed that in contrast to accurate classification, the sample complexity of robust classification has a linear dependence on the ambient dimension $\mathcal{E}$ (Definition 3, page 11). Yin et al. [131] further showed a dependence of the sample complexity on $\mathcal{E}$ for neural networks. Distribution-agnostic bounds for robust classification have been provided by several authors [132–135].

In Section 2.4.2 (page 36) we present works that empirically show that the addition of non-adversarial samples can improve robustness and accuracy, and therefore empirically confirm that robust training has a higher sample complexity than standard training.

In our work we provide a novel reason that explains at least partially why adversarial training has an increased sample complexity. In Chapter 3 (page 56) we first show that the entanglement (Definition 4, page 12) is the leading contributor to the sample complexity. Then, in Chapter 4 (page 82) we show that state-of-the-art robust models learn more entangled decision boundaries which is a result that has previously only been hypothesised for state-of-the-art models and shown for toy-datasets.

Vanishing Gradients

Fundamentally, the reason for vulnerability to small-norm changes is that the classifier does not exhibit flat regions around its train samples, i.e. regions where the loss is approximately constant [136]. Rozsa et al. [137] propose a training scheme to induce flatness around samples and show that this procedure improves robustness and can also improve generalisation performance.

2.3.3 Other Hypothesis

Tanay et al. [138] argue that adversarial examples are the result of an inappropriate placement of the decision boundary compared to the manifold of the training data. They argue that the decision boundary might be tilted along directions of low variance, a result that was later empirically confirmed by Izmailov et al. [139].

With the help of a toy dataset consisting of two concentric spheres embedded in high-dimensional ambient spaces, Gilmer et al. [140] argue that adversarial examples are inevitable for models with non-zero test errors. These adversarial examples can be found on the spheres therefore by the argumentation of Stutz et al. [51] and Ma et al. [63] these are not adversarial examples but generalisation errors.

2.3.4 Conclusions

The reason for the lack of robustness to natural distributions appears to be an over-reliance on spurious correlations in the train set, so an over-reliance on the i.i.d.-assumption. Several authors have shown that these correlations are common in frequently used image benchmarks. However, the reason for vulnerability to adversarial perturbations is not clear. Although part of the vulnerability appears to also stem from the reliance on overly-sensitive and non-semantic features, several studies have proposed and empirically shown that other reasons are valid, too.

In the next section we introduce key studies that propose methods to alleviate the susceptibility of neural networks to synthetic and natural perturbations.

2.4 Robust Training Methods for Neural Networks

To combat neural networks' lack of robustness in object recognition tasks, a large number of potential defence techniques have been proposed. We refer to them as *robust training methods*.

Definition 11. ***Robust Training (RT)***
We define a training method $\mathcal{T} : X \to f$ that, given a dataset X, yields a classifier f. We refer to a training method $\mathcal{T}$ as a robust training method if its goal is to yield a δ-robust classifier as defined in Definition 6 (page 15).

In contrast to robust training methods, *standard training* (ST) methods only aim to reduce the train error and do not focus on the robustness of the classifier. One example for a training method is *empirical risk minimisation* (see Vapnik et al. [125]).

In this section we introduce different methods of robust training and present key works in each of these areas.

2.4.1 Adversarial Training

Szegedy et al. [33] first suggested to append previously generated adversarial examples to the train set. However, solving the minimisation problem in Definition 7 (page 20) is computationally expensive and thus many works developed methods to generate adversarial examples during training. These methods are commonly referred to as *adversarial training*.

Definition 12. ***Adversarial Training (AT)*** *[54, 55]*
Let $\mathcal{L}$ be the loss function of a classier with parameters θ. Adversarial training solves the following minimax-optimisation problem.

$$min_\theta \left[max_\epsilon \mathcal{L}(x + \epsilon, y) \right] \tag{2.9}$$

In other words, the perturbation ϵ is first chosen to maximise the loss $\mathcal{L}$ given the current parameters θ of f. Then, the parameters θ are chosen to minimise the loss on the sample $x + \epsilon$. Adversarial training is one form of robust training (Definition 11, page 35).

To solve the inner maximisation problem in Definition 12 (page 35) multiple different methods have been introduced (see Section 2.2.1, page 19).

Goodfellow et al. [54] propose a method suitable for generating adversarial examples during training, called FGSM (Definition 8, page 21). Later Madry et al. [55] propose an iterative method, called PGD (Definition 9, page 21), that generates stronger examples, though with increased computational cost.

Several authors proposed methods to make adversarial training more computationally efficient. Wong et al. [141], for example, use FGSM but first perturb the original sample with noise. They find that this simple change makes adversarial training with FGSM as effective as training with PGD, though much more computationally efficient. As adversarial training is empirically difficult, Andriushchenko et al. [142] introduce a loss function that aligns the loss' gradient with respect to the original and adversarial example.

Nevertheless, it has frequently been shown that adversarially robust models are still vulnerable to perturbation magnitudes beyond the ones they have been trained on [56].

2.4.2 Data Augmentation and Dataset Enlargement

Since adversarial training is computationally expensive, several authors investigate the effect of *data augmentations* and *dataset enlargement* on robustness. Data augmentations are generated by applying a function to an image that is cheap to evaluate to introduce some desired inductive bias to the model. Dataset enlargement on the other hand, refers to the gathering of novel samples without changes to the original ones. Data augmentation and enlargement cannot usually be clearly distinguished as augmented images are usually appended to the dataset, therefore also enlarging it.

While additional data can improve robustness against natural perturbations [143], several authors find that data augmentation alone does not improve adversarial robustness [144–146]. However, as the set of augmentations gets diverse enough, using data augmentations without additional samples improves adversarial robustness. Hendryck's et al. [147], for example, propose a data augmentation method called *AugMix* that mixes several different augmentation techniques and find im-

proved accuracy and calibration without using additional data. Later, Rebuffi et al. [148] demonstrate that data augmentations can improve adversarial robustness when combined with weight averaging.

Independent of each other several authors find that using additional data during training improves adversarial robustness and standard accuracy [149–152], a finding that empirically confirms the higher sample complexity of robust training mentioned in Section 2.3.2 (page 32). Further, Gowal et al. [153] show that using samples generated by a generative model trained on the original train set can also improve robustness and accuracy without the need for novel samples from outside the original train set.

2.4.3 Dimensionality Reduction

Since synthetic and natural perturbations do not change the ground truth class label (Definition 7, page 20), they leave semantically meaningful features in images unchanged. The goal of dimensionality reduction techniques is to describe a sample with as few variables as possible while keeping its contained information as large as possible. Naturally, dimensionality reduction primarily removes semantically non-meaningful features when applied to images (e.g. [154]).

Several different dimensionality reduction techniques have been proposed for various types of data (see Sorzanoet et al. [155] for an overview). *Autoencoders* are a form of non-linear dimensionality reduction technique that can be applied with moderate success to image datasets [154]. They map its input to its output through a series of layers with a bottleneck-layer of lower dimensionality in the middle. This bottleneck-layer presents a compressed representation of the input data.

Autoencoders are effective in removing Gaussian noise from datasets [156] and have also been a popular choice to defend against adversarial perturbations [157]. Sahay et al. [158] find that autoencoders are successful in removing adversarial perturbations. Zizzo et al. [159] also use an autoencoder to project data into a lower-dimensional space. However, they also utilise the k-nearest neighbours algorithm to detect adversarial examples in that space. A straightforward idea is to stack an autoencoder to remove adversarial perturbations before a neural network, however Gu et al. [62] report that the resulting architecture is even more vulnera-

ble to adversarial examples, possibly because of the unsupervised objective of the autoencoder that omits information about discriminative features in the input.

Sanayal et al. [160] propose a method that regularises the rank of the representation matrices in neural networks. Thus, representations are constrained to lie on low-dimensional subspaces. They find that such regularisation moderately improves generalisation performance and greatly improves adversarial robustness. The intuitive reason why such regularizer works is again that it forces the networks to represent the data by its most discriminative features and removes noise.

In Chapter 5 (page 112) we show that the opposite implication of the results of Sanayal et al. [160] also holds. State-of-the-art robust training methods induce lower-dimensional representations despite not being explicitly regularised to do so. Thus, lower dimensionality is empirically strongly correlated with robustness which partially explains the success of dimensionality reduction techniques in removing adversarial noise. Further, it might offer a direction for future work on robust training methods.

2.4.4 Detection of Out-of-distribution Samples

Given the difficulty of training models to be robust against adversarial attacks [56], some works focused on the detection of adversarial examples instead. There are a wide variety of detection methods, all using different properties of adversarial examples to detect them. For an overview of these algorithms we refer the reader again to the survey of Serban et al. [52]. However, Tramer et al. [161] argue that from a computational point of view detecting adversarial examples is almost as challenging as correctly classifying them.

Using auxiliary methods to detect any out-of-distribution samples is an orthogonal problem to obtaining robust representations. It does not solve the robustness issue and only circumvents it. In Section 2.4.10 (page 43) we highlight some works showing that robust representations have benefits beyond their robustness to out-of-distribution samples. Thus, in this thesis we only consider methods that directly aim for robust representations.

2.4.5 Removing spurious Correlations

As mentioned in Section 2.3.1 (page 30) the presence of non-robust but predictive features in common image benchmarks is a known reason for the lack of robustness. Ahmed et al. [162] study different group robustness methods used to mitigate the reliance on spurious features and find improved performance on samples where these features are removed. Izmailov et al. [163] show that in general standard training induces good representations. When used in combination with re-training of the last layer on datasets where the spurious correlation is broken, standard training can yield representations that are competitive with specialised group robustness algorithms.

As mentioned in Section 2.2.2 (page 23) texture and background are known spurious correlations present in common image benchmarks. Thus, some authors proposed novel datasets with these spurious correlations removed, such as the *Stylized-ImageNet* [69] or the *Waterbirds* [164–166] dataset where water- and land living-birds are placed mostly in front of either land or water scenes to create a spurious background clue.

2.4.6 Regularisation

In the language of statistical learning theory, regularisation refers to the process of restricting a hypothesis' class capacity. In this section we introduce literature that can be summarised as using some form of regularisation method to restrict the hypothesis' class to encompass robust models.

Gradient Regularisation

The *input-output Jacobian* $\mathcal{J} \in \mathbb{R}^{y \times \mathcal{E}}$ of a model $f : \mathbb{R}^{\mathcal{E}} \to \mathbb{R}^{y}$ is the matrix of row-wise stacked gradients $\nabla_x \mathcal{L}$ for every label in y, where $\mathcal{L}$ is the loss function. The Jacobian is frequently employed to explain predictions of neural networks [167].

Tsipras et al. [82] recognised that the Jacobians of adversarially trained neural networks are human-interpretable, so contain semantic features of the corresponding image, in contrast to standard trained network's Jacobians which resemble noise to human observers. Etmann et al. [168] show that these interpretable Jacobians

are not only a byproduct of adversarial training but a general property of robust classifiers.

Based on these previous observations, several works improve robustness by regularising a network's Jacobian. Ross et al. [169] use *double backpropagation* [170] which regularises the 2-norm of the model's gradients and report improved robustness. Chan et al. [171] utilise a generative adversarial network to train a classifier to produce salient Jacobians that resemble the input image. They also report improved robustness. Chan et al. [172] also use a generative adversarial network. This time, however, they utilise a robust teacher model and train a student network such that its gradients are semantically similar to the teacher's as judged by the discriminator model. Simpson et al. [173] also regularise the input gradients and find improved robustness. Du et al. [174] use two different methods to improve the interpretability of gradients. First, they apply fused Lasso regularisation on the saliency map and secondly they use the cosine similarity between the input gradient and the image contour. They find that both models improve adversarial robustness and saliency map interpretability.

Lipschitz Regularisation

A function f is called *Lipschitz-continuous* in some p-norm with Lipschitz constant L if for all inputs x_1, x_2 the following Equation holds.

$$||f(x_1) - f(x_2)||_p \leq L||x_1 - x_2||_p \tag{2.10}$$

The Lipschitz constant of a neural network is related to its robustness as its value describes how much changes in the input can affect the output. Therefore, estimating and regularising the Lipschitz constant of neural networks has been the focus of some works. Tsuzuku et al. [175] propose an algorithm to compute and restrict the Lipschitz-constant during training, yielding a level of certified adversarial robustness. Scaman et al. [176] propose an algorithm to upper bound the Lipschitz constant for neural networks and show that this estimation is usually an NP-hard problem. Anil et al. [177] propose an algorithm to train neural networks with Lipschitz constant $L = 1$ and Cisse et al. [178] propose a network architecture with Lipschitz-constant

smaller than one.

Restricting the Lipschitz-constant of neural networks has also been shown to improve interpretability of predictions [55] and generalisation performance [179].

Early Stopping

Early stopping halts training of a neural network before the train error reaches zero and once a user-specified metric is fulfilled. Thus, it restricts the hypothesis space to those found early during training and fulfilling the aforementioned metric.

Rice et al. [145] find that while continued standard training does not hurt standard test performance, continued adversarial training does hurt robust test performance. They call this phenomenon *robust overfitting* and show that it can be mitigated by early stopping. This leads to adversarial training being on par with more recent robust training methods.

2.4.7 Margin Maximisation

The *margin* of a classifier refers to the minimum distance to the decision boundary around any sample. Thus, it also describes the maximum perturbation magnitude that can be applied before changing the prediction.

Naturally, several works introduce training objectives that maximise the margin around train samples [180–183] through various different objective functions and find improved robustness.

2.4.8 Other Methods

The previously mentioned methods of robust training have gained the most interest in the literature. Nevertheless, several other methods have also been proposed. In this section we briefly mention some of them. For a complete overview we refer the reader to the survey by Serban et al. [52].

Compression

Both at training and inference time neural networks are computationally expensive. Further, since state-of-the-art architectures have tens or even hundreds of millions of

parameters, their memory requirement is also extensive. These are limiting factors for deploying models on phones and other pieces of hardware. Thus, several different methods of model compression have been introduced. These methods can be broadly separated into *quantisation* and *pruning*. Quantisation has been proposed as one method to mitigate memory and compute requirements. A quantised neural network does not use the 32-bit floating point format for storing its weights but integer [184] or even binary numbers [185]. On the other hand, pruning refers to the process of removing certain parameters completely. Quantising and pruning trained neural networks is possible as they have been found to contain redundant parameters that can be removed without degrading generalisation performance. For a survey of quantisation and pruning of neural networks we refer the reader to the survey by Liang et al. [186].

Several works have show that it is possible to obtain networks with smaller parameter counts and higher robustness than standard trained ones [187, 188]. The reported success of these methods is noteworthy as previous works find that larger network architectures improve robustness [79] and Lin et al. [189] find that quantised neural networks are more vulnerable to adversarial perturbations than standard trained ones. The authors argue that is due to an *error amplification effect* that increases the adversarial noise beyond quantisation thresholds along the model's depth. To circumvent error amplification, the authors propose a quantisation method to control the Lipschitz constant of the model which improves robustness.

Distillation

Neural networks are commonly trained on labels encoded as one-hot vectors that are zero everywhere except at index i where they are one. However, at inference time neural networks return a probability distribution by applying the softmax-function that does not return one-hot vectors.

Hinton et al. [190] show that training a *student* neural network with a smaller capacity on the soft-labels produced by a larger *teacher* model can induce similar or even better generalisation performance on benign samples. Papernot et al. [191] revisit this idea but keep the architecture capacities between teacher and teacher network equal. They show that the soft-labels produced by a robustly trained teacher

network induces robust representations in the student network as well.

Pre-training

Unsupervised pretraining refers to an initialization technique of neural network parameters in which every layer is first pretrained in an unsupervised fashion, usually by minimising some reconstruction loss as used in autoencoders. With the development of novel initialisation schemes of neural network parameters, however, the need for pretraining to achieve good generalisation performance is not longer given [192].

Hendrycks et al. [193] investigate the benefit of pre-training when evaluated against label corruption, class imbalance and adversarial attacks. They find that although pre-training is not necessary for good in-distribution generalisation [194], it improves the aforementioned metrics.

Pre-processing

Kurakin et al. [195] process adversarial images with devices like cellphone cameras and recognise that the rate of misclassification of the processed images remains high.

2.4.9 Verifying Robustness

Usually, robust training methods only empirically guarantee robustness, i.e. they statistically test the robustness against a finite amount of samples. However, in some application areas this robustness estimate is not sufficient and methods for the *verification* of the robustness against all samples within a user-given domain have been proposed [135, 196–199]. Balunovic et al. [200] and Casadio et al. [201] proposed to combine verification and adversarial training. During training the network is verified for wrong predictions and if those are found they are added to the train set.

2.4.10 Properties of Robust Representations

Adversarially robust representations exhibit several desirable properties beyond robustness to out-of-distribution perturbations.

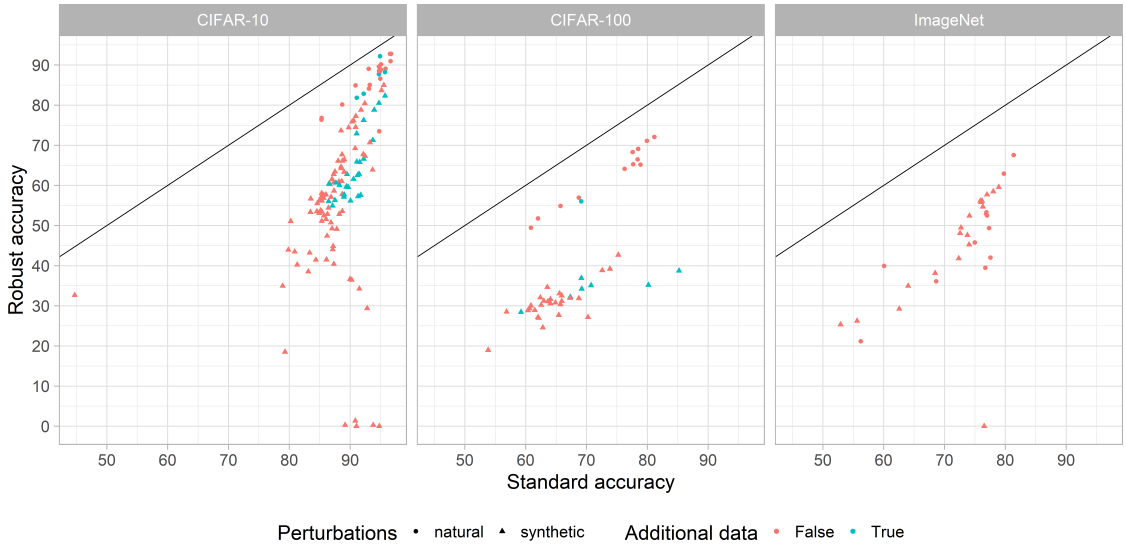


Figure 2.6: Standard and robust accuracy for all robustly trained models retrieved from *RobustBench* [206]. The black line describes the configuration where robust and standard accuracy are equal. For CIFAR-10 robust accuracy and standard accuracy only display a small gap, however, for more complex benchmarks the robust accuracy still lacks the standard accuracy by a significant margin.

As noted in Section 2.4.6 (page 39), the Jacobians of standard trained neural networks usually resemble noise to human observers. First noted by Tsipras et al. [82], Jacobians of adversarially trained neural networks, however, exhibit salient characteristics such that they are humans interpretable. Thus, robust models are generally more explainable [202] than standard trained ones and are also better calibrated [203]. Further, using the representations learned by adversarially robust models for transfer learning leads to superior performance [204, 205].

2.4.11 Conclusions

As of 2023 over 6,000 papers (see Carlini et al. [18]) have been published on the topic of adversarial robustness alone. These works include studies that describe failure cases of neural networks as well as possible reasons and methods to mitigate them. In Figure 2.6 (page 44) we display the gap between generalisation (*standard accuracy*) and robustness (*robust accuracy*, evaluated by *AutoAttack* [207]) for recent robust training method retrieved from *RobustBench* [206]. One can observe that especially for larger datasets, such as CIFAR-100 [23] and ImageNet, robust accuracy trails standard accuracy by a large margin. In addition, the progress has

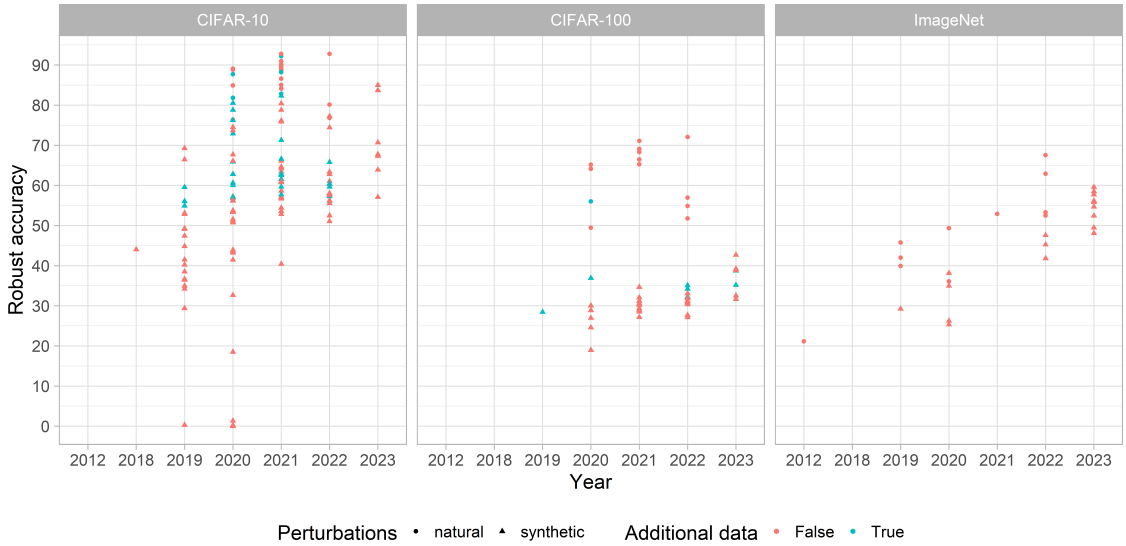


Figure 2.7: Robust accuracy for all robustly trained models retrieved from *RobustBench* [206] ordered by publication year. For all image benchmarks progress with respect to robust accuracy has been small over recent years.

Table 2.2: Geometric properties of data distributions and their neural network representations and the context in which they have been studied. A network's i-th layer representations is denoted as $f_i(X)$. The symbol **x** denotes that no studies have been conducted in that area. The notation Cha.$\cdot$ denotes the chapter of this thesis that studies this topic.

Geometric property	**studied for / with respect to**			
	Datasets X	Representations $f_i(X)$	Robustness of f	Sample complexity of f
Extrinsic Dimension (Sec. 2.5.1)	**x**	[208, 209]	[54, 210–212]	[12, 16, 213]
Intrinsic Dimension (Sec. 2.5.2)	[16]	[214–218]	[63, 160, 219]	[16], Ch. 3
Entanglement (Sec. 2.5.3)	[12]	[220–223]	[86, 220], Ch. 4	[12], Ch. 3
Linearity (Sec. 2.5.4)	**x**	[214, 221, 224]	[225, 226]	**x**
Manifold Capacity and Separation (Sec. 2.5.5)	[17]	[223, 227]	[17, 223], Ch. 4	**x**
Support Vectors (Sec. 2.5.6)	[228]	**x**	**x**	**x**

been slow as one can observe in Figure 2.7 (page 45). Thus, as of today, robustness to out-of-distribution samples is still an open problem [11] and even recent architectural innovations such as Transformers do not solve this issue (see Section 2.2.4, page 27). As such the robustness issue of neural networks is far from being solved and with the continued integration of these models into the real world both as standalone algorithms and as part of other algorithms, further attention to this topic is warranted.

2.5 Geometric Properties of Data Distributions and Representations

After introducing the literature on robustness deficits, their reasons as well as proposed methods to mitigate them in breadth, we give in this section an in-depth overview of the literature that studies the influence of the data's geometric properties on neural networks and the geometric properties of deep neural networks' data representations with respect to their robustness and sample complexity.

In Table 2.2 (page 45) we provide an exhaustive list of papers that study geometric properties with respect to data distributions, representations and neural network robustness and sample complexity. In the following we describe the literature presented in this table in detail.

2.5.1 Extrinsic dimension

The extrinsic dimension of a distribution is equivalently to the ambient space of the manifold near which the distribution concentrates. In the case of images, for example, the extrinsic dimension is the number of pixels and colour channels (Definition 3, page 11).

Extrinsic Dimensions of Representations $f_i(X)$

The extrinsic dimension of a neural network's data representation is simply the width of the particular layer. Nguyen et al. [209] study how the width of deep neural network classifiers affects the decision boundary. They show theoretically that if the width across layers is smaller than the input dimension, the neural network learns a connected decision region and that disconnected decision regions can only be learned if the width of at least one layer is greater than the input dimension. Previously, Fawzi et al. [208] show empirically that the *CaffeNet* architecture [229] trained on ImageNet learns connected decision regions. So, for every pair of images there is a, not necessarily linear, path between them along which all points are classified by the same label as the two images.

Extrinsic Dimensions and Robustness of f

Based on the study of a linear model, Goodfellow et al. [54] predict a dependence of adversarial vulnerability on the input data's dimension. Gilmer et al. [212] study adversarial examples for deep neural networks for a toy-dataset consisting of two concentric spheres embedded in a high dimensional ambient space. Theoretically, they show that for any model that has a non-zero test error, adversarial examples exist on the sphere. Further, they report for such models that the adversarial vulnerability increases with the dimensionality of the ambient space. A similar result for data lying within a unit-hypercube or unit-sphere was reached by Shafahi et al. [211]. Gabriel et al. [210] study the extrinsic dimension's effect on the adversarial vulnerability of deep neural networks for real-world image datasets. They find that with increasing image sizes the norms of input-output gradients increase which is associated with increased adversarial vulnerability.

Extrinsic Dimensions and Sample complexity of f

Narayanan et al. [12, 13] study the sample complexity (Definition 10, page 32) of empirical risk minimisation when the data distribution concentrates near a manifold in a binary classification setting. They show that the sample complexity only depends on the intrinsic dimension of the manifold and the entanglement of the two classes and not on the extrinsic dimension. Empirically, Pope et al. [16] investigate the sample complexity of deep neural network classifiers and find that their sample complexity is not affected by the extrinsic dimension of common image benchmarks when it is varied by nearest-neighbour interpolation. However, it is commonly known that real-world image benchmarks have backgrounds that are highly predictive of the class label [44, 45] and nearest-neighbour interpolations do not change their predictability. Hence, D'Amario et al. [213] study the effect of the extrinsic dimension for backgrounds that are either predictive or not predictive of the label. They find that predictive backgrounds do not affect the sample complexity, confirming the results of Pope et al. [16], while additional extrinsic dimensions that are not predictive of the label increase the sample complexity.

2.5.2 Intrinsic dimension

The intrinsic dimension of a manifold denotes the number of variables that are required to describe the distribution's variations (Definition 2, page 11).

Intrinsic Dimensions of Dataset X

It is a common assumption in machine learning that the intrinsic dimension of natural data is much lower than its ambient dimension [2] and Pope et al. [16] empirically confirm this for several real-world image benchmarks.

Intrinsic Dimensions of Representations $f_i(X)$

Basri et al. [215] demonstrate that deep neural networks can represent data that concentrates near manifolds with low error and by using a number of parameters that is almost optimal.

Ansuni et al. [214] study the intrinsic dimension of hidden layer representations for standard trained networks without label noise. They show that the intrinsic dimension increases in earlier layers indicating a removal of image features such as luminance or contrast that are irrelevant for classification. Following that early increase in intrinsic dimension is a progressive decrease which signals some sort of feature selection of the network. Further, they show that the intrinsic dimension in the last hidden layer is negatively correlated with the generalisation error of the model. A similar finding was independently reported by Recanatesi et al. [218]. Following these works, Brown et al. [217] investigate the intrinsic dimension of hidden representations for networks regularised by either dropout [230] or weight decay and their connection to the generalisation performance. They show that in these networks, increased generalisation performance co-occurs with a decrease in last-layer intrinsic dimension and an increase in peak intrinsic dimension. Brown et al. [217] interpret these findings as the last-layer intrinsic dimension being a proxy for the *simplicity* of the model and as the peak intrinsic dimension being a proxy for the *plausibility* of the model. Thus, both quantities need to be balanced to ensure sufficient generalisation performance.

Gong et al. [216] find that intrinsic dimensions of hidden representations for image datasets are low compared to their ambient space.

In Chapter 5 (page 112) we provide results that show that the connection between the last-layer intrinsic dimension and the generalisation performance [214, 217] does not hold for robustly trained models. Further, the peak intrinsic dimension across layers is negatively correlated with the robustness and thus it is cannot be viewed as a proxy for the model's plausibility as suggested by Brown et al. [217]. Therefore, studying the geometric properties of robust representations yield different and sometimes contradicting results to studies of standard (non-robust) representations.

Intrinsic Dimensions and Robustness of f

Ma et al. [219] investigate the intrinsic dimension of neural networks' hidden representations learned with varying degrees of label noise. They find that during training correctly labelled samples decrease the intrinsic dimension whereas falsely labelled ones increase it. Based on this observation they propose a regularizer to mitigate the effect of label noise on training by monitoring the intrinsic dimension of hidden representations.

Further, Ma et al. [63] show that the local intrinsic dimension around a sample can be used as a method to detect adversarial examples. They report that, compared to noisy and original samples, adversarial examples display significantly higher estimated local intrinsic dimensionality.

Sanyal et al. [160] introduce a regularizer that forces the rank of activations matrices, and thus the intrinsic dimension of representations, to be small and report increased adversarial robustness.

As already noted, in Chapter 5 (page 112) we show the opposite implication of the results of Sanyal et al. [160], namely that state-of-the-art robust training methods reduce the intrinsic dimension despite not being regularised to do so.

Intrinsic Dimensions and Sample Complexity of f

Pope et al. [16] study the intrinsic dimension's effect on the sample complexity of deep neural networks. They utilise several complex image datasets and show that with increasing intrinsic dimension the sample complexity increases as well.

In Chapter 3 (page 56) we propose a crucial extension to the results by Pope et al. [16]. Whereas the authors only utilise complex image benchmarks with high

levels of entanglement, we show that the intrinsic dimensionality's influence on the sample complexity depends on the given level of entanglement. For low levels of entanglement increases in intrinsic dimension, if at all, only marginal affect the sample complexity, and thus neural networks behave more like support vectors classifiers in low-entanglement regimes. Thus, when studying the intrinsic dimension's influence on the sample complexity, the entanglement needs to be considered as well.

2.5.3 Entanglement

The entanglement of a distribution refers to the number of connected hyperplanes required to separate the classes in their original ambient space (Definition 4, page 12).

Entanglement of Representations $f_i(X)$

The *soft nearest-neighbour loss* [231] (SNNL) measures the entanglement of points with different labels (the SNNL is display in Equation 5.12, page 130). The lower the loss, the better separated the classes are. Frosst et al. [220] use the SNNL to entangle the hidden representations of image data in deep neural networks and find increased generalisation performance. They hypothesise that maximising the SNNL encourages learning of class-independent features and yields better similarity structures.

Brahma et al. [221] study data representations of toy- and face recognition datasets along neural network hierarchies and find that those are progressively untangled and flattened. Hauser et al. [222] report a similar result. Contrary, Ansuini et al. [214] study real-world image benchmarks and do not find a flattening of the hidden representations. Instead, they argue that the untangling across network hierarchies is achieved by reduction of the intrinsic dimension.

Dapello et al. [223] find that neural networks develop separable representations in later layers and that the linear separability of the penultimate layer is predictive (positively correlated) of the top-1 accuracy across models. Increasing the magnitudes in adversarial attacks makes manifolds larger, more entangled and less linearly separable which leads to reduced performance.

In Chapter 5 (page 112) we use the soft nearest-neighbour loss to disentangle representations of semantically dissimilar classes and find that this reduces gener-

alisation performance and robustness. We show that the key to robustness is not the class-agnostic concentration, so the spatial entanglement, of representations but learning aligned features across semantically similar classes.

Entanglement and Robustness of f

Sanyal et al. [86] show for a toy dataset in two dimensions that standard training does not result in decision boundaries that provide a sufficiently large margin for adversarial robustness. Adversarial training, on the other hand, results in a larger margin and in a geometrically more complex decision boundary. In addition to an improvement in generalisation performance, Frosst et al. [220] find that adversarial training [55] for the MNIST [19] dataset increases the entanglement of hidden representations when measured with the soft-nearest neighbour loss [231].

In Chapter 4 (page 82) we show for real-world datasets, that robust neural networks learn geometrically more complex decision boundaries, therefore complementing the result of Sanyal et al. [86]. We also provide an upper bound over which provably a geometrically more complex decision boundary is required.

Entanglement and Sample Complexity of f

Narayanan et al. [12] theoretically show that the sample complexity of empirical risk minimisation for binary classification depends positively on the entanglement of the classes.

Despite being trained by empirical risk minimisation, some behaviours of neural networks are not always consistent with the predictions of empirical risk minimisation. One example is their ability to generalise despite having perfect sample expressivity [14, 15].

In Chapter 3 (page 56) we investigate the entanglement's effect on the sample complexity of deep neural networks and find that the entanglement is by far the leading contributor.

2.5.4 Linearity

We refer to a representation as *linear* if and only if all linear combinations

$$\tilde{x} \coloneqq \alpha x_0 + (1 - \alpha) x_1 \tag{2.11}$$

for $\alpha \in [0, 1]$ are of the same label as x_0 and x_1, i.e. the oracle (Definition 5, page 14) classifies all three samples with the same label, so $\mathcal{O}(\tilde{x}) = \mathcal{O}(x_0) = \mathcal{O}(x_1)$.

Linearity of Representations $f_i(X)$

Brahma et al. [221] study a toy dataset and derive a measure for the linearity of their representations, so the curvature of the manifold. They find progressive manifold disentanglement along the hierarchy of deep neural networks. Contrary, Ansuini et al. [214] study the linearity of representations obtained by standard training on common real-world image benchmarks. They do not find a meaningful flattening of the representations but a reduction of intrinsic dimensionality. Thus, whether manifolds are disentangled via flattening or by reduction of their intrinsic dimension appears to be dataset-specific.

Toosi et al. [224] study the representations of natural video frames learned by adversarially robust models. They find that chronologically close frames are represented on almost linear subspaces. Thus, manifold flattening may not be a property of standard trained networks, but of robust networks it is.

Linearity and Robustness of f

Mixup [225] is a simple regularizer that minimises the loss not on raw input samples but on weighted linear combinations of both input samples and labels. The authors report an increased robustness to adversarial attacks when mixup is used. Verma et al. [226] propose *Manifold Mixup* as an extension to (input) Mixup [225]. They show that minimising the loss on linear combinations on hidden representations and their label further improves robustness to adversarial attacks.

2.5.5 Manifold Capacity and Separation

The *manifold capacity* $\mu = |\mathcal{M}|/\mathcal{E}$ is the maximum number of manifolds that can be linearly separated in $\mathcal{E}$ dimensional space. This quantity is estimated using *replica mean-field theoretic manifold analysis* [25, 227, 232] (MAFTMA). The *manifold separation* is defined as follows.

Definition 13. ***Minimum nearest-neighbour distance*** *[17]*
The manifold separation is defined as the minimum nearest neighbour distance in between any two samples $x_i, x_j \in X$ with labels y_i, y_j, respectively,

$$R = min_{y_i,y_j}||x_i - x_j||_p \tag{2.12}$$

in a user-specified p-norm.

Capacity and Separation of Dataset X

Yang et al. [17] propose the minimum nearest-neighbour distance as described in Definition 13 (page 53) and compute its value for common image benchmarks.

Capacity and Separation of Representations $f_i(X)$

Cohen et al. [227] show that along the layers of a trained deep neural network the manifold capacity μ increases and Dapello et al. [223] find that the manifold capacity of the penultimate layer is predictive (positively correlated) of the top-1 accuracy across models.

Capacity, Separation and Robustness of f

Yang et al. [17] find that common image benchmarks are well enough separated in input space for common perturbation magnitudes used in robust training. Dapello et al. [223] find that adversarial perturbations of increasing strength reduce the manifold capacity in the penultimate layer.

In Chapter 4 (page 82) we provide a novel measure for manifold separation that bounds the perturbation magnitude over which a provably geometrically more complex decision boundary is required. For real-world image benchmarks our measure further provides a lower-bound over which label noise is introduced and we show

that training with magnitudes above this bound reduces generalisation performance and robustness to natural perturbations. Finally, we show that the measure by Yang et al. [17] over-estimates the robust radius of common real-world image benchmarks and that our bound is a more appropriate measure of the robust radius.

2.5.6 Support Vectors

Support Vectors of Dataset X

Jimenez et al. [228] find for CIFAR-10 that re-positioning only a small fraction of samples can result in a large change in the shape of the decision boundary. Thus, neural networks are similar to support vector machines as their decision boundary is also defined by the position of a few support vectors.

2.5.7 Conclusions

Looking through the lens of representational geometry to investigate the properties of neural networks is an established research area. It provides insights into the feature learning process, the robustness and the generalisation process of deep neural networks. Further, the methods developed in this area are also applied in neuroscience to derive insights into the human visual system.

There are some important gaps within this literature when it comes to the connections between representational geometry, robustness and sample complexity. In the following section we describe these gaps which motivate the contributions presented in this thesis.

2.6 Our Work

The contributions presented in Chapter 3 (page 61) fit into the robustness-column of Table 2.2 (page 45). Neural networks do not only always follow the predictions made by statistical learning theory (e.g. [14]) and thus studying the influence of manifold properties on them is a necessity. Pope et al. [16] do so and confirm that the intrinsic dimension indeed positively influences the sample complexity for real-world image benchmarks whereas the extrinsic dimension has no influence on

it. D'Amario et al. [213] refine this result by showing that uninformative extrinsic dimensions can hurt the sample complexity. In our work presented in Chapter 3 (page 61) we also introduce the entanglement of class manifolds in the analysis of the sample complexity which both Pope et al. [16] and D'Amario et al. [213] omit. We find that the entanglement's influence on the sample complexity is far greater than the intrinsic dimensionality's and further that the intrinsic dimension's influence depends on the level of entanglement. For low levels of entanglement increasing the intrinsic dimension does not affect the sample complexity. As both Pope et al. [16] and D'Amario et al. [213] only study real-world image benchmarks with high entanglement they could not have obtained such a finding.

In Chapter 4 (page 82) we show that state-of-the-art robust models indeed learn geometrically more complex decision boundaries, a result that has not yet been shown empirically for real-world datasets and architectures. Thus, taken together, we can state that indeed one reason for the observed greater sample complexity of robust training is learning of geometrically more complex decision boundaries.

In Chapter 5 (page 112) we study the geometry of hidden representation and show that the intrinsic dimension of hidden representations is consistently smaller for robust models. Thus, despite not being explicitly regularised to do so, dimensionality reduction is a key mechanism of robust training. We show that this dimensionality reduction is partially obtained by a sharing of features between semantically similar classes.

Chapter 3

The Effect of Manifold Entanglement on Learning in Neural Networks

3.1 Introduction

It is commonly assumed that distributions of natural data, such as images and videos, concentrate near or lie on low-dimension manifolds embedded in higher dimensional ambient spaces [2]. For common image benchmarks this assumption was empirically shown to be valid [16] and is the cornerstone of many algorithms in machine learning (e.g. [154, 226, 233]).

One can describe a distribution concentrating near a manifold with three key geometric properties, namely its intrinsic dimension $\mathcal{I}$ (Definition 2, page 11), its extrinsic dimension $\mathcal{E}$ (Definition 3, page 11) and, in classification settings, the entanglement Σ (Definition 4, page 12) of different classes. The intrinsic dimension is the dimension of the data manifold's tangent space which describes the degrees of freedom of movement along its surface. These degrees of freedom correspond to the number of variables required to describe the distribution. In the case of image data, for example, the intrinsic dimensions correspond to label-preserving transformations. The most common examples for such transformations are scaling, rotation and translation of images but can also be various changes in image backgrounds and illumination (see Figure 1.1, page 2 for illustration). The extrinsic dimension de-

scribes the size of the manifold's ambient space. In the case of images, the ambient space is the space of pixels and colour channels. The entanglement can intuitively be viewed as the number of connected hyperplanes that are necessary to separate the individual classes in the original ambient space. If the classes are linearly separable, only a single hyperplane is required.

The sample complexity (Definition 10, page 32) refers to the number of samples required to learn a model that approximates the Bayes-classifier or oracle classifier (Definition 5, page 14) within user-specified bounds. It has been shown that the sample complexity of a classifier trained with empirical risk minimisation that maps samples from class manifolds to an associated label space is dependent on the manifold's intrinsic dimension and the entanglement but is independent of the extrinsic dimension [12, 13]. In other words, the learning difficulty of a classification problem is proportional to the entanglement and the intrinsic dimension of the associated manifolds.

Although current neural network classifiers are trained to minimise the risk, some of their properties cannot be explained by classical statistical learning theory. One example is their ability to generalise well to an unseen test set despite exhibiting perfect train sample expressivity [14, 15]. However, classical statistical learning theory predicts that classifiers with the capacity to memorize the train samples will not deduce abstract features shared by the examples, i.e. learn, but instead merely memorize the train samples. Memorizing the train samples implies chance performance on test set samples. It is common practice, however, to train heavily over-parametrized neural networks until they achieve zero train loss and observe continuously increasing generalisation performance in the meanwhile [145].

In this section we investigate whether the theoretically predicted effects of manifold entanglement on the sample complexity for empirical risk minimisation also hold for deep neural network classifiers. As neural networks are state-of-the-art algorithms for a variety of tasks, such as image classification [9, 10] and segmentation [234, 235], they are often confronted with distributions that exhibit highly complex geometries in their original embedding spaces. This study complements and expands a recent work by Pope et al. [16] who show that common image benchmarks concentrate near low-dimensional manifolds and find that increases in their intrinsic

dimension positively influence the sample complexity of neural networks. We expand their findings by also considering the effect of the distribution's entanglement. Including the entanglement in the analysis of the sample complexity is interesting for two reasons. Firstly, current neural network architectures employ a linear classifier as the last layer which is commonly the softmax function. So, manifolds that are not linearly separable in their original ambient space need to be disentangled before the softmax function is applied [221, 222]. Secondly, the operation conducted within the human ventral visual stream can also be viewed as a process of disentangling manifolds [1]. Thus, as the sample complexity can intuitively be viewed as the learning difficulty, studying the entanglement's effect on it is relevant for both practical applications of artificial neural networks and investigations into biological neural networks.

We conduct two different sets of experiments. First, we experiment with different artificial datasets for which we can control the intrinsic and extrinsic dimension and the entanglement precisely and measure the sample complexity of fully-connected networks. Then, we expand the analysis to convolutional neural networks trained on real image benchmarks. In both experiments we find that entanglement has by far the strongest positive influence on the sample complexity. Further, we find that the influence of the intrinsic dimension depends on the level of entanglement of a distribution, a result that has not been show empirically for neural networks [16]. So, increases in the intrinsic dimension affect the sample complexity to a greater extent for a distribution with high entanglement than a distribution with low entanglement. Thus, we can confirm that previous theoretical results also hold for neural networks. In addition, we find that intrinsic dimension and entanglement cannot be studied in isolation because of their interdependent effect on the sample complexity.

This chapter is structured as follows. In Section 3.2 (page 59) we describe the methodology of our study. Section 3.3 (page 62) and Section 3.4 (page 73) describe the results for artificial and real-world datasets, respectively. Finally, in Section 3.5 (page 80) we discuss the findings and their practical implications.

3.2 Methodology

3.2.1 Regression Models

As described in Section 2.3 (page 30), the sample complexity of a hypothesis class describes the number of samples required to learn a PAC solution to a learning problem. We approximate the sample complexity by the number of samples from the train set that are required to reach a certain test accuracy threshold. This approach is similar to Pope et al.'s [16] method of empirically determining the sample complexity of a neural network classifier.

We gather an approximation of the sample complexity ι with respect to changes in entanglement Σ, intrinsic $\mathcal{I}$ and extrinsic $\mathcal{E}$ dimensionality and investigate three different regression models to estimate their relationship. Each regression model constitutes a different hypothesis about the influence of the distribution's geometric properties on the sample complexity and their potential interdependence. Estimating regression models allows us to quantify and statistically test any relationship between variables which is not possible by the simple visual investigation of graphs that was done by Pope et al. [16]. Further, the goodness of fit, as measured by the adjusted R^2, is an indicator for the respective validity of the regression models in comparison to each other.

In the following paragraphs we describe the three used regression models and explain their hypothesis about the relationship between the data distribution's geometric properties and the sample complexity of a neural network classifier.

I. No Interdependencies between Geometric Properties

The first hypothesis is written as

$$\iota = c + \alpha\Sigma + \beta\mathcal{I} + \gamma\mathcal{E} \tag{3.1}$$

where α, β and γ are the regression coefficients to be estimated and c is the regression constant. Equation 3.1 (page 59) describes a simple model in which the geometric properties Σ, $\mathcal{I}$ and $\mathcal{E}$ are assumed to be independent of each other. In other words, no single geometric property's influence on the sample complexity depends on the

value of another geometric property.

II. Interdependencies between Geometric Properties

Expanding Equation 3.1 (page 59) by interaction terms between geometric properties yields

$$\iota = c + \alpha\Sigma + \beta\mathcal{I} + \gamma\mathcal{E} + \delta(\Sigma \cdot \mathcal{I}) + \epsilon(\Sigma \cdot \mathcal{E}) + \zeta(\mathcal{I} \cdot \mathcal{E}) \tag{3.2}$$

where the lower-case Greek letters denote again the regression coefficients to be estimated. In Equation 3.2 (page 60) the assumption of independent geometric properties is discarded. The added interaction terms estimate whether the influence of one geometric property on the sample complexity depends on another property's value.

III. Interdependence between Intrinsic Dimension and the Level of Entanglement

Finally, Equation 3.2 (page 60) introduces the entanglement as a categorical variable

$$[\Sigma^{(\sigma)}] = \begin{cases} 1, \text{ if } \Sigma = \sigma \\ 0, \text{ otherwise} \end{cases} \tag{3.3}$$

that indicates which level of entanglement σ is present in the given dataset. For example, if in a binary classification task the classes are linearly separable, then $\sigma = 1$ and $[\Sigma^{(\sigma=1)}] = 1$ and $[\Sigma^{(\sigma=\sigma')}] = 0$ for all other σ'. Thus, if the entanglement of a given distribution has value σ, the indicator variable is equal to one and zero otherwise. The resulting regression model is written as

$$\iota = c + \beta\mathcal{I} + \gamma\mathcal{E} + \sum_{\sigma\in\Sigma} \left(\hat{\alpha}^{(\sigma)}[\Sigma^{(\sigma)}] + \alpha^{(\sigma)}(\mathcal{I} \cdot [\Sigma^{(\sigma)}]) \right) \tag{3.4}$$

where as before lower-case Greek letters (except σ) denote the regression coefficients and the sum is taken over all different entanglement values used in the particular set of experiments. Taking the derivative of Equation 3.2 (page 60) with respect to

Table 3.1: Hyperparameters for the fully-connected and convolutional neural networks used in Section 3.3 (page 62) and Section 3.4 (page 73), respectively.

	Artificial datasets				Real-world datasets	
	Spiral	Step	Block	Sine	SVHN	CIFAR-10
Dataset size l	600	600	600	1200	-	-
Max. train set size	250	250	250	250	-	-
Hidden layer neurons	50-50	25-25	15-15	15-15	-	-
Activation	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU
Test accuracy threshold	0.8	0.9	0.95	0.8	0.9	0.75
Optimiser	Adam	Adam	Adam	Adam	Adam	Adam
Train epochs	75	75	75	75	20	20
Batch size	16	16	16	16	256	256

the intrinsic dimension $\mathcal{I}$ yields

$$\begin{aligned} \frac{\partial \iota}{\partial \mathcal{I}} &= \beta + \sum_{\sigma \in \Sigma} \alpha^{(\sigma)} [\Sigma^{(\sigma)}] \\ &= \beta + \alpha^{(\sigma)} \end{aligned} \tag{3.5}$$

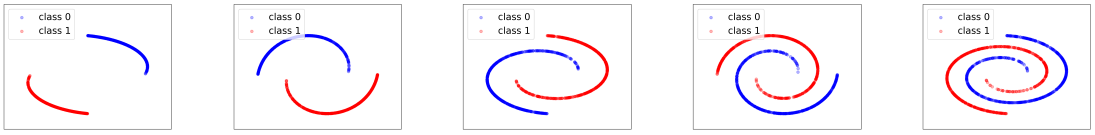
where the second step is due to Equation 3.3 (page 60) and fixing a level of entanglement σ, such that $[\Sigma^{(\sigma)}] = 1$. Thus, Equation 3.3 (page 60) estimates the effect of the intrinsic dimension on the sample complexity given a certain level of entanglement σ. Further,

$$\begin{aligned} \frac{\partial \iota}{\partial \Sigma} &= \sum_{\sigma \in \Sigma} \left(\hat{\alpha}^{(\sigma)} + \alpha^{(\sigma)} \mathcal{I} \right) \\ &= \hat{\alpha}^{(\sigma)} + \alpha^{(\sigma)} \mathcal{I} \end{aligned} \tag{3.6}$$

where again the second step is due to Equation 3.3 (page 60). Equation 3.6 (page 61) estimates the level of entanglement's influence on the sample complexity.

3.2.2 Classifier Architectures

The used hyperparameters are displayed in Table 3.1 (page 61). For the experiments with the artificial datasets in Section 3.3 (page 62) we use a single fully-connected network with two hidden layers, ReLU activations (see Section 2.1.2, page 16) and the Adam optimiser [236]. Between the datasets we vary the number of hidden layer neurons because the datasets vary significantly in their difficulty. For the real-world



(a) $\Sigma_{\text{Arch}} = 0.5$ (b) $\Sigma_{\text{Arch}} = 1.0$ (c) $\Sigma_{\text{Arch}} = 1.5$ (d) $\Sigma_{\text{Arch}} = 2.0$ (e) $\Sigma_{\text{Arch}} = 2.5$

Figure 3.1: Archimedean spiral datasets with different entanglement proxies Σ_{Arch}. For $\Sigma_{\text{Arch}} = 1$ the data is linearly separable.

datasets investigated in Section 3.4 (page 73) we use an architecture consisting of five convolutional layers, again with ReLU activations and the Adam optimiser.

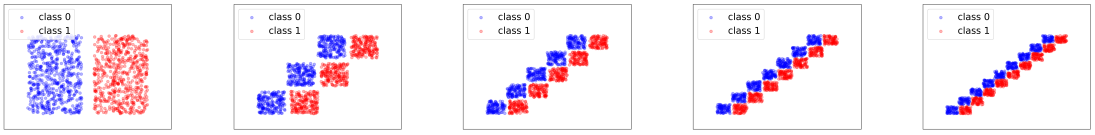
3.3 Artificial Datasets

We utilise four artificial datasets for which we can accurately control the different levels of entanglement. Given the mathematical formulation of these datasets, the entanglement parameter Σ does not correspond to the actual number of connected hyperplanes required for separation. Instead, the entanglement values Σ for each dataset are chosen such that the number of connected hyperplanes required for separation increases smoothly between subsequent values of Σ.

In Section 3.3.1 (page 62) we describe the used artificial datasets and in Section 3.3.2 (page 64) we describe how the intrinsic and extrinsic dimension of these datasets is changed precisely. Finally, in Section 3.3.3 (page 67) we discuss the results.

3.3.1 Overview

The artificial datasets described below are generated as base distributions $X \in \mathbb{R}^{l \times \mathcal{E}}_{[0,1]}$ with l samples in ($\mathcal{E} = 2$)-dimensional ambient space. Each distribution has a fixed entanglement value Σ. Additional intrinsic and extrinsic dimensions are added using the procedure described in Section 3.3.2 (page 64).



(a) $\Sigma_{\text{Step}} = 1$ (b) $\Sigma_{\text{Step}} = 3$ (c) $\Sigma_{\text{Step}} = 5$ (d) $\Sigma_{\text{Step}} = 7$ (e) $\Sigma_{\text{Step}} = 9$

Figure 3.2: Step-function datasets with different entanglement proxies Σ_{Step}. For $\Sigma_{\text{Step}} = 1$ the data is linearly separable.

Archimedean Spirals

A single ($\mathcal{I} = 1$)-dimensional Archimedean spiral embedded in ($\mathcal{E} = 2$)-dimensions can be written in Cartesian coordinates as

$$\text{Arch}(\Sigma_{\text{Arch}}) = (\Sigma_{\text{Arch}} \cos(\Sigma_{\text{Arch}}), \Sigma_{\text{Arch}} \sin(\Sigma_{\text{Arch}})) \tag{3.7}$$

where $\Sigma_{\text{Arch}} > 0$ is the length of the spiral. If two intertwined spirals are generated, Σ_{Arch} is the entanglement proxy for the resulting dataset. Spiral datasets with higher Σ_{Arch} require more connected linear segments to be separated. Figure 3.1 (page 62) displays spiral datasets for different Σ_{Arch}.

Step-function Dataset

The step-function dataset is directly described by the complexity of its optimal decision boundary. It is defined as

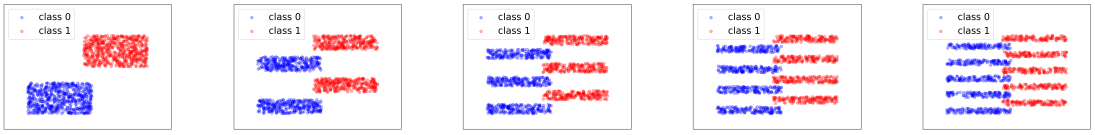
$$\text{Step}(x) = \lceil x \rceil, \; x \in [1, \Sigma_{\text{Step}}] \tag{3.8}$$

where $\lceil \cdot \rceil$ is the floor-function and $\Sigma_{\text{Step}} > 0$ is the maximum value of x. The larger Σ_{Step} the more connected linear segments, i.e. steps, the decision boundary consists of. Figure 3.2 (page 63) displays the dataset for different Σ_{Step}.

Block Dataset

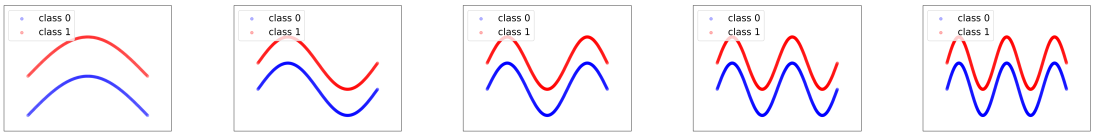
As in the case with the step-function dataset, the block dataset's decision boundary complexity is directly described as

$$\text{Block}(x) = \lceil x \rceil, \; x \in [1, \Sigma_{\text{Block}}] \tag{3.9}$$



(a) $\Sigma_{\text{Block}} = 1$ (b) $\Sigma_{\text{Block}} = 2$ (c) $\Sigma_{\text{Block}} = 3$ (d) $\Sigma_{\text{Block}} = 4$ (e) $\Sigma_{\text{Block}} = 5$

Figure 3.3: Block datasets with different entanglement proxies Σ_{Block}. For $\Sigma_{\text{Block}} = 1$ the data is linearly separable.



(a) $\Sigma_{\text{Sine}} = 1$ (b) $\Sigma_{\text{Sine}} = 2$ (c) $\Sigma_{\text{Sine}} = 3$ (d) $\Sigma_{\text{Sine}} = 4$ (e) $\Sigma_{\text{Block}} = 5$

Figure 3.4: Sine-function datasets with different entanglement proxies Σ_{Sine}. For $\Sigma_{\text{Sine}} = 1$ the data is linearly separable.

where $\lceil \cdot \rceil$ is the floor-function and $\Sigma_{\text{Block}} > 0$ is the maximum value of x. The larger Σ_{Block} the more blocks the dataset consists of and the more connected linear segments the decision boundary requires. The block dataset is similar to the $\hat{\text{L}}$MS-k dataset used by Shah et al. [120]. Figure 3.3 (page 64) displays the dataset for different Σ_{Block}.

Sine-function Dataset

The sine-function dataset consists of two sine-curves where one is shifted along the y-axis,

$$\text{Sine}(x) = \text{sine}(x), \; x \in [0, \Sigma_{\text{Sine}}] \tag{3.10}$$

The number of periods is the proxy for the entanglement Σ_{Sine} of the two sine-curves. Again, the higher Σ_{Sine}, the larger the entanglement. Figure 3.4 (page 64) displays the dataset for different Σ_{Sine}.

3.3.2 Changing the Intrinsic and Extrinsic Dimension

The artificial datasets $X \in \mathbb{R}^{l \times 2}$, where $l \in \mathbb{N}_+$ is the number of samples, lie within a ($\mathcal{E}_{\text{org}} = 2$)-dimensional ambient space, have either intrinsic dimension $\mathcal{I}_{\text{org}} = 1$ (spiral, sine-function datasets) or $\mathcal{I}_{\text{org}} = 2$ (step-function, bock datasets) and have a specific entanglement value Σ.

To increase the intrinsic and extrinsic dimension of these datasets by $\mathcal{I}_{\text{add}}$ or $\mathcal{E}_{\text{add}}$, respectively, the original data matrix X is column-augmented by matrices I and E, denoted as

$$X_{\text{aug}} \coloneqq [X|I|E] \in \mathbb{R}^{l\times(2+\mathcal{I}_{\text{add}}+\mathcal{E}_{\text{add}})} \tag{3.11}$$

Matrix $I \in \mathcal{U}_{[0,1]}^{l\times\mathcal{I}_{\text{add}}}$ contains elements drawn from a uniform distribution $\mathcal{U}$ over $[0,1]$ and adds further intrinsic dimensions $\mathcal{I}_{\text{add}}$ to dataset X. Matrix $E \in 0^{l\times\mathcal{E}_{\text{add}}}$ is a zero-matrix and adds further extrinsic dimensions $\mathcal{E}_{\text{add}}$ to dataset X. Then, the augmented matrix X_{aug} is matrix-multiplied with a random orthogonal matrix

$$O \in \mathbb{R}^{(2+\mathcal{I}_{\text{add}}+\mathcal{E}_{\text{add}})\times(2+\mathcal{I}_{\text{add}}+\mathcal{E}_{\text{add}})} \tag{3.12}$$

to remove the zero-columns in the augmented matrix introduced by concatenation with E. The result is the projected data matrix

$$X_{\text{proj}} = X_{\text{aug}}O \in \mathbb{R}^{l\times(2+\mathcal{I}_{\text{add}}+\mathcal{E}_{\text{add}})} \tag{3.13}$$

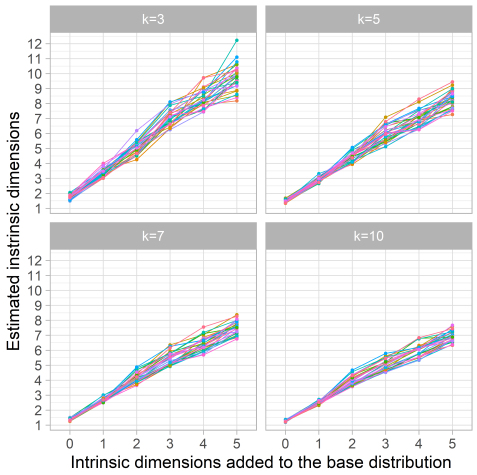
with intrinsic dimensionality $\mathcal{I} = \mathcal{I}_{\text{org}} + \mathcal{I}_{\text{add}}$ and extrinsic dimensionality $\mathcal{E} = \mathcal{E}_{\text{org}} + \mathcal{E}_{\text{add}}$. For all datasets we chose $\mathcal{I}_{\text{add}} \in [1,...,5]$ and $\mathcal{E}_{\text{add}} \in [1,...,5]$ as preliminary experiments did not show qualitatively different results for higher values.

In what follows we show that this procedure reliably increases the intrinsic dimension while having only a negligible effect on the entanglement.

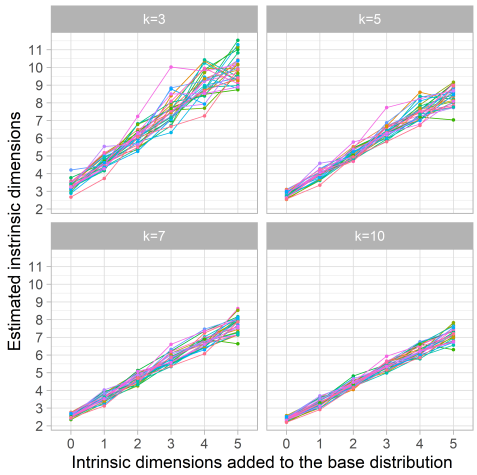
Measuring the Effect of the Added Intrinsic Dimensions on the Estimated Intrinsic Dimensions

We demonstrate that the procedure described above increases the intrinsic dimension of the datasets. As the estimate we use the one proposed by Levina et al. [237] and extended by MacKay et al. [238] since it had been found to yield realistic estimates of the intrinsic dimension for image datasets [16]. This estimate is denoted as

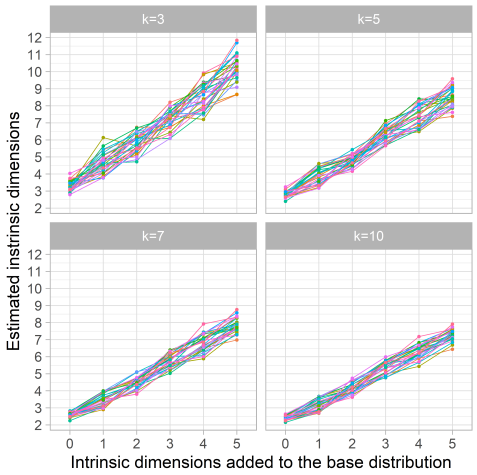
$$\mathcal{I}_k(X) \coloneqq \left[\frac{1}{l(k-1)}\sum_{i=1}^{l}\sum_{j=1}^{k-1}\log\frac{T_k(x)}{T_j(x)}\right]^{-1} \tag{3.14}$$



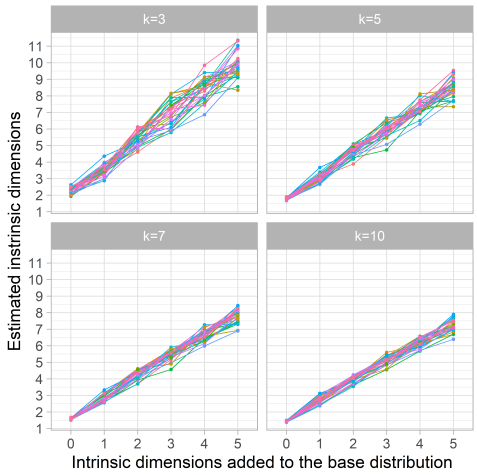
(a) Spiral dataset.



(b) Step-function dataset.



(c) Block dataset.

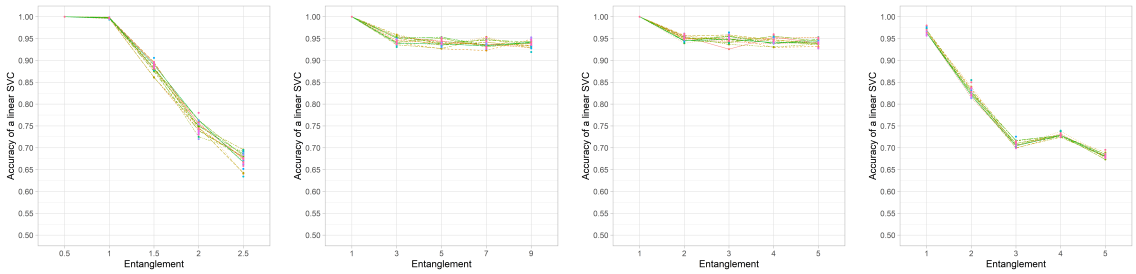


(d) Sine-function dataset.

Figure 3.5: Estimated intrinsic dimension of X_{proj} (Equation 3.13, page 65) for varying values of k (Equation 3.14, page 65). Individual colours denote combinations of extrinsic dimension and entanglement. Equation 3.14 (page 65) slightly overestimates the number of intrinsic dimensions added by the procedure described in Section 3.3.2 (page 64). Crucially, however, for all values of extrinsic dimension and entanglement there is clear positive correlation between added and estimated intrinsic dimension which is valid for all values of k.

where l is the number of samples in $X \in \mathbb{R}^{l \times \mathcal{E}}$, $\mathcal{E}$ the ambient dimension of those, k is the number of nearest neighbours to consider for each $x \in X$ and $T_j(x)$ is the Euclidean distance to the j-th nearest neighbor of x.

In Figure 3.5 we show that the addition of $\mathcal{I}_{\text{add}} > 0$ monotonically increases the estimated intrinsic dimension for all combinations of extrinsic dimension and entanglement. The fact that the estimated intrinsic dimension is higher than the added one, does not qualitatively affect the results.



(a) Spiral. (b) Step-function. (c) Block. (d) Sine-function.

Figure 3.6: Accuracy of a linear support vector classifier trained on X_{proj} (Equation 3.13, page 65). Individual colours denote combinations of intrinsic and extrinsic dimension. The addition of further intrinsic dimensions with the procedure described in Section 3.3.2 (page 64) does only have a negligible affect on the entanglement.

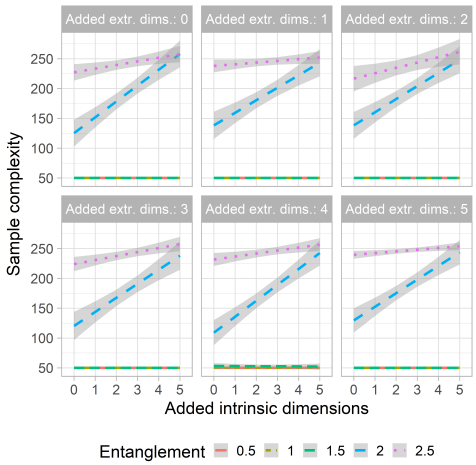
Effect of the Added Intrinsic Dimension on the Entanglement

In real-world image benchmarks variance that is introduced by the intrinsic dimensions can be a contributing factor to their entanglement. For example, datasets where objects are displayed in front of various backgrounds might spread the distribution across a large region in pixel space and reduce the Euclidean distance between objects of different classes that have visually similar backgrounds. Hence, the addition of intrinsic dimensions might inadvertently increase the entanglement of the distribution and delude our findings. To test whether this reasoning is also applicable to the artificial datasets, we use the accuracy of a *linear support vector classifier* [20] (SVC) as a measure of entanglement (see Section 3.4.2, page 74 for a discussion of the linear SVC as an entanglement measure).

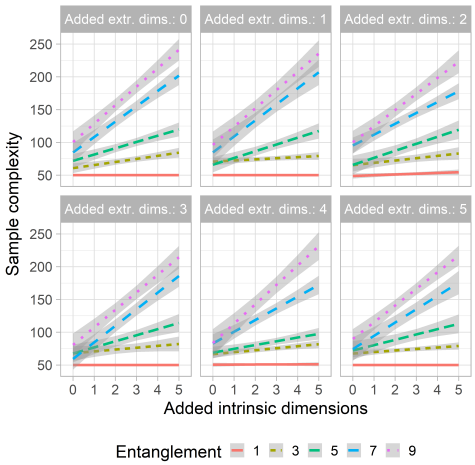
In Figure 3.6 (page 67) we display the accuracy of a linear SVC for different entanglement values over combinations of the intrinsic and extrinsic dimension. We observe that the lines are almost superimposed which means that the addition of intrinsic dimensions does not significantly affect the entanglement.

3.3.3 Results and Conclusions

We follow the experimental procedure outlined in Section 3.2 (page 59). We generate artificial datasets with different levels of entanglement and then add between one and five intrinsic and extrinsic dimensions to those. Then, we train fully-connected neural networks on subsets of different sizes and observe for which sample size the particular accuracy threshold is reached. With this approximated sample complexity

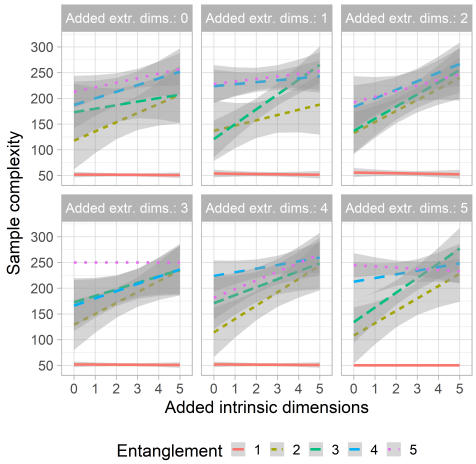


(a) Spiral dataset.

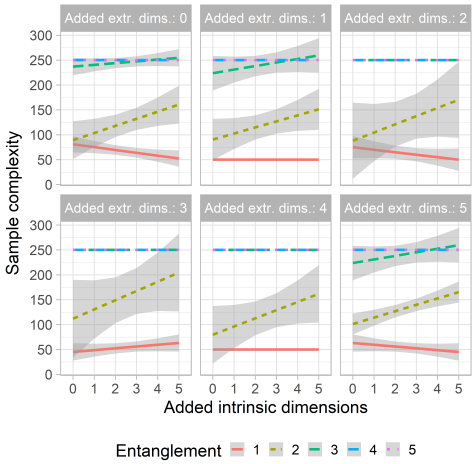


(b) Step-function dataset.

Figure 3.7: Sample complexity depending on the entanglement, intrinsic and extrinsic dimension for the (a) spiral dataset and (b) step-function dataset. Values are averaged over three runs. For the linearly separable datasets ($\Sigma_{\text{Arch}} = 0.5$, $\Sigma_{\text{Arch}} = 1.0$) increases of the intrinsic dimension do not influence the sample complexity. Datasets with higher levels of entanglement exhibit larger increases of the sample complexity when the intrinsic dimension is increased. The addition of further extrinsic dimension does not affect the sample complexity.



(a) Block dataset.



(b) Sine-function dataset.

Figure 3.8: Sample complexity depending on the entanglement, intrinsic and extrinsic dimension for the (a) block dataset and (b) sine-function dataset. Values are averaged over three runs. For the linearly separable block datasets ($\Sigma_{\text{Block}} = 0.5$) increases of the intrinsic dimension do not influence the sample complexity. For the linearly separable sine-function datasets ($\Sigma_{\text{Sine}} = 1.0$) this observation is roughly valid. For higher levels of entanglement, increases in the intrinsic dimension affect the sample complexity more. The extrinsic dimension does not qualitatively affect the sample complexity.

Table 3.2: Spiral dataset. The entanglement is a significantly more important factor for the sample complexity than the intrinsic dimension. Modelling the intrinsic dimension's influence on the sample complexity given a certain level of entanglement, shows that with increasing entanglement the effect of intrinsic dimensionality increases rises. For low levels of entanglement the intrinsic dimension does not affect the sample complexity. The sample complexity is independent of the extrinsic dimension.

	Sample complexity, ι		
	Equation 3.1	Equation 3.2	Equation 3.4
Σ_{Arch}	104.472*** (2.065)	87.782*** (4.667)	
$\mathcal{I}$	5.838*** (0.855)	−4.375* (2.323)	0.000 (0.762)
$\mathcal{E}$	−0.324 (0.855)	0.234 (2.323)	−0.324 (0.341)
$\Sigma_{\text{Arch}} * \mathcal{I}$		6.929*** (1.188)	
$\Sigma_{\text{Arch}} * \mathcal{E}$		−0.252 (1.188)	
$\mathcal{I} * \mathcal{E}$		−0.072 (0.492)	
$[\Sigma_{\text{Arch}}^{(1.0)}]$			−0.000 (3.261)
$[\Sigma_{\text{Arch}}^{(1.5)}]$			0.476 (3.261)
$[\Sigma_{\text{Arch}}^{(2.0)}]$			76.786*** (3.261)
$[\Sigma_{\text{Arch}}^{(2.5)}]$			179.484*** (3.261)
$\mathcal{I} * [\Sigma_{\text{Arch}}^{(1.0)}]$			0.000 (1.077)
$\mathcal{I} * [\Sigma_{\text{Arch}}^{(1.5)}]$			−0.024 (1.077)
$\mathcal{I} * [\Sigma_{\text{Arch}}^{(2.0)}]$			23.786*** (1.077)
$\mathcal{I} * [\Sigma_{\text{Arch}}^{(2.5)}]$			5.429*** (1.077)
Constant	−54.550*** (4.567)	−29.963*** (8.327)	50.810*** (2.458)
Observations	900	900	900
R^2	0.744	0.754	0.960
Adjusted R^2	0.743	0.752	0.959
F Statistic	869.089***	455.279***	2,117.691***
Note:			*p<0.1; **p<0.05; ***p<0.01

the three introduced regression models in Section 3.2 (page 59) are estimated. In this section we discuss the findings and in Figure 3.7 (page 68) and Figure 3.8 (page 68) the results are presented visually.

Spiral Dataset

In Table 3.2 (page 69) we display the estimated regression models for the spiral dataset and in Figure 3.7a (page 68) we visualise these results. We find that the intrinsic dimension and the entanglement both positively influence the sample complexity, however the entanglement's influence is far greater in all three estimated regression models. Whereas the first (Equation 3.1, page 59) and second (Equation 3.2, page 60) regression model, fit the data reasonably well, the regression modelling the intrinsic dimensionality's influence on the sample complexity given a level of entanglement (Equation 3.4, page 60), fits the data nearly perfect with

Table 3.3: Step-function dataset. The entanglement is the most influential factor on the sample complexity. The intrinsic dimensionality's influence on the sample complexity depends strongly on the level of entanglement. For low levels of entanglement for which the data is (almost) linearly separable the intrinsic dimension does not affect the sample complexity while for higher levels of entanglement increases in intrinsic dimensionality affect the sample complexity to a greater extend. In contrast to the other artificial datasets we observe that with increasing extrinsic dimensionality the sample complexity decreases.

	Sample complexity, ι		
	Equation 3.1	Equation 3.2	Equation 3.4
Σ_{Step}	13.875*** (0.316)	6.375*** (0.548)	
$\mathcal{I}$	12.110*** (0.524)	−4.950*** (0.988)	0.214 (0.846)
$\mathcal{E}$	−1.871*** (0.524)	2.045** (0.988)	−1.871*** (0.378)
$\Sigma_{\text{Step}} * \mathcal{I}$		3.598*** (0.140)	
$\Sigma_{\text{Step}} * \mathcal{E}$		−0.598*** (0.140)	
$\mathcal{I} * \mathcal{E}$		−0.371 (0.231)	
$[\Sigma^{(3)}_{\text{Step}}]$			16.587*** (3.620)
$[\Sigma^{(5)}_{\text{Step}}]$			18.849*** (3.620)
$[\Sigma^{(7)}_{\text{Step}}]$			30.317*** (3.620)
$[\Sigma^{(9)}_{\text{Step}}]$			41.944*** (3.620)
$\mathcal{I} * [\Sigma^{(3)}_{\text{Step}}]$			2.810** (1.196)
$\mathcal{I} * [\Sigma^{(5)}_{\text{Step}}]$			8.738*** (1.196)
$\mathcal{I} * [\Sigma^{(7)}_{\text{Step}}]$			21.095*** (1.196)
$\mathcal{I} * [\Sigma^{(9)}_{\text{Step}}]$			26.833*** (1.196)
Constant	6.724*** (2.594)	41.903*** (3.462)	54.560*** (2.729)
Observations	900	900	900
R^2	0.734	0.850	0.862
Adjusted R^2	0.733	0.849	0.861
F Statistic	823.949***	840.355***	557.368***
Note:			*p<0.1; **p<0.05; ***p<0.01

an adjusted $R^2 = 0.96$. For low levels of entanglement, where the data is (almost) linearly separable, an increase of the intrinsic dimension does not affect the sample complexity at all. Only when the entanglement is high, increases in intrinsic dimensionality affect the sample complexity positively and higher levels of entanglement are associated with greater sensitivity to increases in the intrinsic dimension. The extrinsic dimension does not influence the sample complexity.

Step-function Dataset

In Table 3.3 (page 70) we display the estimated regression models for the step-function dataset and in Figure 3.7b (page 68) we visualise these results. The findings are consistent with those for the spiral dataset. The entanglement is a more important factor for the sample complexity than the intrinsic dimension, though

Table 3.4: Block dataset. The entanglement is the most influential factor on the sample complexity. As for higher distributions with higher entanglement the sample complexity already near the maximum number of available train samples, the intrinsic dimension's influence on the sample complexity does not increase with increasing entanglement.

	Sample complexity, ι		
	Equation 3.1	Equation 3.2	Equation 3.4
Σ_{Block}	41.713*** (1.657)	40.642*** (3.819)	
$\mathcal{I}$	11.190*** (1.372)	8.806** (3.802)	−0.317 (2.470)
$\mathcal{E}$	1.444 (1.372)	0.108 (3.802)	1.444 (1.105)
$\Sigma_{\text{Block}} * \mathcal{I}$		0.389 (0.973)	
$\Sigma_{\text{Block}} * \mathcal{E}$		0.040 (0.973)	
$\mathcal{I} * \mathcal{E}$		0.487 (0.805)	
$[\Sigma^{(2)}_{\text{Block}}]$			70.437*** (10.577)
$[\Sigma^{(3)}_{\text{Block}}]$			98.810*** (10.577)
$[\Sigma^{(4)}_{\text{Block}}]$			146.892*** (10.577)
$[\Sigma^{(5)}_{\text{Block}}]$			165.476*** (10.577)
$\mathcal{I} * [\Sigma^{(2)}_{\text{Block}}]$			20.437*** (3.494)
$\mathcal{I} * [\Sigma^{(3)}_{\text{Block}}]$			19.643*** (3.494)
$\mathcal{I} * [\Sigma^{(4)}_{\text{Block}}]$			10.595*** (3.494)
$\mathcal{I} * [\Sigma^{(5)}_{\text{Block}}]$			6.865** (3.494)
Constant	20.218*** (7.331)	26.477* (13.628)	49.034*** (7.973)
Observations	540	540	540
R^2	0.567	0.567	0.723
Adjusted R^2	0.564	0.562	0.718
F Statistic	233.763***	116.430***	137.982***
Note:			*p<0.1; **p<0.05; ***p<0.01

the difference is not as significant as for the spiral dataset. Again we observe that the regression model that includes the entanglement as categorical variables (Equation 3.4, page 60) fits the data best. The influence of the intrinsic dimension depends strongly on the given level of entanglement, where higher entanglement levels result in a greater influence of the intrinsic dimension on the sample complexity. Interestingly, we observe that increases in the extrinsic dimension slightly, but statistically significant, reduce the sample complexity. As this observation is not true for all other datasets, we attribute it to particularities of the step-function dataset.

Block Dataset

In Table 3.4 (page 71) we display the estimated regression models for the block dataset and in Figure 3.8a (page 68) we visualise these results. We find again that the entanglement is the leading contributor to the sample complexity, surpassing the influence of the intrinsic dimension greatly. Further, the influence of the intrinsic

Table 3.5: Sine-function dataset. The entanglement is the most influential factor on the sample complexity. As for higher distributions with higher entanglement the sample complexity already near the maximum number of available train samples, the intrinsic dimension's influence on the sample complexity does not increase with increasing entanglement.

	Sample complexity, ι		
	Equation 3.1	Equation 3.2	Equation 3.4
Σ_{Sine}	50.625*** (2.068)	53.006*** (4.783)	
$\mathcal{I}$	3.262* (1.713)	6.389 (4.763)	−1.786 (1.371)
$\mathcal{E}$	0.024 (1.713)	−0.920 (4.763)	0.024 (0.613)
$\Sigma_{\text{Sine}} * \mathcal{I}$		−1.155 (1.218)	
$\Sigma_{\text{Sine}} * \mathcal{E}$		0.202 (1.218)	
$\mathcal{I} * \mathcal{E}$		0.135 (1.009)	
$[\Sigma^{(2)}_{\text{Sine}}]$			32.738*** (5.871)
$[\Sigma^{(3)}_{\text{Sine}}]$			178.373*** (5.871)
$[\Sigma^{(4)}_{\text{Sine}}]$			189.286*** (5.871)
$[\Sigma^{(4)}_{\text{Sine}}]$			189.286*** (5.871)
$\mathcal{I} * [\Sigma^{(2)}_{\text{Sine}}]$			16.905*** (1.939)
$\mathcal{I} * [\Sigma^{(3)}_{\text{Sine}}]$			4.762** (1.939)
$\mathcal{I} * [\Sigma^{(4)}_{\text{Sine}}]$			1.786 (1.939)
$\mathcal{I} * [\Sigma^{(5)}_{\text{Sine}}]$			1.786 (1.939)
Constant	26.716*** (9.150)	20.415 (17.071)	60.655*** (4.425)
Observations	180	180	180
R^2	0.774	0.775	0.972
Adjusted R^2	0.770	0.767	0.971
F Statistic	200.899***	99.433***	590.620***
Note:			*p<0.1; **p<0.05; ***p<0.01

dimension on the sample complexity depends on the level of entanglement. However, whereas we previously found an increasing influence of the intrinsic dimension on the sample complexity for higher levels of entanglement, we here observe the opposite. The reason for this observation is that for larger levels of entanglement the sample complexity is already close to the maximum value of 250, and so increases in intrinsic dimensionality only have a small impact.

Sine-function Dataset

In Table 3.5 (page 72) we display the estimated regression models for the sine-function dataset and in Figure 3.8b (page 68) we visualise these results. The results are analogous to those for the block dataset. The entanglement is the most important factor for the sample complexity and due to the already high sample complexity of highly entangled distributions, the influence of the intrinsic dimension decreases for high levels of entanglement. The extrinsic dimension does not influence the

sample complexity.

Summary

We observe consistent results for all four artificial datasets. First, the entanglement is the most important factor for the sample complexity, for some datasets it significantly surpasses the importance of the intrinsic dimension. Secondly, the intrinsic dimension's influence on the sample complexity depends on the given level of entanglement. For datasets that are (almost) linearly separable, increases in intrinsic dimension do not affect the sample complexity at all. For higher levels of entanglement the influence of the intrinsic dimension on the sample complexity progressively increases, except for those datasets where the sample complexity is already close to the maximum number of available train samples. Thus, our results show that for these datasets studying the intrinsic dimension's influence on the sample complexity cannot be done in isolation and needs to consider the given level of entanglement of the base distribution.

3.4 Real Datasets

In the previous section we considered several different artificial datasets for which we can control the level of entanglement precisely. We established relationships between the geometric properties of these distributions and the sample complexity of fully-connected neural networks by estimating the regression models presented in Section 3.2.1 (page 59). In this section we expand our analysis to common image benchmarks and convolutional neural networks.

In Section 3.4.2 (page 74) we introduce methods to measure the entanglement of a given distribution where the data generating function is unknown and describe a simple, but effective, way of changing the entanglement. In Section 3.4.3 (page 77) we show how the intrinsic dimension is changed and in Section 3.4.4 (page 79) we discuss the results.

3.4.1 Overview

As we do not have access to the data-generating function for real-world image benchmarks, we cannot control the intrinsic dimension and entanglement directly and precisely. As the extrinsic dimension does not influence the sample complexity, neither theoretically [12, 13] nor empirically [16] and our results again confirm this (with the exception of the step-function dataset), we omit changing the extrinsic dimension for the real-world datasets, and remove $\mathcal{E}$ from the regression models in Equation 3.1 (page 59), Equation 3.2 (page 60) and Equation 3.4 (page 60). We simplify the analysis by considering binary classification problems, where we choose class pairs from the original distribution according to their semantic similarity.

3.4.2 Entanglement of Real Distributions

Entanglement Measures

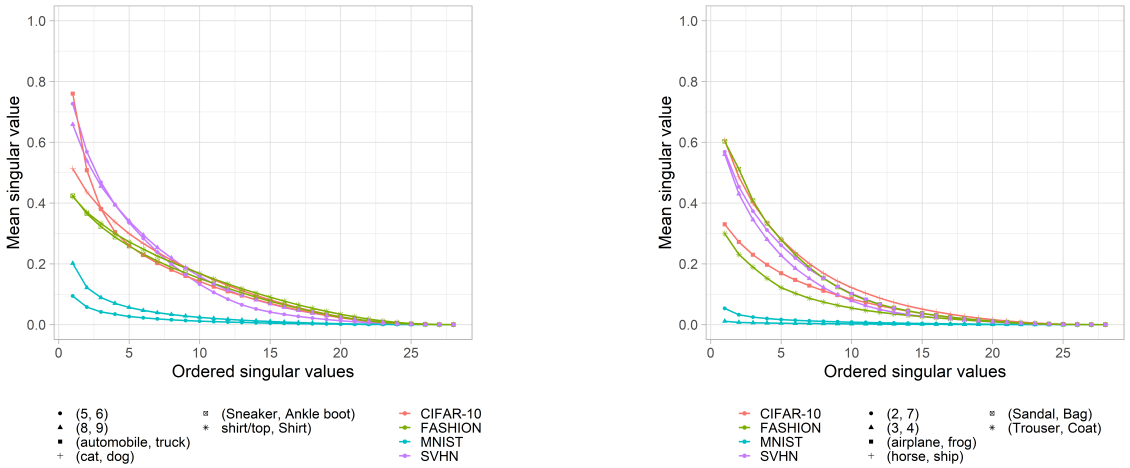
Measuring the entanglement for real-image benchmarks is complicated as it requires knowledge of the data generating function or access to the Bayes classifier. As neither of them are available, we introduce two methods to measure the entanglement in the case of binary classification.

I. Linear Support Vector Classifier A linear support vector (SVC) separates two classes by a single hyperplane such that the margin between the classes is maximised. The more entangled the classes are, the lower the linear support vector's accuracy will be. Thus, the accuracy of a linear support vector can be used as an approximation of the entanglement of the entire distribution.

II. Spectrum of The Decision Function's Hessian The second measure of entanglement utilises a trained binary classifier $f : \mathbb{R}^{\mathcal{E}} \to \mathbb{R}^2_{[0,1]}$ that maps inputs to a vector of class scores. The decision function is denoted as

$$f_{\text{dec}}(x) \coloneqq f^{(1)}(x) - f^{(0)}(x) \tag{3.15}$$

where $f^{(i)}(x)$ is the class score of the i-th class. For all $\bar{x}$ for which $f_{\text{dec}}(\bar{x}) = 0$, the decision function describes the decision boundary of f. Assuming a square



(a) Similar classes.

(b) Dissimilar classes.

Figure 3.9: Averaged spectrum of the decision function's Hessian $\mathcal{H}_{f_d}(\cdot)$ (see Equation 3.16, page 75). The difficulty of a dataset is negatively correlated with its spectrum for (a) semantically or visually similar classes and (b) semantically or visually dissimilar classes.

approximation of the decision function, and thus omitting higher order-terms, the Taylor approximation of f_{dec} around $\bar{x}$ yields

$$T_{f_{\text{dec}}}(x) = f_{\text{dec}}(\bar{x}) + (x - \bar{x})^T \mathcal{J}_{f_{\text{dec}}}(\bar{x}) + \frac{1}{2!}(x - \bar{x})^T \mathcal{H}_{f_{\text{dec}}}(\bar{x})(x - \bar{x}) \tag{3.16}$$

where $\mathcal{J}_{f_{\text{dec}}}(\bar{x})$ is the Jacobian and $\mathcal{H}_{f_{\text{dec}}}(\bar{x})$ is the Hessian of f_{dec} evaluated at $\bar{x}$. The spectrum of the decision function's Hessian $\mathcal{H}_{f_{\text{dec}}}(\bar{x})$ evaluated at $\bar{x}$ for which $f_{\text{dec}}(\bar{x}) = 0$ describes the curvature of f_{dec} locally around $\bar{x}$ and describes how much the decision boundary's shape differs from a linear one.

We compute this measure for two semantically or visually similar and dissimilar classes from the MNIST, FASHION, SVHN and CIFAR-10 datasets. First, we draw one sample from each class, denoted as $x^{(0)}$ and $x^{(1)}$. Then, we solve

$$\bar{x} = wx^{(0)} + (1 - w)x^{(1)} \tag{3.17}$$

for $w \in [0, 1]$ such that $f_{\text{dec}}(\bar{x}) = 0$. We repeat this procedure for 500 random pairs of points and display the mean singular values in Figure 3.9 (page 75). The results are in line with common knowledge. MNIST, for example, is almost solvable with a simple linear classifier and thus it exhibits the flattest spectrum. SVHN and CIFAR-10 on the other hand, are not linearly separable and their spectrum is larger.

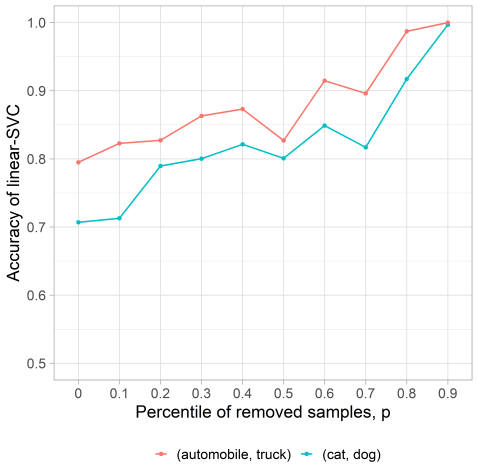


Figure 3.10: Accuracy of a linear support vector classifier trained on differently entangled subset of the CIFAR-10 dataset. Removing samples with higher gradient norms g_i (Equation 3.18, page 76) reduces the entanglement of the dataset.

Since the measure using the Hessian's spectrum delivers the same results as a SVC would, we use the SVC from here on due to its significantly cheaper computation.

Changing the Entanglement

As we do not have access to the data generating function of the real datasets, we cannot change their entanglement by drawing new samples from it. For CIFAR-10 attempts to recreate the original data collection process have been made, however, those were found to be error prone and can introduce large shifts between original and new distribution [46]. Thus, we can only reduce the entanglement of a given distribution.

Intuitively, we can reduce the entanglement of a distribution by removing samples close to the decision boundary of a trained classifier f as their position defines its location. As an approximation for the closeness of a sample x to the decision boundary we compute the Frobenius-norm of the model's gradient evaluated at x_i,

$$g_i \coloneqq \left|\left|\nabla_x f(x_i)\right|\right|_{\text{Frob}} \tag{3.18}$$

As it is commonly known that modern neural networks are susceptible to small norm perturbations (see Section 2.2.1, page 19), i.e. they lie closer to the decision

boundary than the data distribution would allow, we utilise a pre-trained robust model f. We choose Ding et al.'s [183] model which is explicitly trained to maximise the margin around train samples and therefore has a decision boundary that lies farther in the low-density region between classes.

To decrease the entanglement, we compute g_i for all $x_i \in X$ and remove the p-th percentile of samples from X that exhibit the highest gradient norms. Thus, the higher p, the more samples are removed and the lower the entanglement Σ_{Real} becomes. Then,

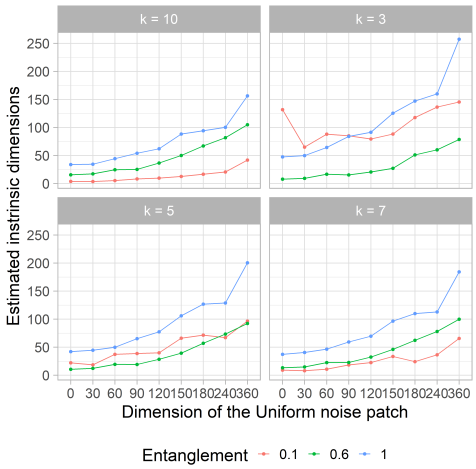
$$\Sigma_{\text{Real}} \coloneqq 1 - p \tag{3.19}$$

with $p \in [0.0, 0.1, ..., 0.9]$ and $\Sigma_{\text{Real}} \in [0.1, 1.0]$. To keep the original dataset size constant, we replace those removed samples with observations with lower gradient norms and perturb them with Gaussian-noise. As neural networks are not naturally robust to this sort of noise, augmenting the smaller dataset with these samples works as a sort of data augmentation that the network is supposed to learn. Thus, the measured sample complexity are still interpretable between differently entangled distributions.

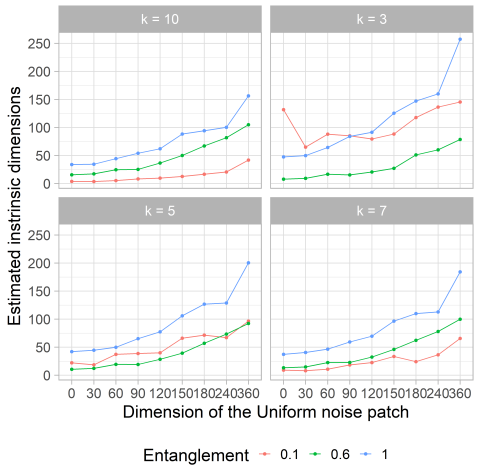
In Figure 3.10 (page 76) we plot the accuracy of a linear SVC on datasets with different entanglement values and observe that the procedure indeed reduces the entanglement, though not strictly monotonous due to statistical effects. As a result, we choose for the experiment three different entanglement values $p \in \{0, 0.5, 0.9\}$ and so $\Sigma_{\text{Real}} \in \{1.0, 0.5, 0.1\}$.

3.4.3 Changing the Intrinsic Dimension

To increase the intrinsic dimension of the image benchmarks we replace a random patch in each image with Uniform-noise of size $\mathcal{I}_{\text{add}}$. A similar procedure to increase the intrinsic dimension of image datasets was used by Pope et al. [16]. We set $\mathcal{I}_{\text{add}} \in [0, 5, 10, 15, 30, 60, 90, 120, 150]$ where $\mathcal{I}_{\text{add}} = 0$ is the dataset with the original intrinsic dimensionality. Pope et al. [16] report an original intrinsic dimensionality for CIFAR-10 between 13 and 25, depending on the number of nearest neighbours k (Equation 3.14, page 65). Thus, the ratios between intrinsic and extrinsic dimensionality for the artificial and the real-world datasets are comparable



(a) CIFAR-10 (*cat*, *dog*).



(b) CIFAR-10 (*automobile*, *truck*)

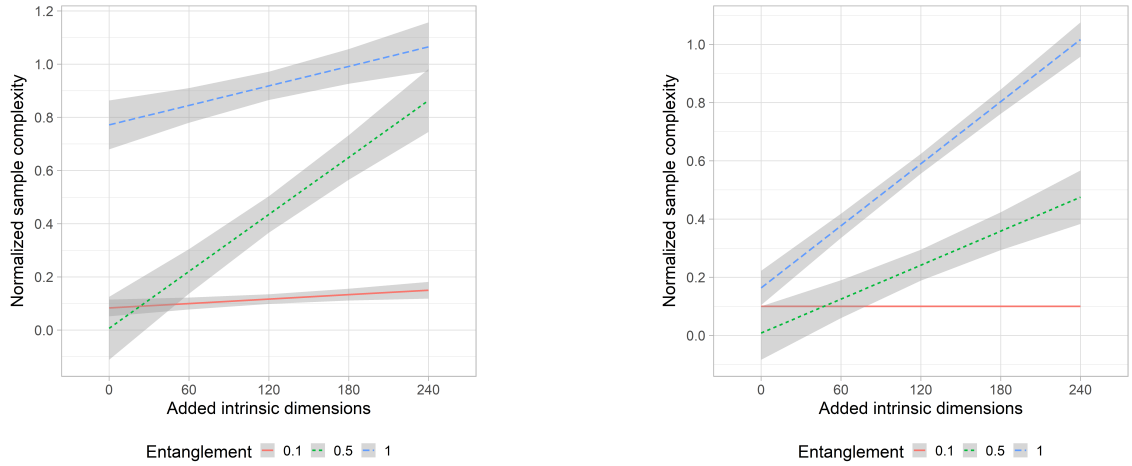
Figure 3.11: Estimated intrinsic dimension of the disentangled real-world datasets. Adding patches of Uniform-noise increases the estimated intrinsic dimension. Distributions with higher entanglement also display higher intrinsic dimension, despite the addition of Gaussian-noise perturbed samples. Thus, the entanglement's effect on the sample complexity is systematically overstated.

in our work.

Discussion

As we do not have access to the data-generating function for the used datasets we require heuristics to change their geometric properties. In Section 3.4.2 (page 76) we show that removing samples with the largest gradient norms reduces the entanglement. Here, we demonstrate that the procedure for increasing the intrinsic dimension works as intended. We compute the estimate of the intrinsic dimensionality using Equation 3.14 (page 65) again.

In Figure 3.11 (page 78) we display the relationship between the estimated intrinsic dimension and the added intrinsic dimension for different values of entanglement. We observe that adding Uniform-noise patches of size $\mathcal{I}_{\text{add}}$ increase the estimated intrinsic dimension, although, the corresponding increase is not one-to-one. The approximated intrinsic dimension $\tilde{\mathcal{I}}$ underestimates the added intrinsic dimension $\mathcal{I}_{\text{add}}$. Further, we observe in Figure 3.11 (page 78) that a reduction in entanglement results in a reduction of the intrinsic dimension when comparing the datasets with the lowest ($\Sigma_{\text{Real}} = 0.1$) and highest levels ($\Sigma_{\text{Real}} = 1.0$) of entanglement despite the addition of Gaussian-noise perturbed samples. This means that we cannot



(a) CIFAR-10 (*cat*, *dog*)

(b) CIFAR-10 (*automobile*, *truck*)

Figure 3.12: Sample complexity depending on the entanglement, intrinsic and extrinsic dimension for the (a) CIFAR-10 (*cat*, *dog*) and (b) CIFAR-10 (*automobile*, *truck*).

precisely tell apart the effects of entanglement and intrinsic dimension on the sample complexity. Due to the co-occurrence of high entanglement and high intrinsic dimension, the effect of the entanglement on the sample complexity will be systematically over-stated. Nevertheless, the experiments in Section 3.4.4 (page 79) show that the regression that best describes the relationship between intrinsic dimension and entanglement models the intrinsic dimension's influence with respect to the level of entanglement. Thus, the co-occurrence of high-entanglement and high intrinsic dimension does not influence the results in a qualitative way.

3.4.4 Results and Conclusions

We choose two binary classification problems from the CIFAR-10 dataset. First, the classes *cat* and *dog* that are semantically similar and, secondly the classes *automobile* and *truck* that are semantically dissimilar. We measure semantic similarity by their shape similarity as the humans visual system is known to rely heavily on shapes for object recognition [8, 68]. Choosing an animal class and a vehicle class as the semantically dissimilar classes would result in a distribution with low original entanglement and thus would be unsuitable for the experiments as they require significantly different levels of entanglement. We follow the same approach as in Section 3.3 (page 62) and estimate the regression models from Equation 3.1 (page 59),

Table 3.6: CIFAR-10 (*cat*, *dog*). The entanglement is significantly more important for the sample complexity than the intrinsic dimension. The intrinsic dimension's influence on the sample complexity depends on the given level of entanglement.

	Normalized sample complexity, ι/l		
	Equation 3.1	Equation 3.2	Equation 3.4
Σ_{Real}	0.894^{***} (0.061)	0.796^{***} (0.106)	
$\mathcal{I}_{\text{add}}$	0.002^{***} (0.0003)	0.001^{**} (0.0005)	0.0003 (0.0003)
$\mathcal{I}_{\text{add}} \cdot \Sigma_{\text{Real}}$		0.001 (0.001)	
$[\Sigma_{\text{Real}}^{(0.5)}]$			−0.077 (0.058)
$[\Sigma_{\text{Real}}^{(1.0)}]$			0.688^{***} (0.058)
$\mathcal{I}_{\text{add}} \cdot [\Sigma_{\text{Real}}^{(0.5)}]$			0.003^{***} (0.0004)
$\mathcal{I}_{\text{add}} \cdot [\Sigma_{\text{Real}}^{(1.0)}]$			0.001^{**} (0.0004)
Constant	-0.189^{***} (0.051)	-0.137^{*} (0.069)	0.083^{**} (0.041)
Observations	45	45	45
R^2	0.858	0.862	0.952
Adjusted R^2	0.851	0.852	0.945
F Statistic	126.641^{***}	85.415^{***}	153.593^{***}
Note:			*p<0.1; **p<0.05; ***p<0.01

Equation 3.2 (page 60) and Equation 3.4 (page 60). However, to maintain consistency between different binary datasets we compute the normalized sample complexity, ι/l, that divides the sample complexity by the number of available samples l.

In Table 3.6 (page 80) and Table 3.7 (page 81) we display the estimated regression models and in Figure 3.12 (page 79) we visualise these results. For both binary classification problems we find that the entanglement is, as in the case for the artificial datasets, the significantly more important factor for the sample complexity. We can also confirm for the real-world datasets that the intrinsic dimension's influence depends on the entanglement. For a low level of entanglement increases in the intrinsic dimension do not affect the sample complexity.

3.5 Summary and Discussion

In this chapter we investigate the influence of three geometric properties on the sample complexity of neural networks in binary classification tasks. These properties are the intrinsic and extrinsic dimension and the entanglement. Previous works only considered the intrinsic and extrinsic dimension [16] but we also include the entanglement. We show for artificial and real-world datasets and for fully-connected and convolutional neural networks, that the entanglement is the leading contributor

Table 3.7: CIFAR-10 (*automobile*, *truck*). The entanglement is significantly more important for thee sample complexity than the intrinsic dimension. The intrinsic dimension's influence on the sample complexity depends on the given level of entanglement.

	Normalized sample complexity, ι/l		
	Equation 3.1	Equation 3.2	Equation 3.4
Σ_{Real}	0.551^{***} (0.060)	0.080 (0.053)	
$\mathcal{I}_{\text{add}}$	0.002^{***} (0.0003)	-0.0003 (0.0002)	-0.000 (0.0002)
$\mathcal{I}_{\text{add}} \cdot \Sigma_{\text{Real}}$		0.004^{***} (0.0004)	
$[\Sigma_{\text{Real}}^{(0.5)}]$			-0.092^{**} (0.041)
$[\Sigma_{\text{Real}}^{(1.0)}]$			0.063 (0.041)
$\mathcal{I}_{\text{add}} \cdot [\Sigma_{\text{Real}}^{(0.5)}]$			0.002^{***} (0.0003)
$\mathcal{I}_{\text{add}} \cdot [\Sigma_{\text{Real}}^{(1.0)}]$			0.004^{***} (0.0003)
Constant	-0.203^{***} (0.049)	0.048 (0.034)	0.100^{***} (0.029)
Observations	45	45	45
R^2	0.764	0.939	0.957
Adjusted R^2	0.753	0.935	0.951
F Statistic	67.967^{***}	212.099^{***}	173.320^{***}
Note:			*p<0.1; **p<0.05; ***p<0.01

to the sample complexity and that it is significantly more important than the intrinsic dimension. Crucially, the intrinsic dimensionality's influence on the sample complexity depends on the given level of entanglement. This finding is important as previous work [16] only considers complex image benchmarks such as ImageNet and CIFAR-10, which are high-entanglement distributions. Thus, we complement previous works and show that the influence of the intrinsic dimension on the sample complexity cannot be studied in isolation.

Practical implications The observation that for fully-connected and convolutional neural networks increases in intrinsic dimension do not influence the sample complexity when classes are well separated is relevant for practical purposes. The intrinsic dimensions denote the factors of variation of a distribution, so those variations that do not change the label of an image. If a neural network is trained on a well-separated distribution, i.e. one with low entanglement, it behaves similar to a support vector classifier whose sample complexity is also not influenced by the intrinsic dimension. Thus, well-separated distributions could be augmented by many label-preserving transformations without increasing the sample complexity and requiring an architecture with greater capacity.

Chapter 4

Complexity of Robust Decision Boundaries

4.1 Introduction

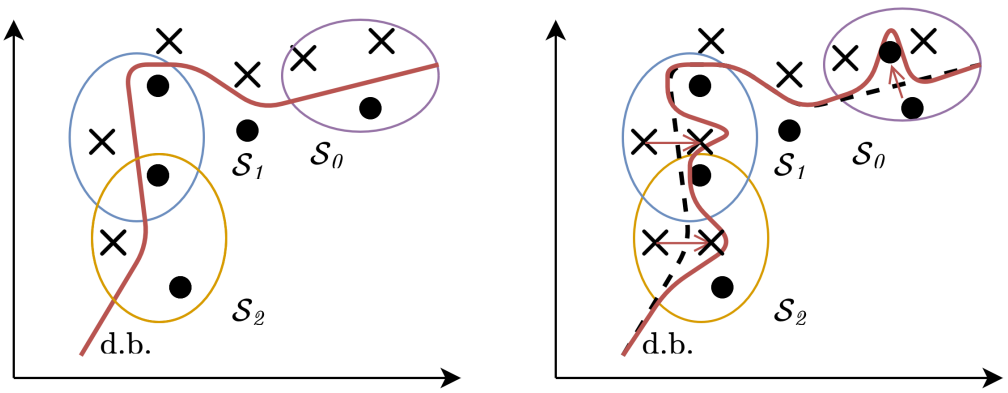
Over the past years deep neural networks have matched or even surpassed human-level performance in object classification tasks [9, 10]. However, this remarkable achievement was partially over-shadowed by the observation that otherwise well-generalising neural networks display a surprising lack of robustness compared to humans, when confronted with a wide variety of distribution shifts between train and test data. These distribution shifts can be the result of small-norm, for humans imperceptible, synthetic perturbations (see Section 2.2.1, page 19) such as adversarial examples (Definition 7, page 20) and small- and large-norm natural perturbations (see Section 2.2.2, page 23), such as changes in lighting [41], background [43, 78] or introduced by differences in the dataset creation process [46]. These observations resulted in a large body of literature on robust training methods (Definition 11, page 35) that aims to increase the robustness of neural network classifiers (see Section 2.4, page 35) to the aforementioned perturbation types and helped to close the robustness gap between humans and neural networks against natural perturbations [80].

Nevertheless, it remains a common observation across training methods that increased robustness results in reduced test accuracy, a phenomenon that is commonly referred to as the accuracy-robustness tradeoff (see Section 2.2.3, page 26). Several

authors hypothesised that one reason for this tradeoff might be that robust training requires different, and possibly geometrically more complex, decision boundaries compared to standard training [55, 118, 120, 126, 131, 239] that only aims to increase accuracy. If this hypothesis is valid, the need for greater capacity [118] and increased sample complexity of robust training could partially be explained [126].

The decision boundary is an important property to study. Its geometric complexity, i.e. the number of its connected hyperplanes, is an indicator for the complexity of the data distribution and the learning difficulty [12, 13], the margin between the decision boundary and the train samples determines the classifier's robustness [183, 208, 239] and the decision boundary's position in input space can be used for explaining model predictions [240]. However, studying the decision boundary of neural networks is challenging. They are highly non-linear, high dimensional and are build on-top of a largely opaque feature representation. Further, they could theoretically consist of several disconnected decision regions [209].

In this chapter we investigate the hypothesis that robust training requires geometrically more complex decision boundaries in the original pixel space than accurate training. To circumvent the aforementioned problems, we study decision boundaries in a model-agnostic, but dataset-specific, way. We assume the existence of an *accurate decision boundary* of unknown geometric complexity that is obtained by standard training which solely minimises the training loss. We compare this accurate decision boundary to a *robust decision boundary* that would be required if the data was altered by worst-case perturbations of its samples. These worst-case perturbations are derived by dividing the input distribution into linearly separable sets of nearest neighbours and investigating the perturbation magnitudes required to make them non-linearly separable. As such, the magnitude of these worst-case perturbations are a lower-bound on the perturbation magnitudes over which a geometrically more complex decision boundary is provably required for the considered dataset. We show that state-of-the-art robust training methods indeed learn geometrically more complex decision boundaries than their standard trained counterparts. Further, we demonstrate that the *minimum nearest-neighbour distance* R between different classes (Definition 13, page 53) is an over-estimation of the robust radius for real-world datasets and that our derived perturbation magnitude, called R^*, is



(a) Original. (b) Perturbed.

Figure 4.1: Illustration of the main idea in $\mathcal{E} = 2$ dimensions and three example sets $\mathcal{S}_j$, $j \in 0, 1, 2$. The red line describes the decision boundary (d.b.) separating the two classes within X. (a) The input distribution is separated into three linearly separable sets $\mathcal{S}_j$ of the $\mathcal{E}$ nearest-neighbours of a sample. (b) If $\mathcal{S}_j$ changes such that the samples are collinear, $\mathcal{S}_j$ is no longer linearly separable and the complexity of the decision boundary increases.

a more appropriate upper bound.

This chapter is structured as follows. First, in Section 4.2 (page 84) we theoretically derive a dataset-specific, model-agnostic, upper bound R^* on the perturbation magnitude δ over which provably a geometrically more complex decision boundary is required. Then, in Section 4.3 (page 94) we compute this bound for several real-world image benchmarks and show the implications for robust training with magnitudes above this bound.

4.2 Bounding the Increase in Decision Boundary Complexity

The robustness of a classifier $f : \mathbb{R}^{1\times\mathcal{E}} \to \mathbb{R}^y$ (Definition 6, page 15) is usually given as an upper bound on a perturbation magnitude $\delta \in \mathbb{R}_+$ in some p-norm under which perturbations of the samples do not cause the model to change its classification decision. We derive a model-agnostic lower-bound $R^* \in \mathbb{R}_+$ on the perturbation magnitude δ over which provably a geometrically more complex decision boundary is required.

The basic idea behind deriving such a bound is to separate the input distribution $X \in \mathbb{R}^{l\times\mathcal{E}}$ into l linearly separable sets $\mathcal{S}_1, ..., \mathcal{S}_l$ of $\mathcal{E}$ nearest neighbours, where l is

the number of samples and $\mathcal{E}$ is the ambient dimension. Then, we investigate the perturbation magnitudes that are required to remove the linear separability property of each $\mathcal{S}_j$. This approach allows us to treat the geometric complexity of the original un-perturbed decision boundary as the unknown base case and only investigate the increase in geometric complexity that occurs for a certain perturbation magnitude. The intuition behind this approach is illustrated in Figure 4.1 (page 84). In Figure 4.1a (page 84) the original decision boundary is displayed. For illustration purposes only three linearly separable sets $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$ are encircled. If the two data samples marked by the cross and the one marked by the dot, are perturbed such that all samples within each $\mathcal{S}_j$ are collinear, as displayed in Figure 4.1b (page 84), the geometric complexity of the decision boundary has increased as each set is not linearly separable any more.

4.2.1 Separation of Data into Nearest-Neighbour Sets

We describe the procedure for binary classification ($y = 2$) with labels y_0 and y_1 and discuss its extension to the multi-class case ($y > 2$) further down.

We are given a dataset $X \in \mathbb{R}^{l \times \mathcal{E}}$ with l samples and ambient dimension $\mathcal{E}$. We separate X into l sets $\mathcal{S}_1, ..., \mathcal{S}_l$. Each $\mathcal{S}_j$, $j = 1, ..., l$, consists of a sample $x'_j \in X$ of class y_0 and its ordered $\mathcal{E}$-nearest neighbours with labels y_1 in Euclidean distance, denoted $x_1, ..., x_\mathcal{E}$. Thus, the ambient dimension $\mathcal{E}$ of the dataset also defines the cardinality of each $\mathcal{S}_j$ and each $\mathcal{S}_j$ contains exactly one element from one class and $\mathcal{E}$ elements from the other. With this definition, x_1 is the closest sample of x'_j of the other class and we define their distance as

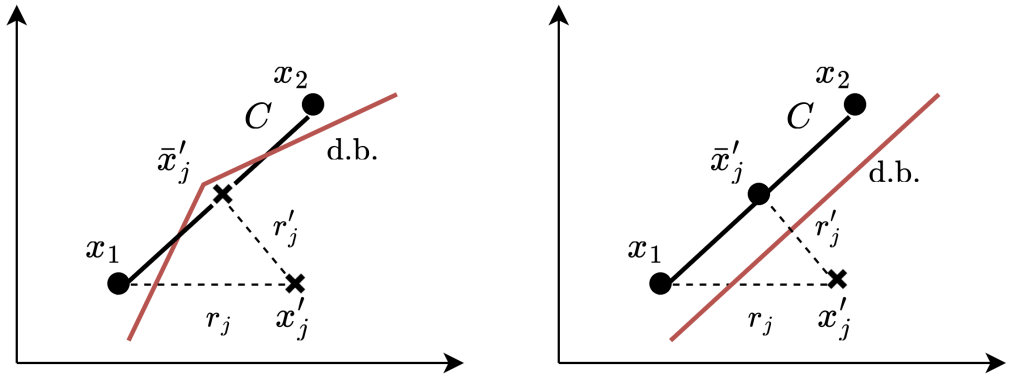
$$r_j \coloneqq ||x'_j - x_1||_2 \tag{4.1}$$

Thus,

$$\mathcal{S}_j \coloneqq \{x'_j, x_1, ..., x_\mathcal{E}\} \tag{4.2}$$

is the set of x'_j and its $\mathcal{E}$ nearest neighbours. By definition,

$$|\mathcal{S}_j| = \mathcal{E} + 1 \tag{4.3}$$



(a) $f(x'_j) = f(\bar{x}'_j)$. (b) $f(x'_j) \neq f(\bar{x}'_j)$.

Figure 4.2: Illustration of a set $\mathcal{S}_j$ in $\mathcal{E} = 2$ dimensions. (a) If oracle $\mathcal{O}$ (Definition 5, page 14) does not assign a class change between x'_j and $\bar{x}'_j$, the decision boundary's (d.b.) complexity increases. (b) If $\mathcal{O}$ assigns a class change to $\bar{x}'_j$, $\mathcal{S}_j$ is still linearly separable, however, if $r'_j < 0.5r_j$, robust training for $\delta \geq r'_j$ introduces label noise.

where $|\cdot|$ denotes the cardinality of a set. As the *Vapnik–Chervonenkis dimension* (VC-dimension) of a linear classifier is also equal to $\mathcal{E} + 1$ [20], each $\mathcal{S}_j$ is linearly separable as long as $\{x'_j, x_1, ..., x_\mathcal{E}\}$ are not collinear. Hence, x'_j can be separated from $\{x_1, ..., x_\mathcal{E}\}$ with a linear classifier with perfect accuracy, that is with no wrong classifications. The assumption of non-collinearity of the nearest-neighbour set is reasonable as real-world distributions are usually high-dimensional and do not lie on perfectly flat manifolds [233]. If the data were to lie on a flat manifold, perturbations orthogonal to it would either not change or decrease the complexity of the decision boundary.

4.2.2 The Decision Boundary of Nearest-Neighbour Sets

Each set $\mathcal{S}_j$ is, by construction, linearly separable and the samples are assumed not to be collinear. In Section 4.3 (page 94) we find that this assumption is always valid for real-world datasets in their original pixel space. The basic idea is to project x'_j onto the convex hull of $x_1, ..., x_\mathcal{E}$, denoted as $\mathcal{C}(\{x_1, ..., x_\mathcal{E}\})$, to make the samples in $\mathcal{S}_j$ collinear. Then, the set is not linearly separable any more and $\mathcal{S}_j$ requires not one but two connected hyperplanes to be separated. We denote the projection of x'_j onto the convex hull of its nearest neighbours as $\bar{x}'_j$ which can be computed by

solving the optimisation problem

$$\begin{aligned} \bar{x}'_j &\coloneqq \text{argmin}_{\hat{x}\in\mathcal{C}(\{x_i\}_{i=1}^{\varepsilon})} ||x'_j - \hat{x}||_2 \text{ s.t.} \\ \hat{x} &= \sum_{i=1}^{d} w_i x_i,\ 0 \leq w_i \leq 1,\ \sum_{i=1}^{d} w_i = 1 \end{aligned} \tag{4.4}$$

The above optimisation problem can be solved efficiently by computing a linear support vector classifier (SVC) [20] for each $\mathcal{S}_j$ and determining the distance between x'_j and the SVC's separating hyperplane which is called the *margin*. We describe this algorithm in Section 4.2.5 (page 92) and its pseudocode is displayed in Algorithm 1 (page 92).

The optimisation problem in Equation 4.4 (page 87) is solved by finding the sample $\hat{x}$ for which $||x'_j - \hat{x}||_2$ is minimised, which is then referred to as $\bar{x}'_j$. We define this minimal distance between x'_j and its distance-minimising convex-hull projection as

$$r'_j \coloneqq ||x'_j - \bar{x}'_j||_2 \tag{4.5}$$

We discuss the relationship between the minimal nearest-neighbour distance r_j (Equation 4.1, page 85), the minimal convex hull distance r'_j (Equation 4.5, page 87) and the SVC's margin m in Section 4.2.3 (page 88).

Class memberships of convex hull projections

So far we assumed that when x'_j is projected onto the convex hull of its nearest neighbours, the ground truth label does not change. If for the given dataset the 2-norm is a valid proxy for semantic similarity, then if $0.5r_j > r'_j$, this assumption is reasonable. However, for real-world distributions, such as natural images, this assumption is not valid. In this case one has to distinguish between three possible scenarios of the class membership of $\bar{x}'_j$. We refer to $\mathcal{O}$ as the *oracle* (Definition 5, page 14) that is capable of assigning the ground truth label to any sample. The oracle's output is a distribution over possible labels and taking the *argmax* of this output yields the ground truth label. An example for an oracle might be a human observer, and possibly even a neural network displaying perfect levels of accuracy and robustness. We only consider the case in which $0.5r_j > r'_j$ because then the

robust radius of $\mathcal{S}_j$ is overestimated by $0.5r_j$. We discuss the opposite scenario in Section 4.2.3 (page 88).

No class change (NCC): If no class change between x'_j and its corresponding $\bar{x}'_j$ occurs, so

$$\text{argmax } \mathcal{O}(\bar{x}'_j) = \text{argmax } \mathcal{O}(x'_j) \tag{4.6}$$

i.e. the oracle assigns the same label to both, then a geometrically more complex decision boundary is required for robust training with perturbation magnitudes $\delta \geq r'_j$.

Class change (CC): If a class change occurs, i.e.

$$\text{argmax}\mathcal{O}(\bar{x}'_j) \neq \text{argmax}\mathcal{O}(x'_j) \tag{4.7}$$

so the oracle does not assign the same class to x'_j and $\bar{x}'_j$ any more, then a geometrically more complex decision boundary is not required. However, in this case $0.5r_j$ is not the actual robust radius of the distribution. Therefore, robust training for perturbation magnitudes $\delta \geq r'_j$ introduces label noise since $\bar{x}'_j$ is wrongly labelled.

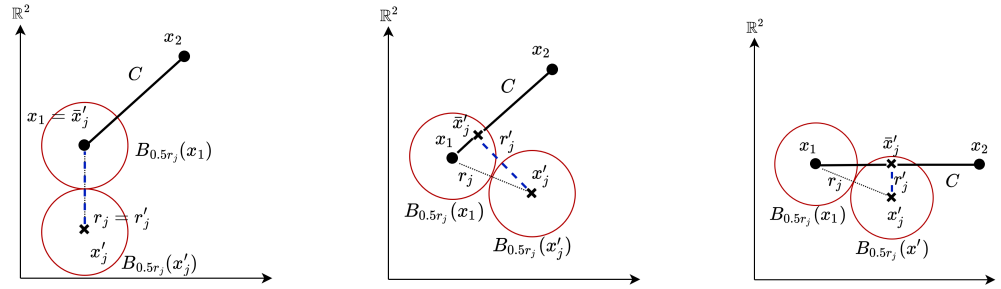
Ambiguous class (AC): Finally, if $\bar{x}'_j$ cannot be assigned a ground truth label and the oracle's output is uniform over all labels,

$$\mathcal{O}(\bar{x}'_j) = \left[\frac{1}{y}\right]^{1\times y} \tag{4.8}$$

then $\bar{x}'_j$ lies within the low-density regions between classes. In this case, a geometrically more complex decision boundary is not required. However, as in the scenario in which no class change occurs, robust training for perturbation magnitudes $\delta \geq r'_j$ introduces label noise again, because $0.5r_j$ overestimates the actual robust radius of the distribution.

4.2.3 Properties of Nearest-Neighbour Sets

As discussed above, the relationship between r_j and r'_j is crucial for the analysis. The maximum robust radius of $\mathcal{S}_j$ is equal to $0.5r_j$, so half of minimal nearest neighbour



(a) $r_j^{\text{crit}} = \frac{r'_j}{0.5r_j} = 2$ (b) $r_j^{\text{crit}} = \frac{r'_j}{0.5r_j} > 1$ (c) $r_j^{\text{crit}} = \frac{r'_j}{0.5r_j} < 1$

Figure 4.3: Illustration of nearest-neighbour sets $\mathcal{S}_j$ for $\mathcal{E} = 2$ dimensions. (a) A more complex decision boundary is not required as $||x'_j - \bar{x}'_j||_2 = ||x'_j - x_1||_2 = r'_j = r_j$. (b) A more complex decision boundary is not required as $||x'_j - \bar{x}'_j||_2 = r'_j > 0.5r_j$. (c) A more complex decision boundary is required as $||x'_j - \bar{x}'_j||_2 = r'_j < 0.5r_j$.

distance. We define

$$r_j^{\text{crit}} := \frac{r'_j}{0.5r_j} \tag{4.9}$$

for the given $\mathcal{S}_j$ to distinguish between the following two scenarios.

If $r'_j > 0.5r_j$ and thus $r_j^{\text{crit}} > 1.0$, then $0.5r_j$ is the actual robust radius of $\mathcal{S}_j$. This scenario is depicted in Figure 4.3b (page 89) for $(\mathcal{E} = 2)$-dimensional ambient space. In this case the Euclidean ball of radius $0.5r_j$ centred at x'_j,

$$B_{0.5r_j}(x'_j) = \{x : ||x - x'_j||_2 \leq 0.5r_j\} \tag{4.10}$$

does not intersect with the convex hull $\mathcal{C}(\{x_1, ..., x_\mathcal{E}\})$ of the $\mathcal{E}$ nearest neighbours of x'_j.

On the other hand, if $r'_j < 0.5r_j$ and thus $r_j^{\text{crit}} < 1.0$, then $0.5r_j$ overstates the actual robust radius of $\mathcal{S}_j$. In this case, the Euclidean ball of radius $0.5r_j$ centred at x'_j as defined in Equation 4.10 (page 89), overlaps with the convex hull $\mathcal{C}(\{x_1, ..., x_\mathcal{E}\})$ of the $\mathcal{E}$ nearest neighbours of x'_j. In this scenario the analysis in the previous Section 4.2.2 (page 87) applies and we illustrate it in Figure 4.3c (page 89).

In Figure 4.3a (page 89) we display the scenario in which the projection of x'_j to the convex hull of its nearest neighbours $\mathcal{C}(\{x_i\}_{i=1}^{\mathcal{E}})$ is equivalent to x_1 and $r_j = r'_j$.

4.2.4 Extension to the Entire Dataset

Up until this point, we have focused our analysis on a single set $\mathcal{S}_j$. In this section, we aggregate over all l sets (since $X \in \mathbb{R}^{l \times \mathcal{E}}$) and introduce the equivalents of r_j (Equation 4.1, page 85), r'_j (Equation 4.5, page 87) and r_j^{crit} (Equation 4.9, page 89) that hold for the entire dataset.

The maximum robust radius that is applicable to samples is the minimum over all the minimal nearest-neighbour distances. We define this quantity as

$$R \coloneqq \min_{j \in 1,\dots,l}(\{r_j\}) \tag{4.11}$$

which is equivalent to the definition of R by Yang et al. [17]. The minimum over all distances to the convex hull for all sets is defined as

$$R^* \coloneqq \min_{j \in 1,\dots,l}(\{r'_j\}) \tag{4.12}$$

Finally,

$$R_j^{\text{crit}} \coloneqq \frac{r'_j}{0.5 \cdot \min_{i \in 1,\dots,l}(\{r_i\})} = \frac{r'_j}{0.5R} \tag{4.13}$$

is the equivalent of r_j^{crit} for the entire dataset. Its interpretation is the same with the exception that it utilises the global robust minimum nearest neighbour distance R. If $R_j^{\text{crit}} > 1.0$, so $r'_j > 0.5R$, then for set $\mathcal{S}_j$, $0.5R$ is the actual robust radius. Further, with the definitions from above, it follows that $R^* \leq R$.[1]

When aggregating over all sets $\mathcal{S}_j$,

$$R^{\text{crit}} \coloneqq \frac{\min_{j \in 1,\dots,l}(\{r'_j\})}{0.5 \cdot \min_{j \in 1,\dots,l}(\{r_j\})} = \frac{R^*}{0.5R} \tag{4.14}$$

encodes the relationship between the minimum convex hull projection distance R^* and the minimum robust radius $0.5R$ where R is the minimum nearest neighbour distance. The interpretation of R^{crit} is equivalent to the interpretation of r_j^{crit} (Equation 4.9, page 89) for the entire dataset. For $R^{\text{crit}} > 1.0$, $0.5R$ is the actual robust radius of the distribution whereas for $R^{\text{crit}} < 1.0$, $0.5R$ overestimates the robust

[1] The equation $R^* = R$ implies that the projection of x'_j on the convex hull of its nearest neighbours is a member of the set $\mathcal{S}_j$. This means that locally the manifold of training data is densely sampled. Thus, $R^* \neq R$ implies that the data distributions contains 'gaps'.

radius.

Definition of critical points

We refer to those points $\bar{x}'_j$ for which locally $r'_j < 0.5r_j$, so $r_j^{\text{crit}} < 1$, as *critical* as they require a locally more complex decision boundary under norm-bounded robustness scenarios and cause r_j to be an overestimation of the actual robust radius,

$$\{\bar{x}\}_{\text{local}}^{\text{crit}} \coloneqq \{\bar{x}'_j : r_j^{\text{crit}} < 1,\ j = 1, ..., l\} \tag{4.15}$$

Conversely, we define those points for which $R_j^{\text{crit}} < 1$ as

$$\{\bar{x}\}_{\text{global}}^{\text{crit}} \coloneqq \{\bar{x}'_j : R_j^{\text{crit}} < 1,\ j = 1, ..., l\} \tag{4.16}$$

It follows that $|\{\bar{x}\}_{\text{global}}^{\text{crit}}| \leq |\{\bar{x}\}_{\text{local}}^{\text{crit}}|$. Note that in the multi-class case ($|y| > 2$) a single x'_j can have multiple associated $\bar{x}'_j$ that are elements of $\{\bar{x}\}_{\text{local}}^{\text{crit}}$ or $\{\bar{x}\}_{\text{global}}^{\text{crit}}$, possibly one for every other class in the dataset. Thus,

$$0 \leq |\{\bar{x}\}_{\text{global}}^{\text{crit}}| \leq |\{\bar{x}\}_{\text{local}}^{\text{crit}}| \leq l(y-1) \tag{4.17}$$

where l is the number of samples in $X \in \mathbb{R}^{l \times \mathcal{E}}$ and y is the number of unique class labels.

Multi-class classification

The method described above for the binary scenario ($y = 2$) can easily be extended to multi-class classification ($y > 2$). Instead of determining the set of nearest neighbours $\mathcal{S}_j$ once for the single other class, the computation is repeated $(y-1)$-times for all other classes. The rationale from above holds, as we simply restrict the $B_{0.5r_j}(x'_j)$-ball (Equation 4.10, page 89) to not intersect with any convex hull of nearest neighbours of any of the other $(y-1)$ classes. So, the method scales linearly with the number of classes y in X. Then,

$$r'_j \coloneqq \min_{i \in y \setminus y'_j} (\{r'_j(y_i)\}) \tag{4.18}$$

and

$$r_j \coloneqq \min_{i \in y \setminus y'_j} \left(\{ r_j(y_i) \} \right) \tag{4.19}$$

where $y \setminus y'_j$ denotes the set of unique labels without label y'_j of x'_j and $r'_j(y_i)$ and $r_j(y_i)$ denote the equivalents of r'_j and r_j defined in Equation 4.5 (page 87) and Equation 4.1 (page 85), respectively, computed for class y_i. We always report the results for all classes in a particular dataset, unless stated otherwise.

4.2.5 Algorithm for Convex Hull Projection

Algorithm 1 R^* computation

procedure PROJECTION(X, Y)
  Dataset $X \in \mathbb{R}^{l \times d}$ ▷ l samples with extrinsic dimension $\mathcal{E}$
  Labels $y \in \mathbb{N}^l$ ▷ Labels of X
  for j in $1 \cdots l$ **do**: ▷ Iterate over all samples in X
    $x'_j \leftarrow X[j]$
    $y'_j \leftarrow y[j]$
    for i in unique($y \setminus y'_j$) **do**: ▷ Iterate over labels other than y'_j
      $y_i \leftarrow y[i]$ ▷ $y_i \neq y'_j$
      $\{x_i\}_{i=1}^{d} \leftarrow d$-NearestNeighbours($x'_j, y_i, d$) ▷ NNs of x'_j with label y_i
      $\mathcal{S}_j \leftarrow \{x_1, ..., x_\mathcal{E}, x'_j\}$ ▷ Lin. sep. set $\mathcal{S}$ as cardinality $|\mathcal{S}_j| = \mathcal{E} + 1$
      $h \leftarrow$ SupportVectorClassifier($\mathcal{S}$) ▷ Get max-margin hyperplane h
      $\hat{x} \leftarrow$ OrthogonalProjection(x', h) ▷ Project x' on hyperplane h
      $\bar{x} \leftarrow$ Reflection($\hat{x}, x'$) ▷ Reflect x' around projection $\hat{x}$
      $r'_j(y_i) \leftarrow ||x'_j - \bar{x}'_j||_2$ ▷ r'_j for x'_j and class y
      $r_j(y_i) \leftarrow ||x'_j - x_1||_2$ ▷ r_j for x'_j and class y
    end for
    $r'_j \leftarrow \min_i \left(\{ r'_j(y_i) \} \right)$ ▷ Minimal r'_j over all classes
    $r_j \leftarrow \min_i \left(\{ r_j(y_i) \} \right)$ ▷ Minimal r_j over all classes
  end for
  $R^* \leftarrow \min_{j \in 1,...,l} (\{ r'_j \})$ ▷ Minimal r'_j over all samples
end procedure

The pseudocode to compute R^* and the values it is derived from is displayed in Algorithm 1 (page 92). We iterate over all l samples within $X \in \mathbb{R}^{l \times \mathcal{E}}$ and for each we compute the $\mathcal{E}$ nearest neighbours in Euclidean distance for each of the $y - 1$ classes. Then, we train a support vector classifier (SVC) on the corresponding set $\mathcal{S}_j \coloneqq \{x'_j, x_1, ..., x_\mathcal{E}\}$. As the cardinality of $|\mathcal{S}_j|$ is equal to the VC-dimension of a linear classifier, i.e. $\mathcal{E} + 1$, the SVC exhibits zero errors and maximises the margin m between x'_j and $\{x_i\}_{i=1}^{\mathcal{E}}$. It simply follows that $2m = r'_j \leq r_j$, where $r_j = 2m$

implies that $\bar{x}'_j = x_1$. Computation of $\bar{x}'_j$ is straightforwardly done by orthogonally projecting x'_j onto the hyperplane learned by the SVC. Then, the reflection of x'_j around this projection yields $\bar{x}'_j$ which implies $2m = r'_j$. This computation is efficient and deterministic. The main bottleneck of Algorithm 1 (page 92) is the computation of the $\mathcal{E}$ nearest neighbours of each sample in X.

Computation of $\bar{x}'$ for $\mathcal{S}_j \neq \mathcal{E} + 1$

Defining $\mathcal{S}_j$ such that $|\mathcal{S}_j| > \mathcal{E}+1$ might result in sets that are not linearly separable and our analysis is not applicable to those. Defining $\mathcal{S}_j$ such that $|\mathcal{S}_j| < \mathcal{E} + 1$ results in a vacuous r'_j because it will not be the smallest lower bound any more. In Section 4.3 (page 94) we demonstrate this for the CIFAR-10 dataset.

4.2.6 Conclusions

Computing the geometric complexity of a classifier's decision boundary, so the number of connected hyperplanes needed to approximate it, is an open problem for deep neural networks. Thus, testing whether robust training requires geometrically more complex decision boundaries than standard training, is an open question in the literature. In this section we circumvented this problem by introducing a model-agnostic algorithm. The algorithm relies on well-studied concepts such as the computation of nearest neighbour distances and the training of support vector classifiers and is therefore computationally efficient and deterministic. The algorithm computes an upper bound R^* on the perturbation magnitude δ over which a geometrically more complex decision boundary is provably required for the given dataset. Further, it determines whether the nearest neighbour distance R overestimates the actual robust radius of the dataset which is important to avoid the introduction of label noise during training.

In the next section we run Algorithm 1 (page 92) for several commonly used image benchmarks.

Table 4.1: Results of running Algorithm 1 (page 92) for real image benchmarks. The results are intuitive. The simple datasets, MNIST and FASHION, are well-separated and $0.5R$ accurately determines the maximum robust radius. However, for the more sophisticated datasets, $0.5R$ overestimates the robust radius and R^* is the actual robust radius.

	R	$0.5R$	R^*	R^{crit}	$\lvert\{\bar{x}\}^{\text{crit}}_{\text{local}}\rvert$	$\frac{1}{lc}\lvert\{\bar{x}\}^{\text{crit}}_{\text{local}}\rvert$	$\lvert\{\bar{x}\}^{\text{crit}}_{\text{global}}\rvert$	$\frac{1}{lc}\lvert\{\bar{x}\}^{\text{crit}}_{\text{global}}\rvert$
SVHN	1.577	0.788	0.255	0.323	132061	0.200	2501	0.004
CIFAR-10	2.751	1.375	0.578	0.421	26608	0.059	132	0.000
FASHION	1.599	0.799	0.906	1.133	811	0.002	0	0.000
MNIST	2.398	1.199	1.654	1.379	0	0.000	0	0.000

4.3 Decision Boundary Complexity of Common Image Benchmarks

In the previous section we introduced Algorithm 1 (page 92) to compute a lower bound R^* for the perturbation magnitude δ over which provably a geometrically more complex decision boundary is required. In this section, we compute this bound for several real-world image benchmarks and investigate the implications of robust training with $\delta \geq R^*$.

4.3.1 R^* for Real Image Benchmarks

We run Algorithm 1 (page 92) on the MNIST, FASHION, SVHN and CIFAR-10 datasets. We always use exact nearest neighbour search over the entire original train set. SVHN contains two mislabelled samples which we remove from the dataset prior to the computation[2].

In Table 4.1 (page 94), we display the results. For the MNIST and FASHION datasets they confirm the common knowledge that those are well-separated. Since $R^* > 0.5R$, the robust radius is accurately described by $0.5R$. Nevertheless, as R^* defines a lower bound, no definitive statement can be made about increases in the geometric complexity of the decision boundaries for robust training.

For the more sophisticated benchmarks SVHN and CIFAR-10 we observe that the nearest neighbour distance R is an overestimation of the actual robust radius,

[2]The original SVHN train set contains three labelling errors. The images with the indices 25235 and 65043 are the same but once correctly labelled as a 5 and once wrongly as a one. The image with the index 25235 does not contain a number but is labelled as a nine. The two wrongly labelled images are removed before all computations.

Figure 4.4: Example image-pairs of $\{\bar{x}\}^{\text{crit}}_{\text{global}}$ (right) their associated x'_j (left) for CIFAR-10. A single x'_j can be associated with multiple $\bar{x}'_j \in \{\bar{x}\}^{\text{crit}}_{\text{global}}$, possibly one for all other classes. The images $\bar{x}'_j$ are strongly blurred versions of their corresponding x'_j and so of ambiguous class membership.

Figure 4.5: Example image-pairs of $\{\bar{x}\}^{\text{crit}}_{\text{global}}$ (right) their associated x'_j (left) for SVHN. A single x'_j can be associated with multiple $\bar{x}'_j \in \{\bar{x}\}^{\text{crit}}_{\text{global}}$, possibly one for all other classes. The images $\bar{x}'_j$ are mostly strongly blurred versions of their corresponding x'_j and so of ambiguous class membership.

as $R^* < 0.5R$ and thus $R^{\text{crit}} < 1$. As a result, for both datasets $\{\bar{x}\}^{\text{crit}}_{\text{global}}$ are non-empty and it follows that they require a locally more complex decision boundary for perturbation magnitudes δ with $0.5R \geq \delta \geq R^*$.

$\bar{x}'_j$ are low-density samples

In Section 4.2.2 (page 86) we showed that the exact interpretation of R^* relies on the ground truth label of the projections $\bar{x}'_j$. The question of class membership cannot be answered by some distance metric using an p-norm because for real-world image datasets they are usually a bad proxy for semantic similarity. Thus, we visually investigate several $\bar{x}'_j$.

In Figure 4.4 (page 95) and Figure 4.5 (page 95) we display several randomly sampled example images from $\{\bar{x}\}^{\text{crit}}_{\text{global}}$ and their associated x'_j for the CIFAR-10

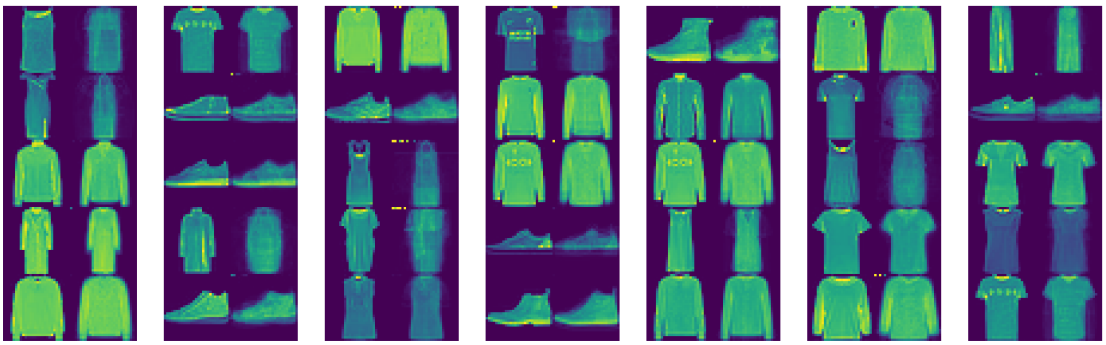


Figure 4.6: Example image-pairs of $\{\bar{x}\}^{\text{crit}}_{\text{local}}$ (right) their associated x'_j (left) for FASHION. Multiple x'_j are associated with elements from $\{\bar{x}\}^{\text{crit}}_{\text{local}}$ for different classes. FASHION does not contain any globally critical points (see Table 4.1, page 94).

Table 4.2: Results for $|\mathcal{S}_j| = 2$. The bound r'_j is vacuous and the results falsely contradict the experimental findings presented in Table 4.1 (page 94).

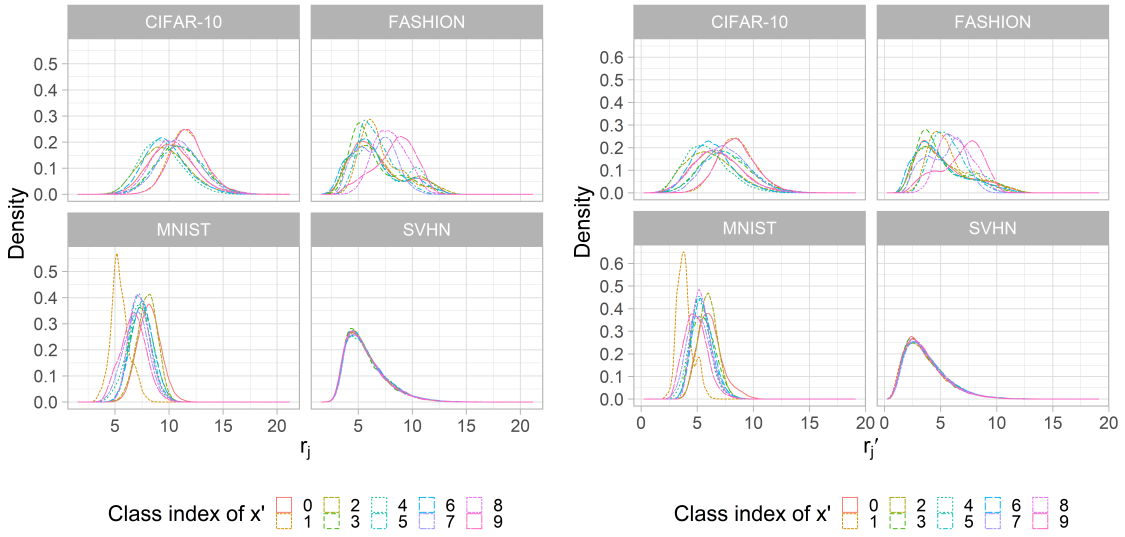
	R	$0.5R$	R^*	R^{crit}_j	$\{\bar{x}\}^{\text{crit}}_{\text{local}}$	$\frac{\lvert\{\bar{x}\}^{\text{crit}}_{\text{local}}\rvert}{l(c-1)}$	$\{\bar{x}\}^{\text{crit}}_{\text{global}}$	$\frac{\lvert\{\bar{x}\}^{\text{crit}}_{\text{global}}\rvert}{l(c-1)}$
CIFAR-10	2.751	1.375	2.313	1.682	0	0.000	0	0.000

and SVHN datasets, respectively. For both datasets we find that the majority of $\bar{x}'_j$ are strongly blurred versions of their corresponding x'_j and do not contain a clearly recognisable object. Therefore, those samples are of ambiguous class membership and lie in the low-density region between classes. Thus, robust training for perturbations magnitudes $\delta \geq R^*$ introduces label noise which is known to hurt accuracy as well as robustness [86]. We empirically confirm this for the CIFAR-10 and SVHN datasets in Section 4.3.3 (page 103).

As shown in Table 4.1 (page 94), MNIST and FASHION do not contain any globally critical points ($|\{\bar{x}\}^{\text{crit}}_{\text{global}}| = 0$) as $r'_j > 0.5R$ for all $j \in 1, ..., l$, and thus $R^{\text{crit}} > 1.0$. In Figure 4.6 (page 96) we display some locally critical points for FASHION which are clearly recognisable as valid class members which we also confirm in Section 4.3.4 (page 108).

Computation of $\bar{x}'$ for $|\mathcal{S}_j| \neq \mathcal{E} + 1$

As mentioned in Section 4.2.5 (page 92), choosing $|\mathcal{S}_j| \neq \mathcal{E} + 1$, could result in sets that are not linearly separable or in vacuous bounds on δ. To show that this is true, we run Algorithm 1 (page 92) on the CIFAR-10 dataset with $|\mathcal{S}_j| = 2$. So, x'_j is



(a) Density of r_j for all $x'_j \in X$.

(b) Density of r'_j for all $x'_j \in X$.

Figure 4.7: Distributions of r_j and r'_j.

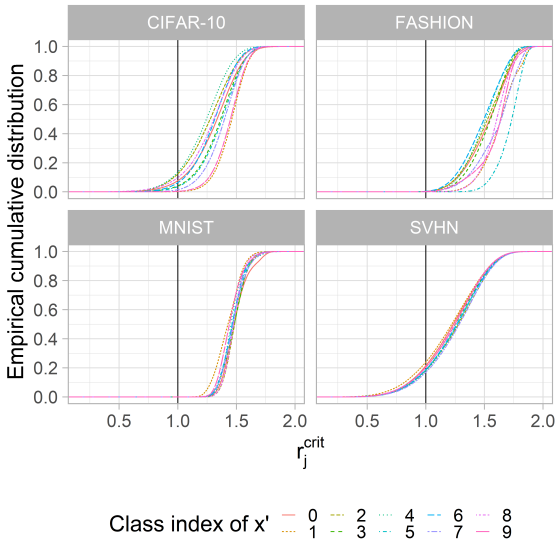
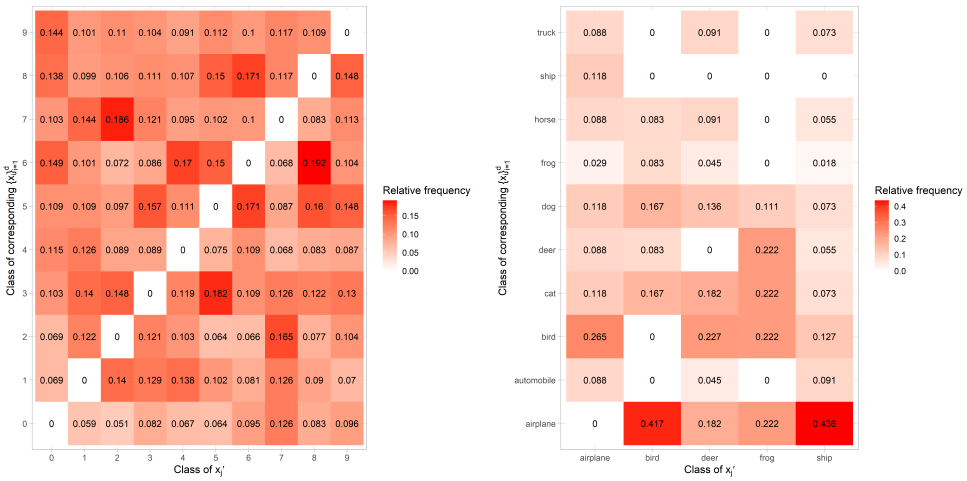


Figure 4.8: Cumulative distribution of r_j^{crit} for all $x'_j \in X$.

projected onto the line segment of its two nearest neighbours x_1 and x_2. As one can observe in Table 4.2 (page 96) the results differ completely from those presented in Table 4.1 (page 94). For $|\mathcal{S}_j| = 2$, r'_j is a vacuous bound on the perturbation magnitude.

Additional statistics

In Figure 4.7 (page 97) we display the densities of r_j and r'_j in 2-norm for all used datasets with respect to the class of x'_j. One can observe that the distributions have



(a) SVHN. (b) CIFAR-10.

Figure 4.9: Class pairs for x'_j and their corresponding $\bar{x}'_j$ for $x'_j \in \{\bar{x}\}^{\text{crit}}_{\text{global}}$.

similar shapes but different modes, except for the SVHN dataset.

In Figure 4.8 (page 97) we display the empirical cumulative distribution functions of r_j^{crit} for all datasets. For CIFAR-10 the number of r_j^{crit} varies significantly for each class while for SVHN they are roughly the same.

In Figure 4.9 (page 98) we display for every sample $x'_j \in \{\bar{x}\}^{\text{crit}}_{\text{global}}$ the label of x'_j and the distribution of labels of the corresponding $\bar{x}'_j$ inherited from the label of the samples $\{x_i\}_{i=1}^{\mathcal{E}}$. For SVHN one can observe that the critical samples lie in close proximity to several, if not all, samples of other classes. This uniform distribution of distances is also visible in Figure 4.7 (page 97) and Figure 4.8 (page 97). Contrary, for CIFAR-10 only the classes *airplane*, *bird*, *deer*, *frog* and *ship* contain critical samples $\{\bar{x}\}^{\text{crit}}_{\text{global}}$ and the nearest neighbours come from a few dominating classes. For the critical samples with labels *bird* and *ship*, for example, the corresponding $\{x_i\}_{i=1}^{\mathcal{E}}$ are of classes *airplane* with a relative frequency of 41.7% and 43.6%, respectively. This might be due to the common uniform background of shades of blue or grey shared by these samples. This observation highlights that robust radii need to be chosen class dependent.

4.3.2 Decision Boundary Complexity of Robust Models

In addition to the visual investigation of the image pairs $(x'_j, \bar{x}'_j)$ in the previous section, we gather the predictions and confidences of thirteen state-of-the-art robust models on $\{\bar{x}\}^{\text{crit}}_{\text{global}}$ from CIFAR-10. These models are obtained again from

Table 4.3: Predictions and confidences of model f for $\{\bar{x}\}^{\text{crit}}_{\text{global}}$ for CIFAR-10. Confidence values are reported as: mean $\pm$ standard deviation. NCC denotes no predicted class change by f and CC denotes a predicted class change between x'_j and $\bar{x}'_j \in \{\bar{x}\}^{\text{crit}}_{\text{global}}$. The robustly trained models do not predict class changes in the vast majority of cases and assign these samples significantly lower confidences than the non-robust models.

Model f		$f(x'_j) = f(\bar{x}'_j)$ (NCC)		$f(x'_j) \neq f(\bar{x}'_j)$ (CC)	
		Fraction	Confidence	Fraction	Confidence
Addepalli et al. [241]	Non-robust	0.64	0.925 ± 0.134	0.36	0.817 ± 0.170
	Robust	0.91	0.499 ± 0.128	0.09	0.281 ± 0.039
Andriushchenko et al. [142]	Non-robust	0.62	0.887 ± 0.154	0.38	0.708 ± 0.192
	Robust	0.79	0.535 ± 0.164	0.21	0.373 ± 0.098
Augustin et al. [202]	Non-robust	0.58	0.813 ± 0.192	0.42	0.666 ± 0.169
	Robust	0.95	0.314 ± 0.162	0.05	0.243 ± 0.051
Ding et al. [183]	Non-robust	0.52	0.913 ± 0.171	0.48	0.826 ± 0.174
	Robust	0.93	0.979 ± 0.057	0.07	0.791 ± 0.113
Engstrom et al. [242]	Non-robust	0.56	0.757 ± 0.215	0.44	0.571 ± 0.181
	Robust	0.86	0.537 ± 0.189	0.14	0.338 ± 0.113
Hendrycks et al. [147]	Non-robust	0.50	0.907 ± 0.144	0.50	0.778 ± 0.173
	Robust	0.76	0.905 ± 0.150	0.24	0.749 ± 0.161
Kireev et al. [77]	Non-robust	0.50	0.814 ± 0.195	0.50	0.726 ± 0.200
	Robust	0.61	0.905 ± 0.151	0.39	0.691 ± 0.209
Modas et al. [143]	Non-robust	0.60	0.900 ± 0.155	0.40	0.768 ± 0.176
	Robust	0.68	0.632 ± 0.155	0.32	0.419 ± 0.125
Rade et al. [87] (*ddpm*)	Non-robust	0.51	0.815 ± 0.178	0.49	0.601 ± 0.190
	Robust	0.95	0.670 ± 0.176	0.05	0.488 ± 0.068
Rade et al. [87] (*extra*)	Non-robust	0.52	0.867 ± 0.181	0.48	0.608 ± 0.182
	Robust	0.75	0.603 ± 0.153	0.25	0.413 ± 0.102
Rebuffi et al. [148]	Non-robust	0.50	0.844 ± 0.177	0.50	0.650 ± 0.171
	Robust	0.94	0.643 ± 0.202	0.06	0.389 ± 0.083
Rice et al. [145]	Non-robust	0.54	0.863 ± 0.168	0.46	0.677 ± 0.204
	Robust	0.92	0.635 ± 0.200	0.08	0.401 ± 0.072
Wong et al. [141]	Non-robust	0.43	0.873 ± 0.183	0.57	0.707 ± 0.185
	Robust	0.74	0.645 ± 0.187	0.26	0.458 ± 0.095

RobustBench and referred to as *robust* models. In addition, we re-initialise these architectures and re-train only with the Adam on the original train set to remove their robust representation. Thus, the re-trained models are their *non-robust* counterparts.

In Table 4.3 (page 99) we observe two major differences between the robust and non-robust models. Firstly, the non-robust models assign high confidences to $\{\bar{x}\}^{\text{crit}}_{\text{global}}$. As the visual inspection shows that $\{\bar{x}\}^{\text{crit}}_{\text{global}}$ are part of the low-density region between classes, high confidence scores indicate a poorly calibrated classifier. In contrast, the robust models usually assign significantly lower confidences to these low-density samples, a result that would be expected from a well-performing classifier. In a subsequent work to ours, Grabinski et al. [203] investigate the calibration of robust models more thoroughly and reached the same conclusion. Secondly,

Table 4.4: Influence of the ambient dimension $\mathcal{E}$ for CIFAR-10. For all ambient dimensions $R^{\text{crit}} < 1.0$. Thus, no qualitative changes are induced by varying the image sizes.

	$\mathcal{E}$	R	$0.5R$	R^*	R^{crit}	$\lvert\{\bar{x}\}^{\text{crit}}_{\text{local}}\rvert$	$\frac{1}{lc}\lvert\{\bar{x}\}^{\text{crit}}_{\text{local}}\rvert$	$\lvert\{\bar{x}\}^{\text{crit}}_{\text{global}}\rvert$	$\frac{1}{lc}\lvert\{\bar{x}\}^{\text{crit}}_{\text{global}}\rvert$
34x34	3468	2.843	1.422	0.497	0.350	38827	0.086	283	0.001
32x32 (org.)	3072	2.751	1.375	0.578	0.421	26608	0.059	132	0.000
30x30	2700	2.514	1.257	0.486	0.386	36167	0.080	236	0.001
28x28	2352	2.324	1.162	0.467	0.402	34784	0.077	214	0.000
26x26	2028	2.157	1.079	0.444	0.412	33066	0.073	200	0.000

we find that robust and non-robust models differ in their predictions of whether a class change has occurred between x'_j and its corresponding $\bar{x}'_j$. Whereas the robust models predict in most of the cases that no class change occurs, the non-robust models predict class changes in half of the cases. As the addition of $\{\bar{x}\}^{\text{crit}}_{\text{global}}$ to the train set increases the geometric complexity of the decision boundary, robust models learn more complex decision boundaries. Thus, we experimentally confirm this previously made hypothesis [55, 118, 126, 131, 239] for real-world architectures and datasets. These results also partially explain why robust training has a greater sample complexity than standard training, since the geometric complexity of decision boundaries is known to increase the sample complexity of deep neural networks (see Chapter 3, page 56).

Results are Independent of the Ambient Dimension $\mathcal{E}$

The point $\bar{x}'_j$ minimises the Euclidean distance between x'_j and the convex hull $\mathcal{C}(\{x_i\}_{i=1}^{\mathcal{E}})$ (see Equation 4.5, page 87). Since the convex hull is defined by the $\mathcal{E}$ nearest neighbours with another label of x'_j, all quantities that are deducted from $\bar{x}'_j$ are functions of the ambient dimension $\mathcal{E}$. Therefore, we investigate whether changes of the ambient dimension change the previously computed quantities.

We report the results for CIFAR-10 in Table 4.4 (page 100). For image distributions an increase in their ambient dimension $\mathcal{E}$, so their resolution, results in higher correlations between pixels and larger Euclidean distances between images. Hence, simultaneously higher values of R and R^* are expected. Further, there is no clear relationship between $\lvert\{\bar{x}\}^{\text{crit}}_{\text{local}}\rvert$ and $\lvert\{\bar{x}\}^{\text{crit}}_{\text{global}}\rvert$ with respect to $\mathcal{E}$. Crucially, for all ambient dimensions $\mathcal{E}$, $R^{\text{crit}} > 1.0$. Thus, $0.5R$ overstates the robust radius for all ambient spaces sizes and R^* is its correct measure. These results show the

Table 4.5: Number of pixels $\lceil (p - \tilde{p}) \rceil$ (Equation 4.20, page 102) that need to be perturbed by ϵ in l_∞-norm to introduce perturbations $\delta > R^*$ in 2-norm.

	$0.5R$	R^*	p	$\lceil (p - \tilde{p}) \rceil$				
				$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$	$\epsilon = \frac{8}{255}$	$\epsilon = \frac{16}{255}$	$\epsilon = \frac{32}{255}$
SVHN	1.003	0.525	1024	4478	1120	280	70	18
CIFAR-10	1.375	0.578	1024	5439	1360	340	85	22
FASHION	0.799	0.906	784	13340	3335	834	209	53
MNIST	1.199	1.654	784	44461	11116	2779	695	174

quantities derived by Algorithm 1 (page 92) are not artefacts of high dimensional spaces but dataset specific properties and that the choice of the ambient dimension $\mathcal{E}$ only alters the results quantitatively but not qualitatively. Further, this results implies that for datasets where the extrinsic dimension is greater than the number of contained samples ($\mathcal{E} > l$), resizing the extrinsic dimension could be a feasible method to make the introduced approach usable.

Extension to the ∞-norm

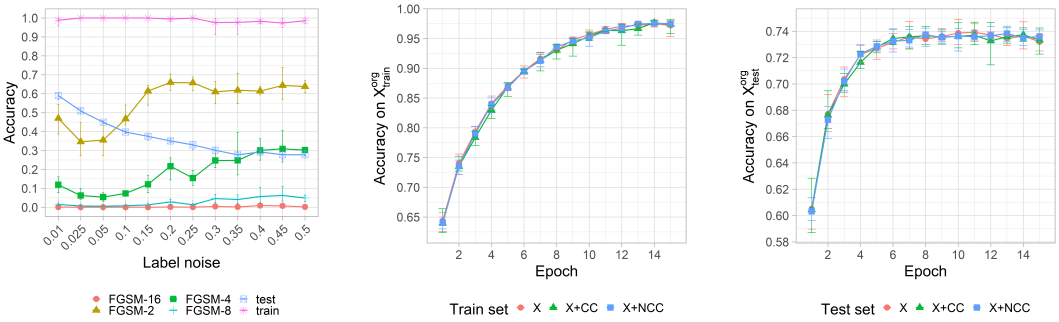
Perturbation magnitudes for robust training and data augmentation are usually given in 2- or ∞-norm. In the previous section we computed R^* in 2-norm so here we expand the analysis to the ∞-norm. Since the ∞-norm is the maximum absolute change ϵ between any two vector dimensions, we compute how many dimensions in common image benchmarks need to be changed to surpass the specific R^*-value in 2-norm. It is common practice in the robustness literature to apply the ∞-norm on the pixel level, so to ignore the colour channel. Denoting the number of pixels in a

dataset as p, with $0 \leq \tilde{p} \leq p$, it is easy to see that

$$\begin{aligned} ||x - \tilde{x}||_2 &> R^* \\ \Leftrightarrow \sqrt{\sum_{i=1}^{p}(x_i - \tilde{x}_i)^2} &> R^* \\ \Leftrightarrow \sqrt{\sum_{i=1}^{\tilde{p}-1}(x_i - \tilde{x}_i)^2 + \sum_{j=\tilde{p}}^{p}(x_j - \tilde{x}_j)^2} &> R^* \\ \Leftrightarrow \sqrt{\sum_{j=\tilde{p}}^{p}\epsilon^2} &> R^* \\ \Leftrightarrow (p - \tilde{p}) &> \left(\frac{R^*}{\epsilon}\right)^2 \end{aligned} \tag{4.20}$$

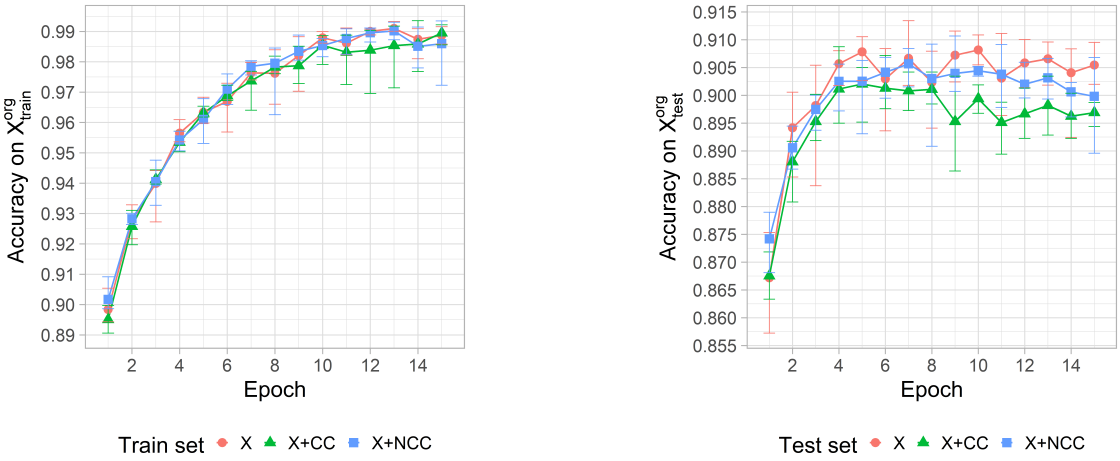
where the first sum in the third line is equal to zero because those pixels are not altered and the pixels in the second sum are all changed by ϵ due to the ∞-norm. Thus, $(p - \tilde{p})$ is the number of pixels that need to be changed by ϵ such that the resulting perturbation magnitude in 2-norm surpasses R^*. We round $(p - \tilde{p})$ to the nearest integer.

In Table 4.5 (page 101) we display the minimum number of pixels $\lceil (p - \tilde{p}) \rceil$ that need to be changed to surpass R^* in 2-norm when perturbations ϵ are applied in ∞-norm. For CIFAR-10, for example, we observe that perturbation magnitudes in ∞-norm of $\epsilon = 4/255$ require $1,360$ pixels to be altered. As this is more than the original number of $1,024$ pixels, R^* is not surpassed in 2-norm. In general, we observe that for the common perturbation magnitude $\epsilon = 8/255$ only a small fraction of pixels need to be altered in both SVHN and CIFAR-10 to surpass the threshold R^* in 2-norm. In following section we show that including samples with perturbation magnitude $\delta \geq R^*$ leads to reduced generalisation performance.



(a) (b) (c)

Figure 4.10: Adding globally critical samples to CIFAR-10. Error-bars denote minimum and maximum over five runs. (a) Mean accuracy on the train and test set and against FGSM-i/255, $i \in \{2, 5, 8\}$ attacks for different levels of label noise introduced by $\{\bar{x}\}^{\text{crit}}_{\text{global}}$. (b) Mean accuracy on $X^{\text{org}}_{\text{train}}$ during training with $\{\bar{x}\}^{\text{crit}}_{\text{global}}$. (c) Mean accuracy on $X^{\text{org}}_{\text{test}}$ during training with $\{\bar{x}\}^{\text{crit}}_{\text{global}}$.



(a) Accuracy on original train set.

(b) Accuracy on original test set.

Figure 4.11: Adding globally critical samples to SVHN. Error-bars denote minimum and maximum over five runs. (a) Mean accuracy on $X^{\text{org}}_{\text{train}}$ during training with $\{\bar{x}\}^{\text{crit}}_{\text{global}}$. (b) Mean accuracy on $X^{\text{org}}_{\text{test}}$ during training with $\{\bar{x}\}^{\text{crit}}_{\text{global}}$.

4.3.3 Robust Training with $\delta \geq R^*$ for real-world datasets

In Section 4.2 (page 84) we derived R^* theoretically for arbitrary datasets. We discussed that the implications of robust training for perturbation magnitudes $\delta \geq R^*$ depend on the class membership of those samples for which $r'_j < 0.5 r_j$. Then, in Section 4.3.1 (page 94) we showed that for sophisticated real-world benchmarks $\{\bar{x}\}^{\text{crit}}_{\text{global}}$ lie within the low-density region between classes. Including these samples that do not display a clearly distinguishable object is equivalent to the addition of label noise which is known to hurt robustness [86]. In this section we show that the

addition of samples with perturbations $\delta \geq r'_j$ for $r'_j < 0.5r_j$ indeed decreases the performance of classifiers according to several metrics on CIFAR-10 and SVHN.

Extension by the globally critical points

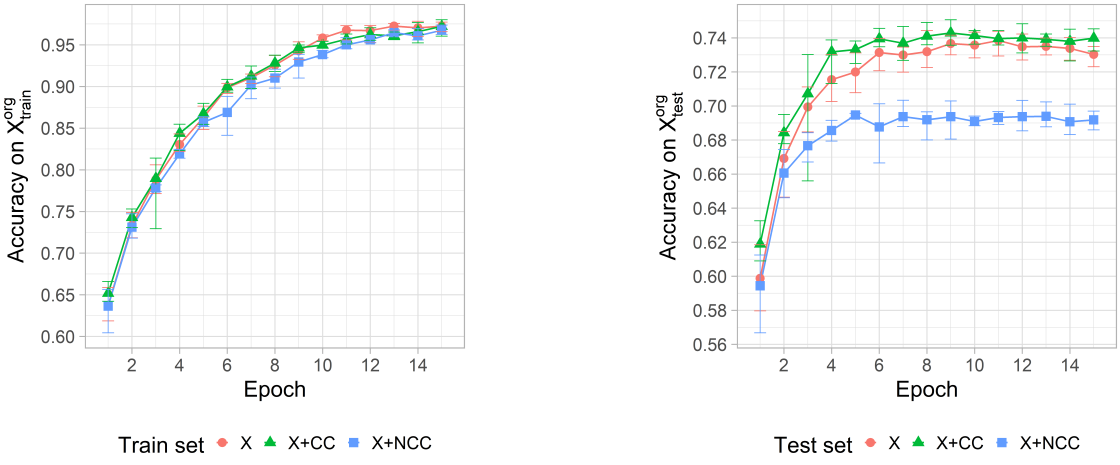
As $|\{\bar{x}\}^{\text{crit}}_{\text{global}}| = 132$ for CIFAR-10 (Table 4.1, page 94), their addition is not measurably impacting generalisation performance as observed in Figure 4.10c (page 103). Thus, to simulate different levels of label noise we add random samples from the original train set $X^{\text{org}}_{\text{train}}$ to $\{\bar{x}\}^{\text{crit}}_{\text{global}}$ to obtain new train sets with different relative amounts of original and critical samples and therefore different amounts of label noise. This experimental setup roughly follows a similar one by Sanyal et al. [86]. We train a convolutional neural network on these datasets and measure its accuracy on the original train and test set as well as against FGSM attacks (Definition 8, page 21) of different strengths.

For CIFAR-10 we observe in Figure 4.10a (page 103) that with increasing label noise test accuracy deteriorates while adversarial accuracy against FGSM attacks increases. Although, due to the small train set, test accuracy is already low, adding samples with no visible class object further deteriorates test accuracy as the model is likely biased towards learning superficial textural clues. Train accuracy on the other hand is not hurt, as those samples can simply be memorised. As $\{\bar{x}\}^{\text{crit}}_{\text{global}}$ are defined by having $r'_j \leq 0.5R$, the distance between $\bar{x}'_j$ and x'_j is small and thus small-norm perturbations as those introduced by FGSM do not result in wrong predictions as the network interpolates between $\bar{x}'_j$ and x'_j.

In Figure 4.11 (page 103) we present the results for SVHN when $\{\bar{x}\}^{\text{crit}}_{\text{global}}$ are added to the original train set. As $|\{\bar{x}\}^{\text{crit}}_{\text{global}}| = 2,501$ (Table 4.1, page 94), the addition of globally critical points to the train set makes a non-negligible difference. Thus, generalisation performance is decreased when perturbation magnitudes $\delta \geq R^*$ are introduced. This affirms the hypothesis in that the reason for the *visibly* unchanged performance for CIFAR-10 is the comparably low number of $\{\bar{x}\}^{\text{crit}}_{\text{global}}$. Thus, robust training for $\delta \geq R^*$, so the addition of $\{\bar{x}\}^{\text{crit}}_{\text{global}}$ to the train set, also negatively affects CIFAR-10 training.

Figure 4.12: Example image-pairs of $\{\bar{x}\}^{\text{crit}}_{\text{local}}$ (right) their associated x'_j (left) for CIFAR-10. Multiple x'_j are associated with elements from $\{\bar{x}\}^{\text{crit}}_{\text{local}}$ for different classes.

Figure 4.13: Example image-pairs of $\{\bar{x}\}^{\text{crit}}_{\text{local}}$ (right) their associated x'_j (left) for CIFAR-10. Multiple x'_j are associated with elements from $\{\bar{x}\}^{\text{crit}}_{\text{local}}$ for different classes.



(a) Accuracy on original train set.

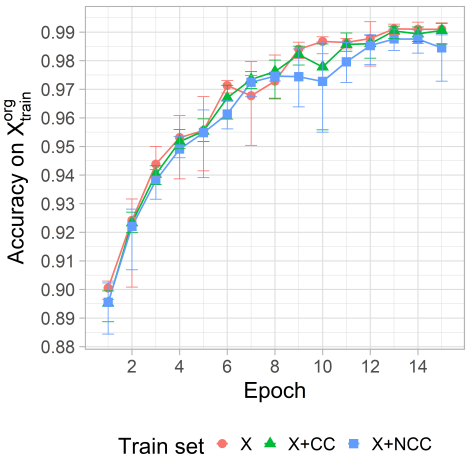
(b) Accuracy on original test set.

Figure 4.14: Adding locally critical samples to CIFAR-10. Error-bars denote minimum and maximum over five runs. (a) Mean accuracy on $X^{\text{org}}_{\text{train}}$ during training with $\{\bar{x}\}^{\text{crit}}_{\text{local}}$. (b) Mean accuracy on $X^{\text{org}}_{\text{test}}$ during training with $\{\bar{x}\}^{\text{crit}}_{\text{local}}$.

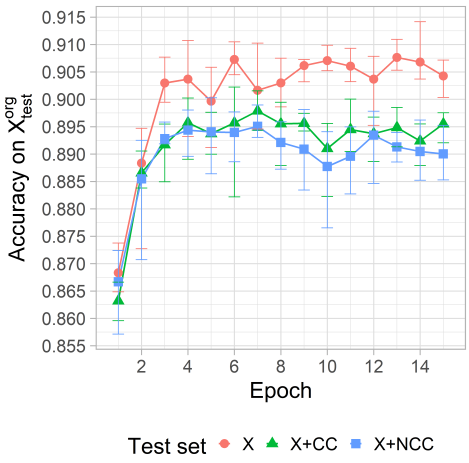
Table 4.6: Accuracy against noise- (top) and blur-perturbations (bottom) created with the routine of Hendrycks et al. [41] for CIFAR-10 and $\{\bar{x}\}^{\text{crit}}_{\text{local}}$. Networks trained on X+CC and X+NCC exhibit better robustness against small-norm noise-perturbations but worse robustness against large-norm blur perturbations. Example images can be found in Figure 4.12 (page 105).

	Gaussian	Shot	Impulse	Speckle
X	0.521	0.501	0.537	0.557
X+CC	0.526	0.504	0.546	0.562
X+NCC	**0.600**	**0.581**	**0.584**	**0.621**

	Zoom	Defocus	Gaussian	Glass	Fog	Brightness	Contrast
X	**0.638**	**0.691**	0.324	0.703	**0.352**	0.694	0.328
X+CC	0.625	0.689	0.273	**0.704**	0.292	**0.709**	0.246
X+NCC	0.607	0.647	**0.423**	0.656	0.342	0.648	**0.343**



(a) Accuracy on original train set.



(b) Accuracy on original test set.

Figure 4.15: Adding locally critical samples to SVHN. Error-bars denote minimum and maximum over five runs. (a) Mean accuracy on $X^{\text{org}}_{\text{train}}$ during training with $\{\bar{x}\}^{\text{crit}}_{\text{local}}$. (b) Mean accuracy on $X^{\text{org}}_{\text{test}}$ during training with $\{\bar{x}\}^{\text{crit}}_{\text{local}}$.

Extension by the locally critical points

It is common practice in adversarial training to pick a single perturbation magnitude δ for all samples under the assumption that no class change is induced by its application. However, this procedure is suboptimal and error-prone as upper-bounds on δ can be influenced by labelling errors in the original train set. Thus, more recent robust training methods work with instance-specific perturbation magnitudes (e.g. [183]).

To further show that the addition of samples for which $\delta \geq R^*$ leads to reduced generalisation performance, we create two new train sets by appending either $\{\bar{x}\}^{\text{crit}}_{\text{local}}$ with the label of its corresponding x'_j (no class change, NCC), denoted X+NCC,

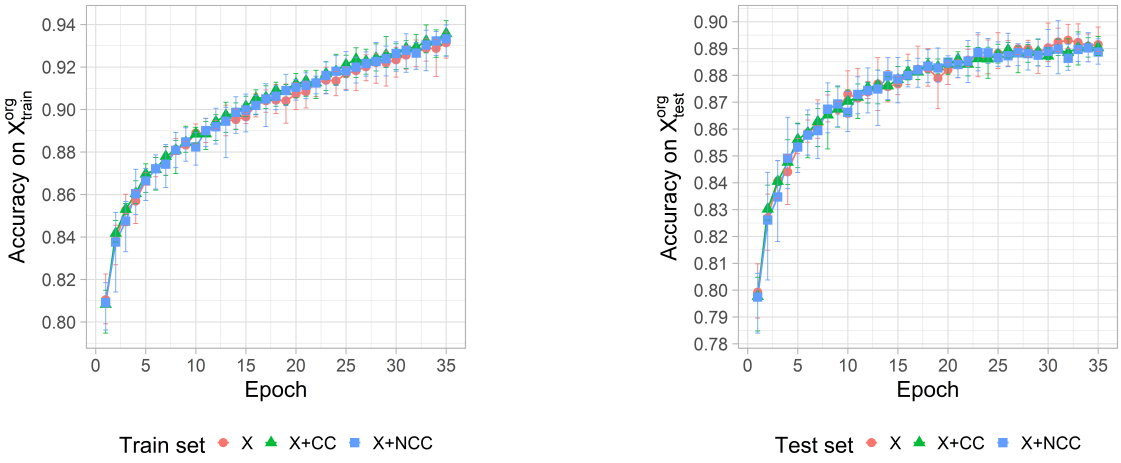
Table 4.7: Predictions and confidences of model f for $\{\bar{x}\}^{\text{crit}}_{\text{local}}$ for CIFAR-10. Confidence values are reported as: mean ± standard deviation. NCC denotes no predicted class change by f and CC denotes a predicted class change between x'_j and $\bar{x}'_j \in \{\bar{x}\}^{\text{crit}}_{\text{local}}$.

Model f		$f(x'_j) = f(\bar{x}'_j)$ (NCC)		$f(x'_j) \neq f(\bar{x}'_j)$ (CC)	
		Fraction	Confidence	Fraction	Confidence
Addepalli et al. [241]	Non-robust	0.43	0.872 ± 0.169	0.57	0.796 ± 0.194
	Robust	0.58	0.447 ± 0.144	0.42	0.331 ± 0.092
Andriushchenko et al. [142]	Non-robust	0.32	0.913 ± 0.148	0.68	0.824 ± 0.184
	Robust	0.62	0.479 ± 0.164	0.38	0.370 ± 0.115
Augustin et al. [202]	Non-robust	0.32	0.849 ± 0.177	0.68	0.784 ± 0.190
	Robust	0.62	0.373 ± 0.162	0.38	0.270 ± 0.102
Ding et al. [183]	Non-robust	0.34	0.864 ± 0.169	0.66	0.798 ± 0.190
	Robust	0.57	0.904 ± 0.151	0.43	0.774 ± 0.194
Engstrom et al. [242]	Non-robust	0.34	0.839 ± 0.179	0.66	0.776 ± 0.191
	Robust	0.57	0.495 ± 0.182	0.43	0.372 ± 0.120
Hendrycks et al. [147]	Non-robust	0.32	0.898 ± 0.143	0.68	0.847 ± 0.165
	Robust	0.40	0.869 ± 0.171	0.61	0.788 ± 0.197
Kireev et al. [77]	Non-robust	0.30	0.827 ± 0.181	0.70	0.785 ± 0.193
	Robust	0.37	0.838 ± 0.188	0.63	0.756 ± 0.212
Modas et al. [143]	Non-robust	0.34	0.921 ± 0.138	0.66	0.868 ± 0.167
	Robust	0.46	0.528 ± 0.165	0.54	0.420 ± 0.148
Rade et al. [87] (*ddpm*)	Non-robust	0.35	0.895 ± 0.146	0.64	0.850 ± 0.167
	Robust	0.65	0.615 ± 0.181	0.35	0.446 ± 0.137
Rade et al. [87] (*extra*)	Non-robust	0.29	0.878 ± 0.160	0.71	0.805 ± 0.179
	Robust	0.53	0.490 ± 0.146	0.47	0.380 ± 0.103
Rebuffi et al. [148]	Non-robust	0.30	0.841 ± 0.170	0.70	0.780 ± 0.180
	Robust	0.65	0.569 ± 0.185	0.35	0.410 ± 0.128
Rice et al. [145]	Non-robust	0.35	0.906 ± 0.149	0.65	0.843 ± 0.185
	Robust	0.57	0.626 ± 0.196	0.43	0.466 ± 0.156
Wong et al. [141]	Non-robust	0.35	0.888 ± 0.158	0.65	0.816 ± 0.182
	Robust	0.56	0.535 ± 0.179	0.44	0.414 ± 0.131

or the label of its corresponding nearest neighbours $\{x_i\}^{\mathcal{E}}_{i=1}$ (class change, CC), denoted X+CC, to the original train set $X^{\text{org}}_{\text{train}}$. We train a network on each of these three datasets and report accuracies on the original train and original test set. In Figures 4.12 (page 105) and 4.13 (page 105) we display some randomly sampled images from $\{\bar{x}\}^{\text{crit}}_{\text{local}}$ for CIFAR-10 and SVHN, respectively.

For CIFAR-10, we can observe in Figures 4.14a (page 105) and Figure 4.14b (page 105) that while train accuracy is not hurt, test accuracy deteriorates when trained on X+NCC. This is likely due to $\{\bar{x}\}^{\text{crit}}_{\text{local}}$ biasing the model towards learning superficial textural clues which does not deteriorate train but does reduce generalisation performance. The addition of $\{\bar{x}\}^{\text{crit}}_{\text{local}}$ when assigned a different label than x'_j improves generalisation performance. This is likely due to the network interpolating between $\{x_i\}^{\mathcal{E}}_{i=1}$ which appears to help for CIFAR-10.

Finally, we also test the CIFAR-10 models against the benchmark of common



(a) Accuracy on original train set.

(b) Accuracy on original test set.

Figure 4.16: Adding locally critical points to FASHION. Error-bars denote minimum and maximum over five runs. (a) Mean accuracy on $X^{\text{org}}_{\text{train}}$ during training with $\{\bar{x}\}^{\text{crit}}_{\text{local}}$. (b) Mean accuracy on $X^{\text{org}}_{\text{test}}$ during training with $\{\bar{x}\}^{\text{crit}}_{\text{local}}$.

perturbations proposed by Hendrycks et al. [41] (see Figure 5.1, page 116 for example images). We obtain similar results to the label noise experiments above. In Table 4.6 (page 106) we observe that accuracy against small-norm noise perturbations is increased whereas accuracy against large-norm blur perturbations is mostly decreased. Intuitively, a flat loss surface around training points or obfuscated gradients [56] help to protect against small-norm changes, whereas large-norm changes need to be defended against by learning semantic concepts.

For SVHN we observe that both train sets X+NCC and X+CC reduce generalisation performance as observed in Figure 4.15 (page 106).

The results for CIFAR-10 and SVHN show that the maximum perturbation magnitudes for robust training need to be chosen carefully as they can deteriorate the generalisation to accuracy and robustness benchmarks while train accuracy is unharmed.

4.3.4 Robust Training with $\delta \geq R^*$ for toy datasets

The FASHION dataset does not contain any $\{\bar{x}\}^{\text{crit}}_{\text{global}}$, so $0.5R$ is its actual robust radius. In Figure 4.16 (page 108) we observe no difference in train and test performance when the $\{\bar{x}\}^{\text{crit}}_{\text{local}}$ are added to the original train set. This result is expected as $R^* > 0.5R$ and the dataset is known to be simple and well separated which is

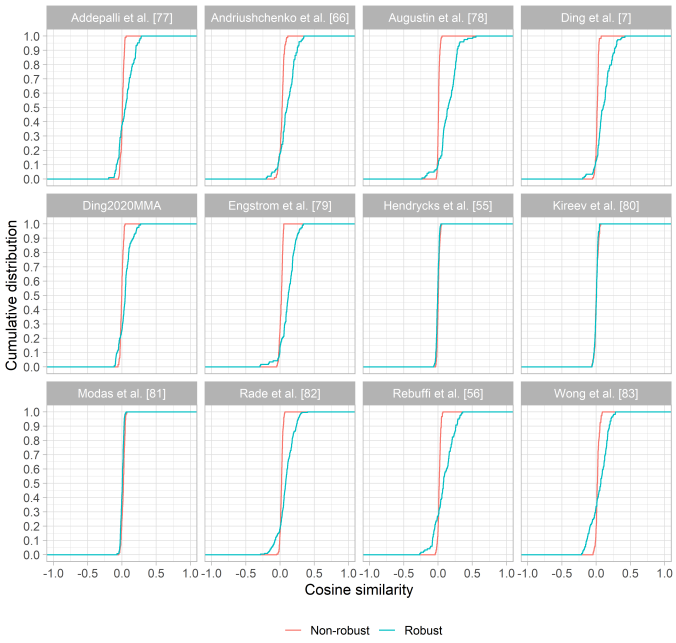


Figure 4.17: Cumulative distribution of cosine similarities between $\bar{x}'_j - x'_j$ and $x_{\text{adver}} - x'_j$.

further proven in Figure 4.6 (page 96) as there is no class change between x'_j and their associated $\bar{x}'_j$.

4.3.5 The Relationship between DeepFool and R^*

DeepFool [57] is an algorithm that finds an adversarial example x_j^{adver} while minimising the perturbation ϵ (see Definition 7, page 20).

If we assume to have a *perfectly* robust model f with robust radius of $\geq 0.5r'_j$ for every sample (for complex datasets: $= 0.5r'_j$) then the closest point on the decision boundary is precisely in the middle of the line segment between x'_j and $\bar{x}'_j$, with distance $0.5r'_j$ from both. As the model's gradient points towards the direction of steepest ascent, it points towards the closest point on the decision boundary. As the model is perfectly robust, the minimum distance to this point is $0.5r'_j$ and the point on the decision boundary is exactly the aforementioned middle of the line segment between x'_j and $\bar{x}'_j$. Thus, this middle of the line segment should also be the adversarial sample x_j^{adver} that is found by DeepFool, as DeepFool minimises the introduced perturbation magnitude. As a result, more robust models should have the vectors $\bar{x}'_j - x'_j$ and $x_{\text{adver}} - x'_j$ being more aligned.

We compute the cosine similarities

$$\tilde{c}_j = \text{cosine}(x_{\text{adver}} - x'_j, \bar{x}'_j - x'_j) \in [-1, 1] \tag{4.21}$$

for all samples $x_j \in X$ and display the distributions for the robust and non-robust neural network pairs. We observe that for the robust models the distributions contain significantly more positive values, even though those are nowhere near being perfectly robust. Thus, our method could be used in conjunction with DeepFool to investigate the distance and shape of the decision boundary around the critical points which are those that determine the introduction of label noise and the geometric complexity of the decision boundary.

4.3.6 Conclusions

We run Algorithm 1 (page 92) on several commonly used image datasets and find that for real-world ones the nearest neighbour distance R overestimates the robust radius. Contrary, our introduced quantity R^* yields an accurate estimation. We demonstrate that training with perturbation magnitudes $0.5R \geq \delta \geq 0.5R^*$ introduces label noise which reduces generalisation performance and robustness, while train accuracy is unharmed. The computation of R^* is not qualitatively affected by the ambient dimension of the dataset and is both computationally efficient and deterministic. Finally, we show that state-of-the-art robust models indeed learn geometrically more complex decision boundaries compared to their standard trained counterparts.

4.4 Summary and Discussion

To investigate the hypothesis that robust neural networks learn geometrically more complex decision boundaries than their non-robust counterparts, we introduced a novel algorithm to compute a bound R^* on the minimum perturbation magnitude over which provably a geometrically more complex decision boundary is required. This bound is computed for a single dataset, so is model-agnostic, and thus its derivation circumvents all of the problems that are usually encountered when non-

linear decision boundaries are studied. Due to its use of well-studied concepts in the literature, its computation is efficient and deterministic.

Empirically we show for real-world image benchmarks that R^* also bounds the perturbation magnitude over which label noise is introduced. We train neural networks on dataset containing samples that were perturbed by magnitudes greater than R^* and find decreased generalisation performance and decreased robustness to large norm-perturbations. However, the introduction of these samples does not result in decreased train accuracy or decreased robustness to small-norm perturbations. Hence, on first sight training with large perturbation magnitudes does not appear to have any shortcomings. It is only on second sight when the performance reduction of such training is visible. As the commonly used minimal nearest neighbour distance R [17] overestimates the robust radius, R^* is a superior measure of it. Finally, we also show that state-of-the-art robust models learn geometrically more complex decision boundaries than their non-robust counterparts.

Chapter 5

Intrinsic Dimension and Feature Sharing of Robust Representations

5.1 Introduction

As discussed in Chapter 2 (page 9) deep neural networks used for object recognition still display a substantial discrepancy between standard and robust accuracy (see Figure 2.6, page 44). Since this discrepancy exists for small-norm synthetic as well as small- and large-norm natural perturbations, it is a widespread problem in real-life applications. To close this gap a wide variety of robust training methods (Definition 11, page 11) have been proposed (see Chapter 2.4, page 35). Although these methods all share the goal of achieving robustness, they differ significantly in the objective function they choose to optimise. For example, adversarial training (Definition 12, page 35) optimises a minimax-problem for robustness against worst case samples, margin maximisation enlarges the distance to the decision boundary around train samples (see Section 2.4.7, page 41) and data augmentation and dataset enlargement methods introduce helpful priors via the addition of novel samples (see Section 2.4.2, page 36). While recent work shows a slowly closing robustness gap between humans and neural networks [80], the mechanisms of robust training methods driving this progress are, however, still not fully understood. Since neural networks are increasingly used in safety-critical applications, it is crucial to know what mechanisms these methods rely on.

Thus far, robust training methods have largely been compared by their in- and

out-of-distribution accuracy and sometimes with respect to their train and inference time. In this work we study and compare the internal mechanisms that underlie their robustness. We employ measurements of the intrinsic dimension (Definition 2, page 11) and apply them to the hidden representations learned by these neural networks. The intrinsic dimension of a distribution denotes the number of variables required to describe its variations and can thus be seen as one possible measure of its complexity (see Section 2.1.1, page 10).

In the robustness literature some works study the effect of regularising the rank of weight- [243] or activation-matrices [160] on adversarial robustness. These regularisation methods, respectively, can or will reduce the intrinsic dimension of hidden representations, and often result in increased robustness to adversarial attacks. As robustness refers to the application of perturbations that do not change the ground truth label, methods that reduce the intrinsic dimension should naturally remove perturbations that are not semantically meaningful. Thus, the success of dimensionality reduction methods in removing adversarial perturbations is not surprising (see Section 2.4.3 page 37). However, current state-of-the-art methods, like the ones mentioned above, do not explicitly regularise the intrinsic dimension of hidden representations to be small. Instead they opt for other tasks whose effect on the intrinsic dimension is not obvious and has not yet been systematically studied. Thus, an investigation of the intrinsic dimension of representations obtained by robust training methods is still pending and could offer insights into the mechanisms that drive robustness.

Independently of the robustness literature, the intrinsic dimension of hidden representations was studied for standard training [214] and dropout and weight-decay regularised networks [217]. These works offer interesting insights into neural network properties by connecting the generalisation ability to in-distribution samples and the feature extraction process to the intrinsic dimension of their hidden representations (see Section 2.5, page 46).

Ansuini et al. [214] study the intrinsic dimension of hidden layer representations for standard trained networks without label noise. They find that earlier layers inflate the intrinsic dimension and later layers decrease it. Further, they report that the intrinsic dimension of the last hidden layer representation correlates neg-

atively with the test accuracy. Following that work, Brown et al. [217] investigate the intrinsic dimension of hidden representations for networks regularised by either dropout or weight-decay and their connection to the generalisation performance. They show that in these networks, increased generalisation performance co-occurs with a decrease in last-layer intrinsic dimension and an increase in the maximum intrinsic dimension across layers.

However, since the ability to generalise accurately and robustly are two separate problems, i.e. robust training induces different features compared to standard training [82], the conclusions made in previous works may not be readily applicable to robust training. Thus, studying the intrinsic dimension of robust representations is complementary to previous works.

This chapter is structured as follows. In Section 5.2 (page 115) we study the effect on the intrinsic dimension of several simple data augmentation techniques that affect generalisation performance and robustness in different ways. We find that for adversarial training, improvements in robustness to adversarial and small-norm natural perturbations co-occur with significant decreases in the peak intrinsic dimension across layer representations. Then, we show that this observation holds also for a wide variety of other state-of-the-art robust training methods. These methods, although not explicitly designed to do so, all reduce the peak intrinsic dimension compared to a standard trained baseline. In Section 5.3 (page 126) we argue that the observed reductions in the intrinsic dimension are due to robust training methods implicitly regularising the networks to learn shared features across semantically similar classes and representing semantically dissimilar classes separately. In contrast, we show that standard training does not yield this specialisation of features by class. Further, we show that these shared features cannot be obtained by simply disentangling semantically different classes which explains observations made by previous authors that reducing proximity of representations of semantically similar classes slightly improves robustness [220, 244].

5.2 Robust Training and Intrinsic Dimension

In this section we investigate the intrinsic dimension of representations that have been obtained by some form of robust training and compare them to those that have been obtained by standard training.

First, we provide an introductory experiment in Section 5.2.2 (page 116) for the relatively simple FASHION dataset and a weak form of adversarial training. Thus, this experiment provides a baseline against which more complex datasets and other state-of-the-art robust training methods can be compared. Then, in Section 5.2.3 (page 119) we expand the analysis to the CIFAR-10 dataset and use a representative set of recently proposed robust training methods from the literature.

5.2.1 Experimental Setup

In this section we describe the metrics that we use throughout this chapter. We re-use those that have been employed in the literature before to assure comparability with our work.

Estimating the Intrinsic Dimension

As defined in Section 2.1.2 (page 16), a deep neural network $f : \mathbb{R}^{1\times\mathcal{E}} \to \mathbb{R}^{y}$ can be written as the composition of L layers, i.e. $f(x) = f_L(f_{L-1}(f_{\dots}(f_1(x))))$ where $f_i(x)$ denotes the representation of $x \in \mathbb{R}^{1\times\mathcal{E}}$ retrieved after the i-th layer.

As in Chapter 3, we utilise Equation 3.14 (page 65) to compute the intrinsic dimension $\mathcal{I}_k$ of a batch of samples where k is the number of neighbours. Brown et al. [217] compute the *peak intrinsic dimension* (PID)

$$\mathrm{PID}_{\mathcal{I}_k} \coloneqq \max_{i=1,\dots,L} \{\mathcal{I}_k(f_i(x))\}_{i=1}^{L-1} \tag{5.1}$$

which describes the maximum intrinsic dimension over all layer representations and the *last-layer intrinsic dimension* (LID)

$$\mathrm{LID}_{\mathcal{I}_k} \coloneqq \mathcal{I}_k(f_{L-1}(x)) \tag{5.2}$$

which describes the intrinsic dimension of the last hidden (pre-logit) layer represen-

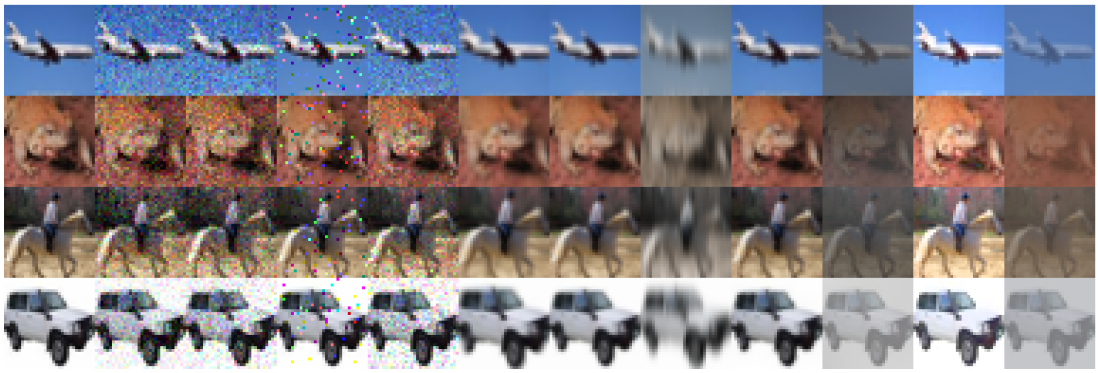


Figure 5.1: Example images created by the routine of Hendrycks et al. [41] for CIFAR-10. From left to right the images are: *Original*, *Gaussian-noise*, *Shot-noise*, *Impulse-noise*, *Speckle-noise*, *Zoom-blur*, *Defocus-blur*, *Gaussian-blur*, *Glass-blur*, *Fog-blur*, *Brightness-blur* and *Contrast-blur*.

tation and is also utilised by Ansuini et al. [214].

Measuring the Robustness

To determine the robustness of a given model against small-norm synthetic perturbations, we utilise the fast gradient sign method (Definition 8, page 21) (FGSM) and projected gradient descent (Definition 9, page 21) (PGD). The natural perturbations are created with the routine of Hendrycks et al. [41] (see Figure 5.1, page 116 for example images). The noise perturbations *Gaussian*, *Shot*, *Impulse* and *Speckle*, as well as the *Zoom*, *Defocus*, *Gaussian* and *Glass* blur perturbations are small-norm perturbations, whereas the other blur perturbations, *Fog*, *Brightness* and *Contrast*, introduce large-norm changes to the original test samples (see Table 5.1, page 117). As already noted in Section 2.2.3 (page 26), using a broad set of perturbation types and magnitudes is necessary because there appears to be a tradeoff between robustness to small- and large-norm perturbations. Later in this Chapter we report several instances of this tradeoff.

5.2.2 Experiments on FASHION

Experimental Setup

To set up some baseline behaviour for the relationship between robustness, PID and LID we first consider a simple convolutional neural network trained on the

Table 5.1: Mean distances in 2-norm between original x and perturbed x^{pert} samples created with the routine of Hendrycks et al. [41] for CIFAR-10. The *Fog*, *Brightness* and *Contrast* perturbations are large-norm perturbations while all other noise and blur perturbations induce small-norm changes. Values are reported as mean ± standard deviation over all available samples l.

	$1/l \ \sum_{i=1}^{l} \|\|x_i - x_i^{\text{pert}}\|\|_2$
Gaussian-noise	4.31 ± 0.15
Shot-noise	4.70 ± 0.53
Impulse-noise	5.36 ± 0.44
Speckle-noise	4.11 ± 0.79
Zoom-blur	3.61 ± 1.21
Defocus-blur	1.13 ± 0.32
Gaussian-blur	4.81 ± 1.29
Glass-blur	1.22 ± 0.33
Fog-blur	11.60 ± 3.72
Brightness-blur	9.07 ± 1.03
Contrast-blur	7.83 ± 2.30

FASHION dataset. We compare the representations of the following two different training methods that differ in their objective function they are trained to optimise.

Standard training (ST) refers to the setup in which the usual cross-entropy loss $\mathcal{L}$ is minimised over all available original sample-label pairs $(x, y)_{i=1}^{l}$,

$$\text{ST: } \min_\theta \mathcal{L}(x, y) \tag{5.3}$$

Adversarial training (AT), which is a special kind of robust training, minimises the following loss function,

$$\text{AT: } \min_\theta \left[\max_\epsilon \mathcal{L}(x + \epsilon, y)\right] \tag{5.4}$$

As in Section 2.4.1 (page 35) the variable θ denotes the neural network's parameters. Adversarial training solves the minimax problem described in Definition 12 (page 35). As this loss is known to be difficult to optimise, we reformulate it in practice as follows.

$$\mathcal{L}(x, y) = \mathcal{L}(x, y) + \alpha \, \mathcal{L}(x^{\text{adv}}, y) \tag{5.5}$$

where $\mathcal{L}$ is the standard cross-entropy loss, $\alpha \in \mathbb{R}_{\geq 0}$ a weighting parameter and x^{adv} is an adversarial example created for the benign sample x. In this section we utilise FGSM-ϵ (Definition 8, page 21) with perturbations magnitude ϵ to generate x^{adv}.

Table 5.2: Robustness, accuracy and intrinsic dimension for the neural networks trained on FASHION. The values α and ϵ in the first two rows denote the hyper-parameters for the loss function in Equation 5.5 (page 117). There is no tradeoff between accuracy and robustness and no tradeoff between small-norm synthetic and large-norm natural perturbations. At the bottom of the table, one can observe that PID (Equation 5.1, page 115) and LID (Equation 5.2, page 115) are both negatively correlated with robustness.

ϵ	0	4/255			8/255			16/255		
α	0	1.0	2.0	3.0	1.0	2.0	3.0	1.0	2.0	3.0
Train	0.985	0.967	0.961	0.958	0.947	0.938	0.937	0.925	0.917	0.916
Test	0.880	0.888	0.889	0.887	0.885	0.884	0.883	0.877	0.873	0.872
FGSM-4/255	0.526	0.878	0.889	0.894	0.924	0.936	0.938	0.949	0.958	0.962
FGSM-8/255	0.335	0.750	0.771	0.782	0.850	0.871	0.874	0.903	0.918	0.925
FGSM-16/255	0.140	0.534	0.561	0.575	0.699	0.728	0.743	0.810	0.837	0.853
PGD-0.01-5	0.128	0.708	0.737	0.749	0.832	0.857	0.860	0.891	0.909	0.918
Gaussian-noise	0.959	0.988	0.990	0.989	0.991	0.993	0.993	0.993	0.992	0.992
Shot-noise	0.961	0.990	0.991	0.990	0.992	0.994	0.994	0.994	0.993	0.993
Impulse-noise	0.940	0.983	0.986	0.985	0.987	0.989	0.988	0.986	0.986	0.983
Speckle-noise	0.964	0.989	0.991	0.991	0.993	0.996	0.995	0.995	0.995	0.995
Zoom-blur	0.909	0.929	0.930	0.930	0.925	0.922	0.923	0.921	0.919	0.910
Defocus-blur	0.977	0.987	0.988	0.987	0.988	0.989	0.989	0.987	0.988	0.987
Gaussian-blur	0.903	0.941	0.943	0.947	0.949	0.952	0.951	0.948	0.946	0.940
Glass-blur	0.978	0.987	0.987	0.986	0.987	0.988	0.988	0.986	0.987	0.986
Fog-blur	0.357	0.366	0.376	0.370	0.374	0.341	0.332	0.360	0.342	0.319
Brightness-blur	0.592	0.833	0.839	0.819	0.852	0.787	0.823	0.895	0.869	0.828
Contrast-blur	0.492	0.532	0.561	0.536	0.537	0.524	0.502	0.576	0.548	0.521
PID	15.328	11.118	11.024	11.038	11.602	11.382	11.756	12.179	12.689	12.383
LID	12.909	9.902	9.606	9.628	9.200	8.960	8.909	8.854	8.393	8.392

For $\alpha = 0$ the loss function is the one for standard training and for $\alpha > 0$ the loss function corresponds to adversarial training. The introduction of the α parameter balances the signal of the benign sample and the signal of the adversarial sample and leads to a more stable optimisation behaviour.

Results

In Table 5.2 (page 118) we display PID (Equation 5.1, page 115), LID (Equation 5.2, page 115) and all robustness metrics presented above for different values of α and ϵ (Equation 5.5, page 117).

We observe that robustness to all perturbation types either remains roughly the same or significantly increases. Thus, for this simple setup no tradeoff between small-norm synthetic and large-norm natural perturbations exists. Further, test accuracy is only marginally decreased and hence no accuracy-robustness tradeoff (see Section 2.2.3, page 26) exists either. Therefore, this simple setup removes confounding factors that alter the results.

As mentioned above, Ansuini et al. [214] and Brown et al. [217] report that the

LID is negatively correlated with the generalisation error. Given the commonly observed accuracy-robustness tradeoff (see Section 2.2.3, page 26), any drop in LID could therefore be due to the improved generalisation performance of the robustly trained model. However, in the setup chosen here, we rule out this confounding effect since test accuracy is not positively affected by adversarial training. Thus, the observed drop in LID is solely due to the adversarial training loss and hence LID is also negatively correlated with robustness. In addition to LID, PID is also negatively correlated with robustness.

Further, Brown et al. [217] argue that LID is a proxy for the *simplicity* of the learned hypothesis and PID for the *plausibility*. Although these results are valid for standard and weight- and dropout regularised training, they do not hold when comparing standard and adversarial training. According to the argumentation of Brown et al. [217] our results suggest that adversarial training yields simpler but less plausible representations, despite being more robust, better generalising and more interpretable (see Section 2.4.10, page 43).

In summary, we can state that interpretation and behaviour of LID and PID differ significantly for standard, regularised and robust training. Our results show that LID and PID are both negatively correlated with robustness.

5.2.3 Experiments on CIFAR-10

In this section we repeat the previous analysis for the CIFAR-10 dataset, which is a real-world image benchmark of much higher complexity than FASHION. We follow the same procedure as before, however, we expand the analysis by utilising several additional training methods that affect generalisation and robustness differently. These methods allow us to study the relationship between generalisation performance, robustness, PID and LID more thoroughly.

Custom Training Methods

We again utilise the standard and adversarial training methods defined in Equation 5.3 (page 117) and Equation 5.4 (page 117), respectively. For adversarial training we set $\alpha = 1$ and $\epsilon = 4/255$ in Equation 5.5 (page 117). In addition, we also use the following novel training methods.

Table 5.3: Robustness, accuracy and intrinsic dimension for the neural networks trained on CIFAR-10. The first row denotes the networks' training method. There is a tradeoff between accuracy and robustness and also one between small-norm synthetic and large-norm natural perturbations. At the bottom of the table, one can observe that PID is negatively correlated with robustness whereas no differences between LID exist for standard trained and adversarially trained models.

	ST	MRT	AT	GN-0.1	GN-1.0	RLT	RLTapp
Train	0.993 ± 0.001	0.991 ± 0.002	0.997 ± 0	0.99 ± 0	0.782 ± 0.079	0.999 ± 0	0.526 ± 0.001
Test	0.757 ± 0.009	0.796 ± 0.003	0.688 ± 0.004	0.738 ± 0.005	0.65 ± 0.003	0.1 ± 0.001	0.488 ± 0.006
FGSM-4/255	0.191 ± 0.013	0.209 ± 0.008	0.519 ± 0.011	0.291 ± 0.01	0.311 ± 0.013	0 ± 0	0.03 ± 0.004
FGSM-8/255	0.083 ± 0.014	0.107 ± 0.01	0.253 ± 0.011	0.127 ± 0.007	0.163 ± 0.01	0 ± 0	0.031 ± 0.011
FGSM-16/255	0.057 ± 0.015	0.086 ± 0.011	0.092 ± 0.008	0.054 ± 0.004	0.072 ± 0.004	0 ± 0	0.044 ± 0.007
PGD-0.01-5	0.001 ± 0	0 ± 0	0.189 ± 0.009	0.016 ± 0.003	0.041 ± 0.004	0 ± 0	0 ± 0
Gaussian-noise	0.639 ± 0.032	0.576 ± 0.028	0.938 ± 0.003	0.897 ± 0.004	0.916 ± 0.008	0.352 ± 0.018	0.53 ± 0.014
Shot-noise	0.606 ± 0.036	0.547 ± 0.029	0.923 ± 0.005	0.897 ± 0.002	0.908 ± 0.008	0.322 ± 0.019	0.497 ± 0.015
Impulse-noise	0.673 ± 0.013	0.655 ± 0.014	0.9 ± 0.007	0.913 ± 0.002	0.907 ± 0.011	0.299 ± 0.031	0.514 ± 0.005
Speckle-noise	0.692 ± 0.029	0.638 ± 0.023	0.943 ± 0.006	0.918 ± 0.003	0.927 ± 0.005	0.385 ± 0.026	0.56 ± 0.007
Zoom-blur	0.793 ± 0.012	0.805 ± 0.017	0.872 ± 0.011	0.871 ± 0.001	0.882 ± 0.002	0.324 ± 0.014	0.622 ± 0.014
Defocus-blur	0.9 ± 0.007	0.906 ± 0.004	0.929 ± 0.005	0.932 ± 0.002	0.955 ± 0	0.664 ± 0.014	0.787 ± 0.003
Gaussian-blur	0.378 ± 0.05	0.328 ± 0.045	0.551 ± 0.011	0.437 ± 0.016	0.534 ± 0.018	0.226 ± 0.009	0.258 ± 0.004
Glass-blur	0.917 ± 0.005	0.922 ± 0.005	0.938 ± 0.005	0.94 ± 0.001	0.958 ± 0.001	0.629 ± 0.018	0.802 ± 0.01
Fog-blur	0.43 ± 0.044	0.446 ± 0.024	0.233 ± 0	0.407 ± 0.003	0.361 ± 0.006	0.165 ± 0.008	0.323 ± 0.025
Brightness-blur	0.909 ± 0.004	0.923 ± 0.003	0.842 ± 0.009	0.897 ± 0.002	0.82 ± 0.005	0.335 ± 0.02	0.774 ± 0.002
Contrast-blur	0.393 ± 0.018	0.389 ± 0.02	0.261 ± 0.007	0.388 ± 0.008	0.348 ± 0.005	0.212 ± 0.017	0.309 ± 0.019
LID	26.4 ± 0.9	26.1 ± 0.9	27.4 ± 1	27 ± 1.8	27.2 ± 0.7	49 ± 0.8	34.6 ± 0.8
PID	59 ± 2.5	56 ± 5	38.2 ± 3.8	51.5 ± 2.5	43.1 ± 2.1	54.2 ± 5.8	64.4 ± 2.2

Mirror-reflection training (MRT) reflects all original samples around the y-axis and appends these mirrored images x^{mirr} along with their original label to the original dataset,

$$\text{MRT: } \min_\theta \left[\mathcal{L}(x, y) + \mathcal{L}(x^{\text{mirr}}, y)\right] \tag{5.6}$$

As images in CIFAR-10 display natural objects, mirror-reflections do not change the ground truth label and thus training with mirror-reflected images provides a strong additional signal for training but no robustness improvements. Adversarial training (Equation 5.4, page 117), on the other hand, improves robustness to small-norm perturbations but hurts robustness due to the accuracy-robustness tradeoff (see Section 2.2.3, page 26). Hence, mirror-reflection training and adversarial training have complementary effects on generalisation performance and robustness compared to standard training (Equation 5.3, page 117).

Further, we utilise *Gaussian-noise training* (GNT-p). As in mirror-reflection training, we append novel samples with their original labels to the original dataset. For Gaussian-noise training these novels samples are defined as $x^{\text{gauss}} = x + \gamma$, where $\gamma \sim M[\mathcal{N}(\mu = 0, \sigma = 128/255)]$ is a sample from a Gaussian-distribution with mean $\mu = 0$ and standard deviation $\sigma = 128/255$ and $M[\cdot]$ is a binary mask constructed

Table 5.4: Summary of Table 5.3 (page 120). Comparisons are made with respect to the standard trained (ST) model. The ↓-symbol indicates that a relative decrease occurred, ↑-symbol that a relative increase occurred and the symbol − that no change occurred. The number of arrow-symbols indicates the strength of the change with respect to the standard trained baseline (ST). For example, adversarial training results in a large improvement in generalisation performance compared to ST, while no changes in LID or PID are measurable. Further adversarial training greatly improves robustness to small-norm perturbations while robustness to large-norm perturbation is hurt.

		AT	MRT	GNT-0.1	GNT-1.0
	Generalisation	↓↓	↑↑↑	↓	↓↓↓
Robustness	small-norm synthetic pert.	↑↑↑	−	↑	↑↑
	small-norm natural pert.	↑↑↑	−	↑	↑↑
	large-norm natural pert.	↓↓↓	−	↓	↓↓
	LID	−	−	−	−
	PID	↓↓↓	−	↓	↓↓

from $\mathcal{E}$ independent Bernoulli trials with parameter p,

$$\text{GNT-}p\text{: } \min_\theta \left[\mathcal{L}(x, y) + \mathcal{L}(x^{\text{gauss}}, y)\right] \tag{5.7}$$

where $\mathcal{E}$ is again the ambient dimension (Definition 3, page 11). Gaussian-noise training also provides robustness benefits against small-norm perturbations compared to standard training, but also reduces generalisation performance. For all training methods we observe a tradeoff between small-norm and large-norm natural perturbations.

Finally, we also provide two training methods that involve random labels to compare the effects of memorization and learning on the intrinsic dimensionality. *Random label training* (RLT) minimises the loss over the original samples but with randomly assigned labels $\pi(y)$,

$$\text{RLT: } \min_\theta \mathcal{L}(x, \pi(y)) \tag{5.8}$$

where $\pi(y)$ is a random permutation of label y_i. In contrast, *random label training, appended* (RLTapp) generates random labels $\pi(y)$ for every original sample but then appends these random sample-label pairs to the original sample-label pairs,

$$\text{RLTapp: } \min_\theta \left[\mathcal{L}(x, y) + \mathcal{L}(x, \pi(y))\right] \tag{5.9}$$

Table 5.5: Standard and robust accuracies for the models by Addepalli et al. [241] and Andriushchenko et al. [142] which do not use any additional data. Here robust training (RT) refers to the model retrieved from the literature, whereas all other models have been trained by the author of this thesis according to the methods introduced in Section 5.2.3 (page 119).

	Addepalli et al. [241] (no add. data)					Andriushchenko et al. [142] (no add. data)				
	RT	ST	MRT	GNT-0.1	GNT-1.0	RT	ST	MRT	GNT-0.1	GNT-1.0
Train	0.889	0.988	0.996	0.997	0.994	0.836	0.981	0.989	0.983	0.976
Test	0.802	0.834	0.889	0.838	0.792	0.798	0.821	0.872	0.817	0.772
FGSM1	0.871	0.101	0.127	0.129	0.201	0.836	0.249	0.262	0.326	0.271
FGSM2	0.744	0.049	0.068	0.033	0.068	0.670	0.148	0.161	0.209	0.138
FGSM3	0.538	0.033	0.052	0.017	0.023	0.421	0.099	0.128	0.155	0.075
PGD1	0.728	0.000	0.000	0.000	0.006	0.643	0.010	0.005	0.051	0.015
Gaussian-noise	0.929	0.593	0.490	0.806	0.899	0.929	0.683	0.408	0.827	0.877
Shot-noise	0.915	0.573	0.465	0.789	0.899	0.913	0.648	0.396	0.805	0.879
Impulse-noise	0.905	0.699	0.675	0.962	0.913	0.893	0.721	0.601	0.941	0.896
Speckle-noise	0.936	0.667	0.591	0.833	0.916	0.939	0.746	0.550	0.850	0.900
Zoom-blur	0.944	0.831	0.858	0.860	0.891	0.944	0.808	0.857	0.886	0.874
Defocus-blur	0.958	0.925	0.943	0.940	0.949	0.956	0.908	0.942	0.940	0.943
Gaussian-blur	0.683	0.311	0.262	0.349	0.541	0.643	0.335	0.324	0.427	0.504
Glass-blur	0.963	0.936	0.954	0.951	0.958	0.964	0.926	0.959	0.950	0.946
Fog-blur	0.230	0.710	0.769	0.619	0.563	0.252	0.643	0.617	0.629	0.485
Brightness-blur	0.879	0.934	0.958	0.947	0.916	0.895	0.940	0.944	0.942	0.906
Contrast-blur	0.262	0.662	0.729	0.530	0.534	0.267	0.580	0.581	0.583	0.414

These methods are chosen because they provide different signals for the relationship between samples and labels to the classifier and thus have different effects on robustness and generalisation performance.

In Table 5.3 (page 120) we report the robustness metrics, PID and LID for the CIFAR-10 dataset and in Table 5.4 (page 121) we summarise these results. In contrast to the experiments with the FASHION dataset presented in Section 5.2.2 (page 116), we do not observe lower LID values for models with greater robustness. On the contrary, more robust models tend to display slightly higher LID values than non-robust models. This observation might be the result of their slightly reduced generalisation performance. In the following paragraph we observe that an increase in LID also occurs for state-of-the-art robust training methods that do not use additional samples.

Nevertheless, for all training methods PID is strongly negatively correlated with robustness, a result that is consistent with the FASHION experiment before. In the following paragraph, we test whether this correlation also holds for recently proposed state-of-the-art robust training methods and in Section 5.3 (page 126) we provide one possible explanation for this phenomenon.

Table 5.6: Standard and robust accuracies for the models by Gowal et al. [153] and Rade et al. [87] which use additional data. Here robust training (RT) refers to the model retrieved from the literature, whereas all other models have been trained by the author of this thesis according to the methods introduced in Section 5.2.3 (page 119).

	Gowal et al. [153] (add. data)					Rade et al. [87] (add. data)				
	RT	ST	MRT	GNT-0.1	GNT-1.0	RT	ST	MRT	GNT-0.1	GNT-1.0
Train	0.982	0.992	0.994	0.995	0.992	0.947	0.996	0.995	0.994	0.992
Test	0.873	0.851	0.883	0.848	0.813	0.890	0.856	0.885	0.854	0.800
FGSM1	0.884	0.017	0.029	0.050	0.073	0.879	0.017	0.028	0.057	0.092
FGSM2	0.759	0.001	0.007	0.004	0.009	0.743	0.001	0.012	0.008	0.012
FGSM3	0.540	0.002	0.007	0.001	0.002	0.503	0.001	0.009	0.004	0.001
PGD1	0.739	0.000	0.000	0.000	0.000	0.727	0.000	0.000	0.000	0.000
Gaussian-noise	0.919	0.448	0.362	0.694	0.818	0.902	0.460	0.460	0.684	0.838
Shot-noise	0.897	0.424	0.338	0.684	0.814	0.864	0.449	0.430	0.664	0.839
Impulse-noise	0.884	0.615	0.587	0.956	0.860	0.848	0.609	0.609	0.953	0.862
Speckle-noise	0.929	0.557	0.481	0.759	0.847	0.906	0.559	0.559	0.738	0.875
Zoom-blur	0.957	0.757	0.830	0.845	0.852	0.958	0.794	0.844	0.832	0.863
Defocus-blur	0.968	0.892	0.931	0.929	0.932	0.965	0.918	0.939	0.928	0.939
Gaussian-blur	0.699	0.227	0.220	0.291	0.398	0.663	0.247	0.175	0.264	0.381
Glass-blur	0.974	0.919	0.944	0.944	0.944	0.972	0.938	0.949	0.942	0.949
Fog-blur	0.301	0.621	0.655	0.585	0.477	0.262	0.674	0.708	0.630	0.521
Brightness-blur	0.937	0.948	0.952	0.942	0.929	0.944	0.942	0.943	0.939	0.937
Contrast-blur	0.284	0.582	0.613	0.538	0.435	0.262	0.606	0.666	0.586	0.483

State-of-the-art Robust Training Methods

As discussed in Section 2.4 (page 35), a large number of robust training methods have been proposed over recent years and adversarial training is one of them. Thus far we have considered adversarial training with FGSM as the training method that is explicitly designed to improve robustness to small-norm perturbations. In this section we replace this form of adversarial training with a state-of-the-art robust training method from the literature. We chose four of these methods that span most of the important robust training approaches discussed earlier (see Section 2.4, page 35).

Addepalli et al. [241] propose a novel variant of adversarial training to defend against large-norm perturbation by regularising the gradient such that the loss on benign and adversarial samples aligns. Thus, it is an example of a gradient regularisation method, that have been popular in the robust training literature (see Section 2.4.6, page 39). Gowal et al. [153] utilise a generative model trained on the original CIFAR-10 dataset and generate novel samples for the model to be trained on. Thus, it is a dataset enlargement method, and the authors report improved robustness over the baseline model. Rade et al. [87] introduce wrongly labelled samples to mitigate the accuracy-robustness tradeoff.

The methods proposed by Addepalli et al. [241] and Andriushchenko et al. [142]

Table 5.7: PID (Equation 5.1, page 115) and LID (Equation 5.2, page 115) for state-of-the-art robustly trained models and their counterparts. PID is negatively correlated with robustness while the behaviour of LID depends on whether additional data is used during training. These findings are consistent across multiple values of k (see Equation 3.14, page 65), so the reduced curvature of robust representations is not the reason for the drop in PID. Intrinsic dimension values are written mean ± standard deviation.

		PID_10	LID_10	PID_5	LID_5	PID_3	LID_3
Addepalli et al. [241]	ST	87.9 ± 11.4	13.5 ± 0.2	113.2 ± 13.8	16.4 ± 0.2	141.6 ± 14.2	20.2 ± 1.1
	RT (no add. data)	66.4 ± 1.8	13.3 ± 0.1	89.7 ± 3	16.4 ± 0.2	116.5 ± 4.1	19.7 ± 0.7
	MRT	100.3 ± 4.2	14.1 ± 0.1	128.6 ± 7.3	17.7 ± 0.3	159.4 ± 13.1	21.4 ± 0.6
	GNT-1.0	51.7 ± 1.6	16.7 ± 0.3	61.8 ± 3	20.8 ± 0.1	74.6 ± 5.6	24.8 ± 0.1
	GNT-0.1	68.6 ± 6.8	14.8 ± 0.4	85.3 ± 9.5	18.3 ± 0.3	103.6 ± 6.9	22.5 ± 0.6
Andriushchenko et al. [142]	ST	76.4 ± 2.8	35.3 ± 0.2	95.9 ± 0.6	43.1 ± 0.6	121.6 ± 0.4	51.4 ± 2.1
	RT (no add. data)	42 ± 0.5	16.2 ± 0.3	51.8 ± 1	19.5 ± 0.6	62.9 ± 0.8	22.8 ± 1.5
	MRT	78.5 ± 3.5	36 ± 1.1	101.9 ± 3.6	44.6 ± 1.1	122 ± 5.1	51.9 ± 2
	GNT-1.0	59.9 ± 2.3	39.2 ± 1.1	77.8 ± 3.9	48.9 ± 2.5	97.7 ± 7.5	58.3 ± 4.4
	GNT-0.1	71.1 ± 1.2	36.9 ± 0.2	91.2 ± 0.4	46.2 ± 0.4	115.2 ± 1.7	55.3 ± 0.6
Gowal et al. [153]	ST	63.8 ± 1.2	12.8 ± 0.1	84.7 ± 6.1	15.7 ± 0.5	114.4 ± 13.8	19 ± 0.8
	RT (add. data)	63.2 ± 1.3	22.8 ± 0.5	78.1 ± 1.6	28 ± 0.8	99.5 ± 2.7	32.4 ± 0.8
	MRT	63.1 ± 1.5	13.2 ± 0.3	91.7 ± 13.9	16.2 ± 0.2	123.3 ± 26.8	19.4 ± 0.2
	GNT-1.0	59.4 ± 1.3	16.5 ± 0.3	75.3 ± 5.7	20.4 ± 0.4	99.7 ± 12.6	24.8 ± 0.6
	GNT-0.1	60.5 ± 1	14.6 ± 0.4	78.7 ± 3.7	18 ± 0.4	108.5 ± 17.6	22.1 ± 1.2
Rade et al. [87]	ST	62.4 ± 2.1	13.1 ± 0.4	77.9 ± 4.7	16.2 ± 0.5	114.8 ± 11.5	19.9 ± 1.1
	RT (add. data)	61.3 ± 2.8	18.7 ± 0.5	78.7 ± 0.9	22.8 ± 0.7	97.4 ± 3	27.2 ± 1
	MRT	68.2 ± 1.8	13.9 ± 0.1	87.8 ± 10	17.4 ± 0.6	119.5 ± 20.3	21.2 ± 0
	GNT-1.0	60.4 ± 1.6	17.3 ± 0.2	73.5 ± 2.1	21.3 ± 0.1	89.6 ± 2.2	27.3 ± 0.9
	GNT-0.1	67.8 ± 2.4	14.2 ± 0.1	79.8 ± 2	17.5 ± 0.1	95.5 ± 6.2	20.7 ± 0.5

do not use any additional data beyond the original train set while the other two methods do so. We retrieve these models from *RobustBench* [206] and refer to them as *robustly trained* (RT) ones. In addition to these four state-of-the-art robustly trained models, we re-train every one according to the training methods introduced in Section 5.2.3 (page 119). Thus, for state-of-the-art robust model we obtain four additional non-robust models with the same architecture.

In Table 5.5 (page 122) and Table 5.6 (page 123) we report the standard and robust accuracies for all four original models. Again, we observe a tradeoff between small-norm and large-norm perturbations. However, the tradeoff between accuracy and robustness does only exist for models that do not use additional data. This result can be explained by the greater sample complexity of robust training (see Section 2.3, page 30 and Chapter 4, page 82).

In Table 5.7 (page 124) we display the PID and LID. When comparing state-of-the-art robust training introduced by the particular authors to standard training, we observe the same results as for adversarial training with FGSM for FASHION and CIFAR-10. PID is strongly negatively correlated with robustness, a finding that is, as before, also true for Gaussian-noise training. For LID, however, a distinction between robust training methods that use or do not use additional data needs to

be made. The robust training methods that do not use any additional data display either no changes in LID or report lower values, whereas those that use additional data report higher LID values. Those methods that use additional data also report increased generalisation performance. Thus, the findings of Ansuini et al. [214] and Brown et al. [217] do not hold for robustly trained models and LID is only negatively correlated with generalisation performance with standard training.

The Effect of Curvature on the Intrinsic Dimension Estimate

The estimate of the intrinsic dimension measure in Equation 3.14 (page 65) is influenced by the manifold's curvature. As it uses the k nearest neighbours of each sample, manifold regions with high curvature reduce the nearest neighbour distances and thus inflate the estimated intrinsic dimension. However, the manifold's *true* intrinsic dimension is independent of the curvature.

In a recent work Toosi et al. [224] showed that adversarially trained networks straighten the representations of natural videos. in other words, subsequent frames lie on subspaces that are close to being linear and interpolations between adjacent frames produce synthetic frames similar to the original ones. As the straightening of a manifold around a sample increases the distances to its nearest neighbours and thus reduces the estimated intrinsic dimension, the observed drop in PID for robust representations might not be due to an actual drop of the true intrinsic dimension but could simply be a measurement error due to a suboptimal estimation technique.

To rule out such a measurement error we re-compute the intrinsic dimension for smaller values of k for which geodesic and Euclidean distances are closer. In Table 5.7 (page 124) we observe that quantitatively the estimation of the intrinsic dimension changes, however, qualitatively the results remain the same across different values of k. Although reduced curvature contributes to a lower estimated intrinsic dimension, it is not the only explanation and a decrease in the true intrinsic dimension is occurring as well.

5.2.4 Conclusions

We chose a representative subset of recently proposed state-of-the-art robust training methods and find that despite significantly different approaches all methods

reduce the PID of hidden representation compared to standard trained benchmarks. This drop in PID occurs regardless of whether there has been a relative increases or decrease in generalisation performance. Thus, the observation by Brown et al. [217], that PID needs to be sufficiently high for good generalisation performance does only hold for regularised networks but not for robustly trained ones. This finding is further supported by the observation that Gaussian-noise training improves robustness and also reduces PID compared to the standard trained baseline. Thus, PID is negatively correlated with robustness regardless of the actual robust training method employed. Even drastically different approaches reduce the PID despite not being trained to do so. Further, the negative correlation between LID and generalisation performance also does not hold for robustly trained models. Generally, LID is negatively correlated with robustness even if the test accuracy drops in comparison to the standard trained model. Only when additional data is used and the generalisation performance exceeds the standard trained one's, LID increases. Again, this finding is at odds with those reported earlier by Ansuini et al. [214] and Brown et al. [217] for standard trained and regularised networks.

These results suggest that a lower PID is a fundamental property shared by robust training methods and is implicitly induced by those. It also explains the success of dimensionality reduction techniques in removing adversarial noise (see Section 2.4.3, page 37) and the success of regularising the rank of weight-[243] or activation-matrices [160].

In the following section we hypothesise that one mechanism that underlies this drop in PID is that neural networks represent semantically similar classes with similar features, i.e. features that are aligned in representation space.

5.3 Feature Sharing

The intrinsic dimension of a distribution refers to the number of variables required to describe its variations, thus it is a measure of the distribution's complexity. In the previous section we observed for toy- and real-world datasets and a variety of robust training methods, including training with Gaussian-noise data augmentation, a negative correlation between PID and robustness. Hence, robust representations

are characterised by fewer directions of variance and are thus less complex than those obtained by standard training.

In this section we investigate how this reduced complexity manifests itself when label-specific information is taken into account. We investigate the amount of information that representations of semantically similar classes share by analysing the alignment between label-specific representations.

In a related line of work other authors have investigated the similarity structure of representations and find links to their robustness and generalisation performance. Frosst et al. [220] regularise neural networks to spatially entangle representations in the hidden layers regardless of their class. They report increased accuracy and robustness possibly because models learn shared features across classes. However, the authors do not draw any connections to the intrinsic dimension of these entangled representations. It is clear that spatially entangling samples does not necessarily lead to a reduction in intrinsic dimension. In another work, Bai et al. [244] report that the linearised subnetworks of robust models are highly aligned along semantically similar classes. In contrast, we consider raw representations retrieved directly from the original networks, so with all non-learn components still in place.

In the following section we introduce the setup of the experiments including the proposed alignment measure. Then, in Section 5.3.2 (page 128) we describe the results.

5.3.1 Experimental Setup

The reduced intrinsic dimension observed for robustly trained models indicates that the distribution of representations over all classes is arranged along fewer directions of variance in representation space. Intuitively, this means that the model might re-use certain features for the classification of different classes.

One possible way to determine the alignment between representations is to compute the cosine similarity between their *right-singular vectors*. We write $X^{y=c}$ to denote those samples of dataset $X \in \mathbb{R}^{l \times \mathcal{E}}$ that are of class c. The top singular vector, so the one with the largest associated *singular value*, of $X^{y=c}$ describes the main direction of variance along which the samples $X^{y=c}$ are arranged. Using the notation $f_i(x)$ to describe the hidden representation of the i-th layer of x (see Section 2.1.2,

page 16), we denote the top-singular vector of $f_i(X^{y=c})$ as $\text{SVD}^{\text{top}}(f_i(X^{y=c})) \in \mathbb{R}^{1\times\mathcal{E}}$. Given two different classes c and c', the alignment of their top singular vectors can now be computed as,

$$S(f, X, c, c') = \text{cosine}\Big(\text{SVD}^{\text{top}}(f_i(X^{y=c})), \text{SVD}^{\text{top}}(f_i(X^{y=c'}))\Big) \in [-1, 1] \tag{5.10}$$

where $\text{cosine}(\cdot,\cdot)$ is the *cosine similarity*. In practice, we always choose the representations of the penultimate (pre-logit) layer as neural networks are known to build discriminative representations primarily in this layer. If S is close to 1, the class representations of c and c' share a similar main direction in representation space and thus use a similar set of features. As different neural networks might represent similar features differently in representation space, comparisons between two different models are not possible. However, given a certain model, alignment of different classes can be computed and compared to the alignment of other models.

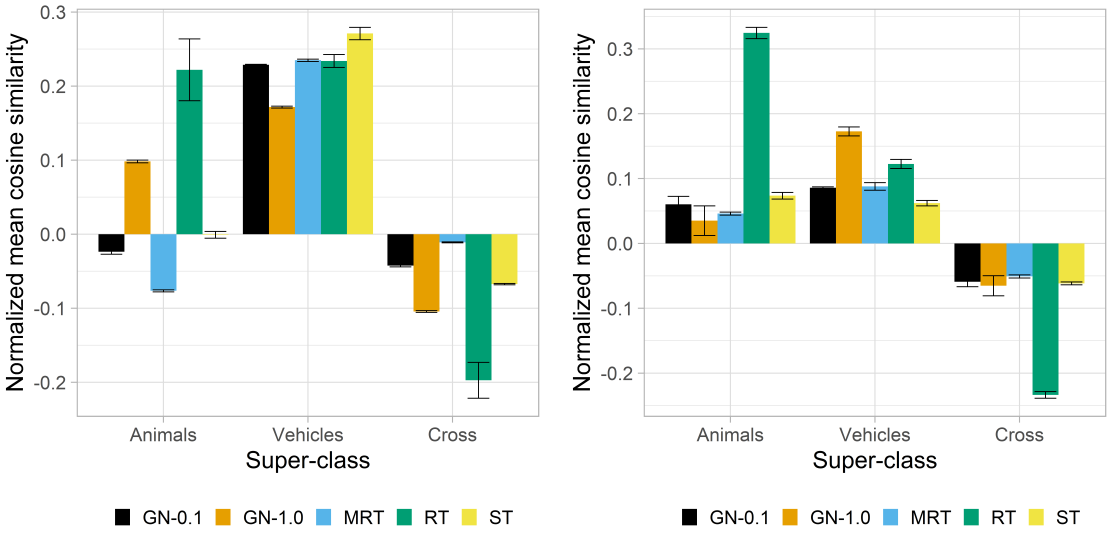
To analyse whether such an alignment occurs we utilise again the CIFAR-10 dataset, as its ten classes can be divided into the super-classes *vehicles* and *animals*. For each of these super-classes we compute the pair-wise similarities S for all associated subclasses, c, c', written as $\{S(f, X, c, c')\}_{c,c'\in super-class}$. The *normalised cosine similarity* is obtained by dividing by the mean cosine similarity across all classes. Formally, the normalised cosine similarity for the super-class *vehicles* is defined as

$$S_{norm}(f, X, y_{vehicle}) = \frac{\frac{1}{y_{vehicle}} \sum_{c,c'\in y_{vehicle}} (\{S(f, X, c, c')\})}{\frac{1}{y} \sum_{c,c'\in y} (\{S(f, X, c, c')\})} \tag{5.11}$$

and analogously for the super-class *animals*, where $y_{vehicle}$ ($y_{animals}$) is the number of sub-classes within the *vehicles* (*animals*) super-class and y the number of classes in the entire dataset. Further, we define the *cross-super-class* cosine similarity for all class-pairs where one is from the *animals* super-class and the other one from the *vehicles* super-class.

5.3.2 Results and Conclusions

In Figure 5.2 (page 129) and Figure 5.3 (page 130) we display the normalised cosine similarities (Equation 5.11, page 128) for the models evaluated in the previous



(a) Model: [241] (b) Model: [142]

Figure 5.2: Normalised cosine similarities between the top singular vectors per class. RT aligns semantically similar classes and dis-aligns semantically dissimilar classes to a much greater extent than ST.

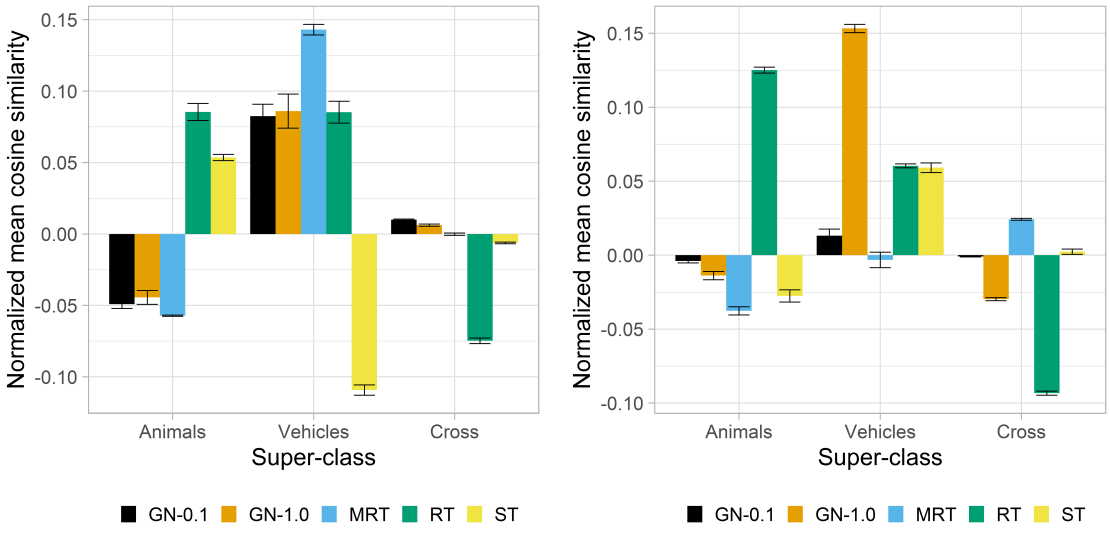
section.

When comparing the representations obtained by state-of-the-art robust training with those obtained by standard training, we see that representations within the super-classes are much more aligned in the robust representation space. Further, representations that belong to different super-classes are arranged dissimilarly. Interestingly, standard training tends to result in representations that are similar within one super-class but dissimilar in the other.

In the previous section we observed that training with Gaussian-noise results in an increased robustness and a reduction in PID. However, the feature sharing is not as pronounced as for the state-of-the-art robustly trained models which is possibly due to the significant robustness gap between robust training and Gaussian-noise data augmentation. Nevertheless, when comparing the differences between Gaussian-noise training for $p = 0.1$ and $p = 1.0$, a stronger feature alignment occurs for larger p.

5.3.3 Disentangling Super-classes

The above results show that robust training yields representations that are similar within super-classes and dissimilar across super-classes. This observation can be re-



(a) Model: [153] (b) Model: [87]

Figure 5.3: Normalised cosine similarities between the top singular vectors per class. RT aligns semantically similar classes and dis-aligns semantically dissimilar classes to a much greater extent than ST.

sult of either entanglement of semantically similar features or disentanglement of semantically dissimilar features. To investigate which one of these effects is the driving force behind increased robustness, we regularise representations to be *across-class disentangled* (ACD). We utilise the *soft nearest-neighbour loss* (SNNL) introduced by Frosst et al [220],

$$\mathcal{L}_{\text{SNNL}}(x, y; T) = -\frac{1}{b} \sum_{i \in 1..b} \log \left(\frac{\sum_{\substack{j \in 1..b \\ j \neq i \\ y_i = y_i}} e^{-\frac{||\bar{x}_i - \bar{x}_j||_2^2}{T}}}{\sum_{\substack{k \in 1..b \\ k \neq i}} e^{-\frac{||\bar{x}_i - \bar{x}_k||_2^2}{T}}} \right) \tag{5.12}$$

where $T \in \mathbb{R}_+$ is the *temperature*. We minimise the following loss function during training

$$\mathcal{L}_{\text{ACD}}(x, f_{L-1}(x), y; 1) = \mathcal{L}(x, y) + \alpha \mathcal{L}_{\text{SNNL}}(f_{L-1}(x), y; T) \tag{5.13}$$

with $\alpha \in \mathbb{R}$ and where $\mathcal{L}$ is the standard cross entropy loss. We only apply the SNNL to the pre-logit activations as those display the clearest separation by super-classes for the robust models. We utilise the CNN from Section 5.2.3 (page 119) again and train it with different values of α for the SNNL.

The results are presented in Table 5.8 (page 131) where one can observe that disentangling super-classes marginally *hurts* robustness against all perturbation types

Table 5.8: Accuracies of a six-layer CNN trained on CIFAR-10 with ACD-α (Equation 5.13). Disentangling representations of semantically dissimilar classes marginally impairs robustness.

	ST	ACD-0.5	ACD-1	ACD-2	ACD-5
TrainAcc	0.993	0.997	0.992	0.995	0.991
TestAcc	0.757	0.751	0.743	0.739	0.705
FGSM-4/255	0.191	0.112	0.095	0.094	0.069
FGSM-8/255	0.083	0.068	0.055	0.057	0.043
FGSM-16/255	0.057	0.059	0.047	0.056	0.051
PGD-0.01-5	0.001	0.322	0.314	0.298	0.285
Gaussian-noise	0.639	0.634	0.554	0.622	0.616
Shot-noise	0.606	0.597	0.527	0.589	0.573
Impulse-noise	0.673	0.675	0.609	0.644	0.633
Speckle-noise	0.692	0.678	0.620	0.664	0.659
Zoom-blur	0.793	0.779	0.775	0.771	0.734
Defocus-blur	0.900	0.891	0.885	0.882	0.858
Gaussian-blur	0.378	0.371	0.332	0.378	0.363
Glass-blur	0.917	0.910	0.901	0.902	0.886
Fog-blur	0.430	0.441	0.430	0.442	0.410
Brightness-blur	0.909	0.911	0.899	0.894	0.884
Contrast-blur	0.393	0.401	0.391	0.388	0.370

for different α. Thus, it appears it is not the separation of semantically dissimilar classes that underlies robust representations but the entanglement of semantically similar ones, which confirms the feature sharing hypothesis mentioned earlier.

5.4 Summary and Discussion

In this section we investigate the connection between the intrinsic dimension of hidden representations and the robustness of deep neural network classifiers. Whereas previous works show that regularising the intrinsic dimension, and thus reducing the intrinsic dimension, benefits robustness [160, 243], we establish the opposite connection by demonstrating that state-of-the-art robust training methods make the intrinsic dimension small, despite not being explicitly optimised to do so. This negative correlation between PID and robustness stands in contrast to previous works that consider weight-decay or dropout regularised networks, for which increases in PID are associated with better generalisation performance [217]. For robustly trained neural networks, the PID is generally smaller than for standard trained ones, even if the generalisation performance is improved due to the use of

additional data. Thus, the conclusions of previous works are not readily applicable to robustly trained neural networks. This conclusion also extends to the connection between LID and generalisation performance for robustly trained networks where we do not observe the previously reported negative correlation [214].

Following these experiments, we conduct an analysis to establish a possible reason for this observed drop in PID. We computed the cosine similarities between the representation's main direction of variance in representation space and find that robust networks tend to align semantically similar classes much closer than standard trained ones. Intuitively, this signals some sort of sharing of features across several semantically similar classes. We empirically show that it is not the separation of semantically dissimilar classes that improves robustness but the sharing of features across semantically similar ones. Although such a result has not yet been explicitly reported in the literature, previous works discuss related findings. Frosst et al. [220] show that entangling representation improves robustness. Further, Bai et al. [244] consider linear subnetworks in which all non-linear activation have been removed and demonstrate a similar clustering of semantically similar class representations. In this chapter we show a related finding and draw a connection to the intrinsic dimension of hidden representations.

Practical implications Robust training methods that optimise for different objective functions all display significant reductions in the intrinsic dimensionality of their hidden representations. Thus this observation offers a useful path for future research. As shown in Section 2.4 (page 35), a wide variety of robust training methods have been proposed and the reduction in intrinsic dimension appears to be mechanisms shared by most of them. Therefore, finding ways to estimate the intrinsic dimension and regularise it during training might be a path towards closing the robustness gap.

Chapter 6

Conclusions

In this thesis we studied the robustness of artificial neural networks in supervised visual object recognition from a geometric perspective. As the fundamental operation carried out by biological and artificial neural networks is the progressive untangling of complex label-specific image distributions it offers an interesting perspective on the robustness topic. In Chapter 2 (page 9) we first introduced the related literature dealing with neural networks' lack of robustness in breadth. Following that, we gave an exhaustive overview of the literature that studied the influence of geometric properties of data distributions on neural networks. In the intersection between these two bodies of literature we identified three main open questions and proposed to answer to them.

In Chapter 3 (page 56) we showed that the entanglement of distributions is the leading contributor to the sample complexity of deep neural networks and complement prior work by showing that the intrinsic dimension's influence on the sample complexity depends on the given level of entanglement. For low levels of entanglement artificial neural networks behave similar to support vector machines in the sense that their sample complexity is also not affected by the intrinsic dimensionality. For higher levels of entanglement though, the intrinsic dimensionality's influence on the sample complexity increases.

Then, in Chapter 4 (page 82) we studied the geometric complexity of decision boundaries from a novel dataset-specific point of view and empirically confirmed the previously made hypothesis that robust neural networks learn geometrically more complex decision boundaries. In combination with our first result we were thus

able to partially explain the increased sample complexity of robust training by the increased entanglement of robust decision boundaries. We further introduced an upper bound on the perturbation magnitude in image space over which provably a geometrically more complex decision boundary is required. This bound improves over previous bounds which rely on the minimum nearest neighbour distance and we show that those significantly overestimate the actual robust radius of a commonly used image datasets.

Finally in Chapter 5 (page 112) we investigated different state-of-the-art robust training methods and the underlying mechanisms they use to learn robust representations. We found that a key similarity between otherwise very different robust training approaches is the lower intrinsic dimension of their hidden representations compared to standard training. We demonstrated that one reason for the reduced dimensionality is that semantically similar classes share features.

6.1 Limitations and Future Work

Convolutional neural networks are still the most commonly used architecture in computer vision, however, recently Transformers have gained attention in this area as well. As several studies have compared the robustness of convolutional neural networks to Transformers (see Section 2.2.4, page 27), expanding our experiments in Chapter 3 (page 56) and Chapter 5 (page 112) to these models is a straightforward idea. Further, as the learning mechanisms of Transferors are largely unexplored, a comparative study between those and CNNs is a possible direction for future work, too.

In this thesis, we chose several commonly image benchmarks that have small sample sizes. Expanding the analysis in all chapters to larger datasets, such as CIFAR-100 and ImageNet is also a possible direction for future work. However, we do not expect that such as expansion yields significantly different results, because the chosen datasets can be viewed as lower bounding the complexity and if the results hold true for the used datasets they should also hold true for even more complex ones.

As shown by Pope et al. [213] and D'Amario et al. [213], differentiating between

informative and uninformative extrinsic dimensions can yield fundamentally different results when studying the sample complexity. A similar distinction can also be made for the intrinsic dimensions.

Bibliography

[1] J. J. DiCarlo and D. D. Cox, *Untangling invariant object recognition,* Trends in cognitive sciences (2007).

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning,* MIT press, 2016.

[3] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, *3d object representations for fine-grained categorization,* Proceedings of the IEEE international conference on computer vision workshops (2013).

[4] M. Mishkin, L. G. Ungerleider, and K. A. Macko, *Object vision and spatial vision: two cortical pathways,* Trends in neurosciences (1983).

[5] L. G. Ungerleider and J. V. Haxby, *'What' and 'where' in the human brain,* Current opinion in neurobiology (1994).

[6] D. H. Hubel and T. N. Wiesel, *Receptive fields and functional architecture of monkey striate cortex,* The Journal of physiology (1968).

[7] N. Kriegeskorte and R. A. Kievit, *Representational geometry: integrating cognition, computation, and the brain,* Trends in cognitive sciences,(2013).

[8] J. J. DiCarlo, D. Zoccolan, and N. C. Rust, *How does the brain solve visual object recognition?,* Neuron (2012).

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *Imagenet classification with deep convolutional neural networks,* Twenty-sixth Annual Conference on Neural Information Processing Systems (2012).

[10] A. Radford, L. Metz, and S. Chintala, *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks,* 4th International Conference on Learning Representations (2016).

[11] R. Geirhos, C. Medina Temme, J. Rauber, H. Schütt, M. Bethge, and F. Wichmann, *Generalisation in humans and deep neural networks,* Thirty-second Annual Conference on Neural Information Processing Systems (2018).

[12] H. Narayanan and P. Niyogi, *On the Sample Complexity of Learning Smooth Cuts on a Manifold,* COLT (2009).

[13] H. Narayanan and S. Mitter, *Sample complexity of testing the manifold hypothesis,* Twenty-Fourth Annual Conference on Neural Information Processing System (2010).

[14] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, *Understanding deep learning (still) requires rethinking generalization,* Communications of the ACM (2021).

[15] M. Belkin, D. Hsu, S. Ma, and S. Mandal, *Reconciling modern machine-learning practice and the classical bias–variance trade-off,* Proceedings of the National Academy of Sciences (2019).

[16] P. Pope, C. Zhu, A. Abdelkader, M. Goldblum, and T. Goldstein, *The Intrinsic Dimension of Images and Its Impact on Learning,* International Conference on Learning Representations (2020).

[17] Y.-Y. Yang, C. Rashtchian, H. Zhang, R. R. Salakhutdinov, and K. Chaudhuri, *A closer look at accuracy vs. robustness,* Thirty-fourth Annual Conference on Neural Information Processing Systems (2020).

[18] N. Carlini, *A Complete List of All (arXiv) Adversarial Example Papers,* Accessed: 20.07.2023.

[19] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, *Handwritten digit recognition with a back-propagation network,* Advances in neural information processing systems (1990).

[20] B. E. Boser, I. M. Guyon, and V. N. Vapnik, *A training algorithm for optimal margin classifiers,* Proceedings of the fifth annual workshop on Computational learning theory (1992).

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

[22] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, *Reading Digits in Natural Images with Unsupervised Feature Learning,* NIPS Workshop on Deep Learning and Unsupervised Feature Learning (2011).

[23] A. Krizhevsky, G. Hinton, and others, *Learning multiple layers of features from tiny images ,* University of Toronto (2009).

[24] Y. Bengio, A. Courville, and P. Vincent, *Representation learning: A review and new perspectives,* IEEE transactions on pattern analysis and machine intelligence (2013).

[25] C. Stephenson, J. Feather, S. Padhy, O. Elibol, H. Tang, J. McDermott, and S. Chung, *Untangling in invariant speech recognition,* Thirty-third Annual Conference on Neural Information Processing Systems (2019).

[26] F. Rosenblatt, *The perceptron: a probabilistic model for information storage and organization in the brain,* Psychological review (1958).

[27] K. Fukushima, *Neural network model for a mechanism of pattern recognition unaffected by shift in position-neocognitron,* IEICE Technical Report, A (1979).

[28] K. Fukushima and S. Miyake, *Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition,* Competition and cooperation in neural nets (1982).

[29] K. Fukushima, *Artificial vision by multi-layered neural networks: Neocognitron and its advances,* Neural networks (2013).

[30] J. Schmidhuber, *Deep learning in neural networks: An overview,* Neural networks (2015).

[31] K. Fukushima, *Visual feature extraction by a multilayered network of analog threshold elements,* IEEE Transactions on Systems Science and Cybernetics (1969).

[32] X. Glorot, A. Bordes, and Y. Bengio, *Deep sparse rectifier neural networks,* Proceedings of the fourteenth international conference on artificial intelligence and statistics (2011).

[33] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, *Intriguing properties of neural networks,* 2nd International Conference on Learning Representations (2014).

[34] J.-H. Jacobsen, J. Behrmann, R. Zemel, and M. Bethge, *Excessive Invariance Causes Adversarial Vulnerability,* International Conference on Learning Representations (2018).

[35] A. Nguyen, J. Yosinski, and J. Clune, *Deep neural networks are easily fooled: High confidence predictions for unrecognizable images,* Proceedings of the IEEE conference on computer vision and pattern recognition (2015).

[36] K. Lenc and A. Vedaldi, *Understanding image representations by measuring their equivariance and equivalence,* Proceedings of the IEEE conference on computer vision and pattern recognition (2015).

[37] S. Soatto and A. Chiuso, *Modeling Visual Representations: Defining Properties and Deep Approximations,* 4th International Conference on Learning Representations (2016).

[38] A. Fawzi and P. Frossard, *Manitest: Are classifiers really invariant?,* British Machine Vision Conference (BMVC) (2015).

[39] C. Kanbak, S.-M. Moosavi-Dezfooli, and P. Frossard, *Geometric robustness of deep networks: analysis and improvement,* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2018).

[40] M. A. Alcorn, Q. Li, Z. Gong, C. Wang, L. Mai, W.-S. Ku, and A. Nguyen, *Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects,* Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (2019).

[41] D. Hendrycks and T. Dietterich, *Benchmarking Neural Network Robustness to Common Corruptions and Perturbations,* International Conference on Learning Representations (2018).

[42] Z. Zhu, L. Xie, and A. Yuille, *Object recognition with and without objects,* Proceedings of the 26th International Joint Conference on Artificial Intelligence (2017).

[43] S. Beery, G. Van Horn, and P. Perona, *Recognition in terra incognita,* Proceedings of the European conference on computer vision (ECCV) (2018).

[44] B. Carter, S. Jain, J. W. Mueller, and D. Gifford, *Overinterpretation reveals image classification model pathologies,* Thirty-fifth Annual Conference on Neural Information Processing Systems (2021).

[45] K. Xiao, L. Engstrom, A. Ilyas, and A. Madry, *Noise or Signal: The Role of Image Backgrounds in Object Recognition,* International Conference on Learning Representations (2021).

[46] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, *Do imagenet classifiers generalize to imagenet?,* International Conference on Machine Learning (2019).

[47] D. Hendrycks, K. Zhao, S. Basart, J. Steinhardt, and D. Song, *Natural adversarial examples,* Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2021).

[48] V. Shankar, A. Dave, R. Roelofs, D. Ramanan, B. Recht, and L. Schmidt, *Do image classifiers generalize across time?,* Proceedings of the IEEE/CVF International Conference on Computer Vision (2021).

[49] D. Rolnick, A. Veit, S. Belongie, and N. Shavit, *Deep learning is robust to massive label noise,* arXiv preprint arXiv:1705.10694 (2017).

[50] P. W. Koh, S. Sagawa, H. Marklund, S. M. Xie, M. Zhang, A. Balsubramani, W. Hu, M. Yasunaga, R. L. Phillips, I. Gao, and others, *Wilds: A benchmark of in-the-wild distribution shifts,* International Conference on Machine Learning (2021).

[51] D. Stutz, M. Hein, and B. Schiele, *Disentangling adversarial robustness and generalization,* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2019).

[52] A. Serban, E. Poll, and J. Visser, *Adversarial examples on object recognition: A comprehensive survey,* ACM Computing Surveys (CSUR) (2020).

[53] J. Su, D. V. Vargas, and K. Sakurai, *One pixel attack for fooling deep neural networks,* IEEE Transactions on Evolutionary Computation (2019).

[54] I. J. Goodfellow, J. Shlens, and C. Szegedy, *Explaining and Harnessing Adversarial Examples,* International Conference on Learning Representations (2015).

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

[56] A. Athalye, N. Carlini, and D. Wagner, *Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples,* International conference on machine learning (2018).

[57] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, *Deepfool: a simple and accurate method to fool deep neural networks,* Proceedings of the IEEE conference on computer vision and pattern recognition (2016).

[58] M. Alzantot, Y. Sharma, S. Chakraborty, H. Zhang, C.-J. Hsieh, and M. B. Srivastava, *Genattack: Practical black-box attacks with gradient-free optimization,* Proceedings of the genetic and evolutionary computation conference (2019).

[59] S. Luke, *Essentials of Metaheuristics ,* Springer (2013).

[60] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard, and S. Soatto, *Analysis of universal adversarial perturbations,* arXiv preprint arXiv:1705.09554 (2017).

[61] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, *The space of transferable adversarial examples,* arXiv preprint arXiv:1704.03453 (2017).

[62] S. Gu and L. Rigazio, *Towards Deep Neural Network Architectures Robust to Adversarial Examples,* 3rd International Conference on Learning Representations (2015).

[63] X. Ma, B. Li, Y. Wang, S. M. Erfani, S. Wijewickrema, G. Schoenebeck, D. Song, M. E. Houle, and J. Bailey, *Characterizing Adversarial Subspaces Using Local Intrinsic Dimensionality,* International Conference on Learning Representations (2018).

[64] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, *Adversarial examples are not bugs, they are features,* Thirty-third Annual Conference on Neural Information Processing Systems (2019).

[65] J. Jacobsen, J. Behrmann, N. Carlini, F. Tramèr, and N. Papernot, *Exploiting Excessive Invariance caused by Norm-Bounded Adversarial Robustness,* CoRR (2019).

[66] F. Tramèr, J. Behrmann, N. Carlini, N. Papernot, and J.-H. Jacobsen, *Fundamental tradeoffs between invariance and sensitivity to adversarial perturbations,* International Conference on Machine Learning (2020).

[67] I. Goodfellow, H. Lee, Q. Le, A. Saxe, and A. Ng, *Measuring invariances in deep networks,* Twenty-Third Annual Conference on Neural Information Processing Systems (2009).

[68] F. A. Soto and E. A. Wasserman, *Visual object categorization in birds and primates: Integrating behavioral, neurobiological, and computational evidence within a "general process" framework,* Cognitive, Affective, & Behavioral Neuroscience (2012).

[69] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, *ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness,* International Conference on Learning Representations (2018).

[70] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, *ImageNet: A Large-Scale Hierarchical Image Database,* CVPR (2009).

[71] K. Hermann, T. Chen, and S. Kornblith, *The origins and prevalence of texture bias in convolutional neural networks,* Thirty-fourth Annual Conference on Neural Information Processing Systems (2020).

[72] M. J. Choi, A. Torralba, and A. S. Willsky, *Context models and out-of-context objects,* Pattern Recognition Letters (2012).

[73] G. Jacob, R. Pramod, H. Katti, and S. Arun, *Qualitative similarities and differences in visual object representations between brains and deep networks,* Nature communications (2021).

[74] D. Whitney and D. M. Levi, *Visual crowding: A fundamental limit on conscious perception and object recognition,* Trends in cognitive sciences (2011).

[75] D. M. Levi, *Crowding—An essential bottleneck for object recognition: A mini-review,* Vision research (2008).

[76] A. Volokitin, G. Roig, and T. A. Poggio, *Do deep neural networks suffer from crowding?,* The Thirty-first Annual Conference on Neural Information Processing Systems (2017).

[77] K. Kireev, M. Andriushchenko, and N. Flammarion, *On the effectiveness of adversarial training against common corruptions,* Uncertainty in Artificial Intelligence (2022).

[78] R. Taori, A. Dave, V. Shankar, N. Carlini, B. Recht, and L. Schmidt, *Measuring robustness to natural distribution shifts in image classification,* Thirty-fourth Annual Conference on Neural Information Processing Systems (2020).

[79] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo, and others, *The many faces of robustness: A critical analysis of out-of-distribution generalization,* Proceedings of the IEEE/CVF International Conference on Computer Vision (2021).

[80] R. Geirhos, K. Narayanappa, B. Mitzkus, T. Thieringer, M. Bethge, F. A. Wichmann, and W. Brendel, *Partial success in closing the gap between human and machine vision,* Thirty-fifth Annual Conference on Neural Information Processing Systems (2021).

[81] L. Engstrom, A. Ilyas, S. Santurkar, D. Tsipras, B. Tran, and A. Madry, *Adversarial robustness as a prior for learned representations,* arXiv preprint arXiv:1906.00945 (2019).

[82] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry, *Robustness May Be at Odds with Accuracy,* International Conference on Learning Representations (2019).

[83] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. Jordan, *Theoretically principled trade-off between robustness and accuracy,* International conference on machine learning (2019).

[84] A. Raghunathan, S. M. Xie, F. Yang, J. Duchi, and P. Liang, *Adversarial Training Can Hurt Generalization,* ICML 2019 Workshop on Identifying and Understanding Deep Learning Phenomena (2019).

[85] A. Raghunathan, S. M. Xie, F. Yang, J. Duchi, and P. Liang, *Understanding and Mitigating the Tradeoff between Robustness and Accuracy,* International Conference on Machine Learning (2020).

[86] A. Sanyal, P. K. Dokania, V. Kanade, and P. Torr, *How Benign is Benign Overfitting?,* International Conference on Learning Representations (2020).

[87] R. Rade and S.-M. Moosavi-Dezfooli, *Helper-based adversarial training: Reducing excessive margin to achieve a better accuracy vs. robustness trade-off,* ICML 2021 Workshop on Adversarial Machine Learning (2021).

[88] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, *Image segmentation using deep learning: A survey,* IEEE transactions on pattern analysis and machine intelligence (2021).

[89] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, *Deep learning for generic object detection: A survey,* International journal of computer vision (2020).

[90] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille, *Adversarial examples for semantic segmentation and object detection,* Proceedings of the IEEE International Conference on Computer Vision (2017).

[91] G. F. Elsayed, I. Goodfellow, and J. Sohl-Dickstein, *Adversarial Reprogramming of Neural Networks,* International Conference on Learning Representations (2019).

[92] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, *Nerf: Representing scenes as neural radiance fields for view synthesis,* Communications of the ACM (2021).

[93] R. Yonetani, T. Taniai, M. Barekatain, M. Nishimura, and A. Kanezaki, *Path planning using neural a* search,* International conference on machine learning (2021).

[94] S. H. Huang, N. Papernot, I. J. Goodfellow, Y. Duan, and P. Abbeel, *Adversarial Attacks on Neural Network Policies,* 5th International Conference on Learning Representations (2017).

[95] P. R. Montague, *Reinforcement learning: an introduction, by Sutton, RS and Barto, AG,* Trends in cognitive sciences (1999).

[96] Y. Qin, N. Carlini, G. Cottrell, I. Goodfellow, and C. Raffel, *Imperceptible, robust, and targeted adversarial examples for automatic speech recognition,* International conference on machine learning (2019).

[97] W. E. Zhang, Q. Z. Sheng, A. Alhazmi, and C. Li, *Adversarial attacks on deep-learning models in natural language processing: A survey,* ACM Transactions on Intelligent Systems and Technology (TIST) (2020).

[98] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, *Attention is all you need,* The Thirty-first Annual Conference on Neural Information Processing Systems (2017).

[99] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale,* International Conference on Learning Representations (2021).

[100] D. Hendrycks, X. Liu, E. Wallace, A. Dziedzic, R. Krishnan, and D. Song, *Pretrained Transformers Improve Out-of-Distribution Robustness,* Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (2020).

[101] Y. Fu, S. Zhang, S. Wu, C. Wan, and Y. Lin, *Patch-Fool: Are Vision Transformers Always Robust Against Adversarial Perturbations?,* International Conference on Learning Representations (2021).

[102] Y. Bai, J. Mei, A. L. Yuille, and C. Xie, *Are Transformers more robust than CNNs?,* Thirty-fifth Annual Conference on Neural Information Processing Systems (2021).

[103] N. Papernot, P. McDaniel, A. Swami, and R. Harang, *Crafting adversarial input sequences for recurrent neural networks,* MILCOM 2016-2016 IEEE Military Communications Conference (2016).

[104] B. Sengupta and K. J. Friston, *How Robust are Deep Neural Networks?,* arXiv preprint arXiv:1804.11313 (2018).

[105] Y. Shi, I. Daunhawer, J. E. Vogt, P. Torr, and A. Sanyal, *How robust is unsupervised representation learning to distribution shift?,* The Eleventh International Conference on Learning Representations (2022).

[106] Y. Wang, S. Jha, and K. Chaudhuri, *Analyzing the robustness of nearest neighbors to adversarial examples,* International Conference on Machine Learning (2018).

[107] N. Papernot, P. D. McDaniel, and I. J. Goodfellow, *Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples,* CoRR (2016).

[108] L. Breiman, *Classification and regression trees,* Routledge, 2017.

[109] A. Fawzi, H. Fawzi, and O. Fawzi, *Adversarial vulnerability for any classifier,* The Thirty-second Annual Conference on Neural Information Processing Systems (2018).

[110] C. Guo, M. Lee, G. Leclerc, J. Dapello, Y. Rao, A. Madry, and J. Dicarlo, *Adversarially trained neural representations are already as robust as biological neural representations,* International Conference on Machine Learning (2022).

[111] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, *Autoencoding beyond pixels using a learned similarity metric,* International conference on machine learning (2016).

[112] M. Rosca, B. Lakshminarayanan, D. Warde-Farley, and S. Mohamed, *Variational Approaches for Auto-Encoding Generative Adversarial Networks,* CoRR (2017).

[113] D. P. Kingma and M. Welling, *Auto-Encoding Variational Bayes,* 2nd International Conference on Learning Representations (2014).

[114] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial nets,* The Twenty-eighth Annual Conference on Neural Information Processing Systems (2014).

[115] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, *EMNIST: Extending MNIST to handwritten letters,* 2017 international joint conference on neural networks (IJCNN) (2017).

[116] S. Singla and S. Feizi, *Salient ImageNet: How to discover spurious features in Deep Learning?,* International Conference on Learning Representations (2021).

[117] B. Joe, S. J. Hwang, and I. Shin, *Learning to Disentangle Robust and Vulnerable Features for Adversarial Detection,* CoRR (2019).

[118] P. Nakkiran, *Adversarial Robustness May Be at Odds With Simplicity,* CoRR (2019).

[119] G. Valle-Perez, C. Q. Camargo, and A. A. Louis, *Deep learning generalizes because the parameter-function map is biased towards simple functions,* International Conference on Learning Representations (2018).

[120] H. Shah, K. Tamuly, A. Raghunathan, P. Jain, and P. Netrapalli, *The pitfalls of simplicity bias in neural networks,* Thirty-fourth Annual Conference on Neural Information Processing Systems (2020).

[121] K. Hermann and A. Lampinen, *What shapes feature representations? exploring datasets, architectures, and training,* Thirty-fourth Annual Conference on Neural Information Processing Systems (2020).

[122] J. Jo and Y. Bengio, *Measuring the tendency of CNNs to Learn Surface Statistical Regularities,* CoRR (2017).

[123] D. Yin, R. Gontijo Lopes, J. Shlens, E. D. Cubuk, and J. Gilmer, *A fourier perspective on model robustness in computer vision,* The Thirty-third Annual Conference on Neural Information Processing Systems (2019).

[124] S. Singla, B. Nushi, S. Shah, E. Kamar, and E. Horvitz, *Understanding failures of deep networks via robust feature extraction,* Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2021).

[125] V. Vapnik, *Principles of risk minimization for learning theory,* Advances in neural information processing systems (1991).

[126] L. Schmidt, S. Santurkar, D. Tsipras, K. Talwar, and A. Madry, *Adversarially robust generalization requires more data,* The Thirty-second Annual Conference on Neural Information Processing Systems (2018).

[127] A. N. Bhagoji, D. Cullina, and P. Mittal, *Lower Bounds on Adversarial Robustness from Optimal Transport,* Thirty-thrid Annual Conference on Neural Information Processing Systems (2019).

[128] E. Dobriban, H. Hassani, D. Hong, and A. Robey, *Provable Tradeoffs in Adversarially Robust Classification,* IEEE Trans. Inf. Theory (2023).

[129] C. Dan, Y. Wei, and P. Ravikumar, *Sharp statistical guaratees for adversarially robust gaussian classification,* International Conference on Machine Learning (2020).

[130] R. Bhattacharjee, S. Jha, and K. Chaudhuri, *Sample Complexity of Robust Linear Classification on Separated Data,* International Conference on Machine Learning (2021).

[131] D. Yin, R. Kannan, and P. Bartlett, *Rademacher complexity for adversarially robust generalization,* International conference on machine learning (2019).

[132] J. Khim and P. Loh, *Adversarial Risk Bounds for Binary Classification via Function Transformation,* CoRR (2018).

[133] I. Attias, A. Kontorovich, and Y. Mansour, *Improved generalization bounds for robust learning,* Algorithmic Learning Theory (2019).

[134] O. Montasser, S. Hanneke, and N. Srebro, *Vc classes are adversarially robustly learnable, but only improperly,* Conference on Learning Theory (2019).

[135] H. Ashtiani, V. Pathak, and R. Urner, *Black-box certification and learning under adversarial perturbations,* International Conference on Machine Learning (2020).

[136] Z. Yao, A. Gholami, Q. Lei, K. Keutzer, and M. W. Mahoney, *Hessian-based analysis of large batch training and robustness to adversaries,* The Thirty-second Annual Conference on Neural Information Processing Systems (2018).

[137] A. Rozsa, M. Gunther, and T. E. Boult, *Towards Robust Deep Neural Networks with BANG,* 2018 IEEE Winter Conference on Applications of Computer Vision (WACV) (2018).

[138] T. Tanay and L. Griffin, *A boundary tilting perspective on the phenomenon of adversarial examples,* arXiv preprint arXiv:1608.07690 (2016).

[139] R. Izmailov, S. Sugrim, R. Chadha, P. McDaniel, and A. Swami, *Enablers of adversarial attacks in machine learning,* MILCOM 2018-2018 IEEE Military Communications Conference (MILCOM) (2018).

[140] J. Gilmer, L. Metz, F. Faghri, S. S. Schoenholz, M. Raghu, M. Wattenberg, and I. Goodfellow, *The Relationship Between High-Dimensional Geometry and Adversarial Examples,* arXiv preprint arXiv:1801.02774 (2018).

[141] E. Wong, L. Rice, and J. Z. Kolter, *Fast is better than free: Revisiting adversarial training,* International Conference on Learning Representations (2019).

[142] M. Andriushchenko and N. Flammarion, *Understanding and improving fast adversarial training,* Thirty-fourth Annual Conference on Neural Information Processing Systems (2020).

[143] A. Modas, R. Rade, G. Ortiz-Jiménez, S. Moosavi-Dezfooli, and P. Frossard, *PRIME: A Few Primitives Can Boost Robustness to Common Corruptions,* Computer Vision - ECCV 2022 - 17th European Conference (2022).

[144] S. Gowal, C. Qin, J. Uesato, T. A. Mann, and P. Kohli, *Uncovering the Limits of Adversarial Training against Norm-Bounded Adversarial Examples,* CoRR (2020).

[145] L. Rice, E. Wong, and Z. Kolter, *Overfitting in adversarially robust deep learning,* International Conference on Machine Learning (2020).

[146] D. Wu, S.-T. Xia, and Y. Wang, *Adversarial weight perturbation helps robust generalization,* Thirty-fourth Annual Conference on Neural Information Processing Systems (2020).

[147] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, *AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty,* International Conference on Learning Representations (2019).

[148] S.-A. Rebuffi, S. Gowal, D. A. Calian, F. Stimberg, O. Wiles, and T. A. Mann, *Data Augmentation Can Improve Robustness,* Thirty-fifth Annual Conference on Neural Information Processing Systems (2021).

[149] J.-B. Alayrac, J. Uesato, P.-S. Huang, A. Fawzi, R. Stanforth, and P. Kohli, *Are Labels Required for Improving Adversarial Robustness?,* The Thirty-thrid Annual Conference on Neural Information Processing Systems (2019).

[150] Y. Carmon, A. Raghunathan, L. Schmidt, P. Liang, and J. C. Duchi, *Unlabeled data improves adversarial robustness,* Proceedings of the 33rd International Conference on Neural Information Processing Systems (2019).

[151] A. Najafi, S.-i. Maeda, M. Koyama, and T. Miyato, *Robustness to adversarial perturbations in learning from incomplete data,* The Thirty-second Annual Conference on Neural Information Processing Systems (2019).

[152] R. Zhai, T. Cai, D. He, C. Dan, K. He, J. E. Hopcroft, and L. Wang, *Adversarially Robust Generalization Just Requires More Unlabeled Data,* CoRR (2019).

[153] S. Gowal, S.-A. Rebuffi, O. Wiles, F. Stimberg, D. A. Calian, and T. A. Mann, *Improving robustness using generated data,* Thirty-fifth Annual Conference on Neural Information Processing Systems (2021).

[154] S. Rifai, Y. N. Dauphin, P. Vincent, Y. Bengio, and X. Muller, *The Manifold Tangent Classifier,* Advances in Neural Information Processing Systems (2011).

[155] H. Xie, J. Li, and H. Xue, *A survey of dimensionality reduction techniques based on random projection,* CoRR (2017).

[156] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, *Extracting and composing robust features with denoising autoencoders,* Proceedings of the 25th international conference on Machine learning (2008).

[157] V. Srinivasan, A. Marbán, K. Müller, W. Samek, and S. Nakajima, *Counterstrike: Defending Deep Learning Architectures Against Adversarial Samples by Langevin Dynamics with Supervised Denoising Autoencoder,* CoRR (2018).

[158] R. Sahay, R. Mahfuz, and A. E. Gamal, *Combatting Adversarial Attacks through Denoising and Dimensionality Reduction: A Cascaded Autoencoder Approach ,* 2019 53rd Annual Conference on Information Sciences and Systems (CISS)(2019).

[159] G. Zizzo, C. Hankin, S. Maffeis, and K. Jones, *Deep Latent Defence,* CoRR (2019).

[160] A. Sanyal, V. Kanade, P. H. Torr, and P. K. Dokania, *Robustness via deep low-rank representations,* arXiv preprint arXiv:1804.07090 (2018).

[161] F. Tramer, *Detecting adversarial examples is (nearly) as hard as classifying them,* International Conference on Machine Learning (2022).

[162] F. Ahmed, Y. Bengio, H. van Seijen, and A. Courville, *Systematic generalisation with group invariant predictions,* International Conference on Learning Representations (2020).

[163] P. Izmailov, P. Kirichenko, N. Gruver, and A. G. Wilson, *On feature learning in the presence of spurious correlations,* The Thirty-sixth Annual Conference on Neural Information Processing Systems (2022).

[164] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, *The caltech-ucsd birds-200-2011 dataset,* California Institute of Technology (2011).

[165] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, *Places: A 10 million image database for scene recognition,* IEEE transactions on pattern analysis and machine intelligence (2017).

[166] S. Sagawa*, P. W. Koh*, T. B. Hashimoto, and P. Liang, *Distributionally Robust Neural Networks,* International Conference on Learning Representations (2020).

[167] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim, *Sanity checks for saliency maps,* The Thirty-second Annual Conference on Neural Information Processing Systems (2018).

[168] C. Etmann, S. Lunz, P. Maass, and C. Schönlieb, *On the Connection Between Adversarial Robustness and Saliency Map Interpretability,* International conference on machine learning (2019).

[169] A. Ross and F. Doshi-Velez, *Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients,* Proceedings of the AAAI Conference on Artificial Intelligence (2018).

[170] H. Drucker and Y. Le Cun, *Improving generalization performance using double backpropagation,* IEEE transactions on neural networks (1992).

[171] A. Chan, Y. Tay, Y. S. Ong, and J. Fu, *Jacobian Adversarially Regularized Networks for Robustness,* International Conference on Learning Representations (2020).

[172] A. Chan, Y. Tay, and Y.-S. Ong, *What it thinks is important is important: Robustness transfers through input gradients,* Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2020).

[173] B. Simpson, F. Dutil, Y. Bengio, and J. P. Cohen, *GradMask: Reduce Overfitting by Regularizing Saliency,* International Conference on Medical Imaging with Deep Learning–Extended Abstract Track (2019).

[174] K. Du, S. Chang, H. Wen, and H. Zhang, *Fighting Adversarial Images With Interpretable Gradients,* ACM Turing Award Celebration Conference-China (ACM TURC 2021) (2021).

[175] Y. Tsuzuku, I. Sato, and M. Sugiyama, *Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks,* The Thirty-second Annual Conference on Neural Information Processing Systems (2018).

[176] A. Virmaux and K. Scaman, *Lipschitz regularity of deep neural networks: analysis and efficient estimation,* Advances in Neural Information Processing Systems (2018).

[177] C. Anil, J. Lucas, and R. Grosse, *Sorting out Lipschitz function approximation,* International Conference on Machine Learning (2019).

[178] M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier, *Parseval networks: Improving robustness to adversarial examples,* International conference on machine learning (2017).

[179] J. Sokolić, R. Giryes, G. Sapiro, and M. R. Rodrigues, *Robust large margin deep neural networks,* IEEE Transactions on Signal Processing (2017).

[180] G. Elsayed, D. Krishnan, H. Mobahi, K. Regan, and S. Bengio, *Large margin deep networks for classification,* The Thirty-second Annual Conference on Neural Information Processing Systems (2018).

[181] A. Matyasko and L.-P. Chau, *Margin maximization for robust classification using deep learning,* 2017 International Joint Conference on Neural Networks (IJCNN) (2017).

[182] Z. Yan, Y. Guo, and C. Zhang, *Adversarial margin maximization networks,* IEEE transactions on pattern analysis and machine intelligence (2019).

[183] G. W. Ding, Y. Sharma, K. Y. C. Lui, and R. Huang, *MMA Training: Direct Input Space Margin Maximization through Adversarial Training,* International Conference on Learning Representations (2019).

[184] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, *Quantization and training of neural networks for efficient integer-arithmetic-only inference,* Proceedings of the IEEE conference on computer vision and pattern recognition (2018).

[185] H. Pouransari, Z. Tu, and O. Tuzel, *Least squares binary quantization of neural networks,* Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (2020).

[186] T. Liang, J. Glossner, L. Wang, S. Shi, and X. Zhang, *Pruning and quantization for deep neural network acceleration: A survey,* Neurocomputing (2021).

[187] J. Diffenderfer, B. Bartoldson, S. Chaganti, J. Zhang, and B. Kailkhura, *A winning hand: Compressing deep networks can improve out-of-distribution robustness,* Thirty-fifth Annual Conference on Neural Information Processing Systems (2021).

[188] Q. Zhao and C. Wressnegger, *Holistic Adversarially Robust Pruning,* The Eleventh International Conference on Learning Representations (2022).

[189] J. Lin, C. Gan, and S. Han, *Defensive Quantization: When Efficiency Meets Robustness,* International Conference on Learning Representations (2019).

[190] G. E. Hinton, O. Vinyals, and J. Dean, *Distilling the Knowledge in a Neural Network,* CoRR (2015).

[191] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, *Distillation as a defense to adversarial perturbations against deep neural networks,* 2016 IEEE symposium on security and privacy (SP) (2016).

[192] M. V. Narkhede, P. P. Bartakke, and M. S. Sutaone, *A review on weight initialization strategies for neural networks,* Artificial intelligence review (2022).

[193] D. Hendrycks, K. Lee, and M. Mazeika, *Using pre-training can improve model robustness and uncertainty,* International conference on machine learning (2019).

[194] K. He, R. Girshick, and P. Dollár, *Rethinking imagenet pre-training,* Proceedings of the IEEE/CVF International Conference on Computer Vision (2019).

[195] A. Kurakin, I. J. Goodfellow, and S. Bengio, *Adversarial examples in the physical world,* Artificial intelligence safety and security (2018).

[196] Q. Wang, K. Zhang, X. Liu, and C. L. Giles, *Verification of Recurrent Neural Networks Through Rule Extraction,* CoRR (2018).

[197] G. Singh, T. Gehr, M. Mirman, M. Püschel, and M. Vechev, *Fast and effective robustness certification,* The Thirty-second Annual Conference on Neural Information Processing Systems (2018).

[198] G. Katz, D. A. Huang, D. Ibeling, K. Julian, C. Lazarus, R. Lim, P. Shah, S. Thakoor, H. Wu, A. Zeljić, and others, *The marabou framework for verification and analysis of deep neural networks,* International Conference on Computer Aided Verification (2019).

[199] X. Huang, M. Kwiatkowska, S. Wang, and M. Wu, *Safety verification of deep neural networks,* International Conference on Computer Aided Verification (2017).

[200] M. Balunovic and M. Vechev, *Adversarial training and provable defenses: Bridging the gap,* International Conference on Learning Representations (2019).

[201] M. Casadio, E. Komendantskaya, M. L. Daggitt, W. Kokke, G. Katz, G. Amir, and I. Refaeli, *Neural network robustness as a verification property: a principled case study,* International Conference on Computer Aided Verification (2022).

[202] M. Augustin, A. Meinke, and M. Hein, *Adversarial robustness on in-and out-distribution improves explainability,* European Conference on Computer Vision (2020).

[203] J. Grabinski, P. Gavrikov, J. Keuper, and M. Keuper, *Robust Models are less Over-Confident,* The Thirty-sixth Annual Conference on Neural Information Processing Systems (2022).

[204] F. Utrera, E. Kravitz, N. B. Erichson, R. Khanna, and M. W. Mahoney, *Adversarially-Trained Deep Nets Transfer Better: Illustration on Image Classification,* International Conference on Learning Representations (2021).

[205] H. Salman, A. Ilyas, L. Engstrom, A. Kapoor, and A. Madry, *Do adversarially robust imagenet models transfer better?,* Thirty-fourth Annual Conference on Neural Information Processing Systems (2020).

[206] F. Croce, M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal, and M. Hein, *RobustBench: a standardized adversarial robustness benchmark,* Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (2021).

[207] F. Croce and M. Hein, *Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks,* International conference on machine learning (2020).

[208] A. Fawzi, S. Moosavi-Dezfooli, P. Frossard, and S. Soatto, *Classification regions of deep neural networks,* CoRR (2017).

[209] Q. Nguyen, M. C. Mukkamala, and M. Hein, *Neural networks should be wide enough to learn disconnected decision regions,* International Conference on Machine Learning (2018).

[210] C.-J. Simon-Gabriel, Y. Ollivier, L. Bottou, B. Schölkopf, and D. Lopez-Paz, *First-order adversarial vulnerability of neural networks and input dimension,* International Conference on Machine Learning (2019).

[211] A. Shafahi, W. R. Huang, C. Studer, S. Feizi, and T. Goldstein, *Are Adversarial Examples Inevitable?,* 7th International Conference on Learning Representations (2019).

[212] J. Gilmer, L. Metz, F. Faghri, S. S. Schoenholz, M. Raghu, M. Wattenberg, and I. J. Goodfellow, *Adversarial Spheres,* 6th International Conference on Learning Representations (2018).

[213] V. D'Amario, S. Srivastava, T. Sasaki, and X. Boix, *The Data Efficiency of Deep Learning Is Degraded by Unnecessary Input Dimensions,* Frontiers in Computational Neuroscience (2022).

[214] A. Ansuini, A. Laio, J. H. Macke, and D. Zoccolan, *Intrinsic dimension of data representations in deep neural networks,* Thirty-third Annual Conference on Neural Information Processing Systems (2019).

[215] R. Basri and D. W. Jacobs, *Efficient Representation of Low-Dimensional Manifolds using Deep Networks,* International Conference on Learning Representations (2017).

[216] S. Gong, V. N. Boddeti, and A. K. Jain, *On the intrinsic dimensionality of image representations,* Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2019).

[217] B. C. Brown, J. Juravsky, A. L. Caterini, and G. Loaiza-Ganem, *Relating Regularization and Generalization through the Intrinsic Dimension of Activations,* Has it Trained Yet? NeurIPS 2022 Workshop (2022).

[218] S. Recanatesi, M. Farrell, M. Advani, T. Moore, G. Lajoie, and E. Shea-Brown, *Dimensionality compression and expansion in Deep Neural Networks,* CoRR (2019).

[219] X. Ma, Y. Wang, M. E. Houle, S. Zhou, S. Erfani, S. Xia, S. Wijewickrema, and J. Bailey, *Dimensionality-driven learning with noisy labels,* International Conference on Machine Learning (2018).

[220] N. Frosst, N. Papernot, and G. Hinton, *Analyzing and improving representations with the soft nearest neighbor loss,* International conference on machine learning (2019).

[221] P. P. Brahma, D. Wu, and Y. She, *Why deep learning works: A manifold disentanglement perspective,* IEEE transactions on neural networks and learning systems (2015).

[222] M. Hauser and A. Ray, *Principles of Riemannian geometry in neural networks,* The Thirty-first Annual Conference on Neural Information Processing Systems (2017).

[223] J. Dapello, J. Feather, H. Le, T. Marques, D. Cox, J. McDermott, J. J. DiCarlo, and S. Chung, *Neural population geometry reveals the role of stochasticity in robust perception,* Thirty-fifth Annual Conference on Neural Information Processing Systems (2021).

[224] T. Toosi and E. Issa, *Brain-like representational straightening of natural movies in robust feedforward neural networks,* The Eleventh International Conference on Learning Representations (2023).

[225] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, *mixup: Beyond Empirical Risk Minimization,* International Conference on Learning Representations (2018).

[226] V. Verma, A. Lamb, C. Beckham, A. Najafi, I. Mitliagkas, D. Lopez-Paz, and Y. Bengio, *Manifold mixup: Better representations by interpolating hidden states,* International conference on machine learning (2019).

[227] U. Cohen, S. Chung, D. D. Lee, and H. Sompolinsky, *Separability and geometry of object manifolds in deep neural networks,* Nature communications (2020).

[228] G. Ortiz-Jimenez, A. Modas, S.-M. Moosavi, and P. Frossard, *Hold me tight! Influence of discriminative features on deep network boundaries,* Thirty-fourth Annual Conference on Neural Information Processing Systems (2020).

[229] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, *Caffe: Convolutional architecture for fast feature embedding,* Proceedings of the 22nd ACM international conference on Multimedia (2014).

[230] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, *Dropout: a simple way to prevent neural networks from overfitting,* The journal of machine learning research (2014).

[231] R. Salakhutdinov and G. Hinton, *Learning a nonlinear embedding by preserving class neighbourhood structure,* Artificial intelligence and statistics (2007).

[232] S. Chung, D. D. Lee, and H. Sompolinsky, *Classification and geometry of general perceptual manifolds,* Physical Review X (2018).

[233] N. Chen, A. Klushyn, F. Ferroni, J. Bayer, and P. Van Der Smagt, *Learning Flat Latent Manifolds with VAEs,* International Conference on Machine Learning (2020).

[234] R. Girshick, J. Donahue, T. Darrell, and J. Malik, *Rich feature hierarchies for accurate object detection and semantic segmentation,* Proceedings of the IEEE conference on computer vision and pattern recognition (2014).

[235] K. He, G. Gkioxari, P. Dollár, and R. Girshick, *Mask r-cnn,* Proceedings of the IEEE international conference on computer vision (2017).

[236] D. P. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization,* International Conference on Learning Representations (2015).

[237] E. Levina and P. J. Bickel, *Maximum likelihood estimation of intrinsic dimension,* Advances in neural information processing systems (2005).

[238] D. J. MacKay and Z. Ghahramani, *Comments on'maximum likelihood estimation of intrinsic dimension'by E. Levina and P. Bickel (2005),* The Inference Group Website, Cavendish Laboratory, Cambridge University (2005).

[239] W. He, B. Li, and D. Song, *Decision Boundary Analysis of Adversarial Examples,* International Conference on Learning Representations (2018).

[240] M. T. Ribeiro, S. Singh, and C. Guestrin, *"Why should I trust you?" Explaining the predictions of any classifier,* Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining (2016).

[241] S. Addepalli, S. Jain, G. Sriramanan, S. Khare, and V. B. Radhakrishnan, *Towards Achieving Adversarial Robustness Beyond Perceptual Limits,* ICML 2021 Workshop on Adversarial Machine Learning (2021).

[242] L. Engstrom, A. Ilyas, H. Salman, S. Santurkar, and D. Tsipras. *Robustness (Python Library),* 2019.

[243] A. Sanyal, P. H. Torr, and P. K. Dokania, *Stable Rank Normalization for Improved Generalization in Neural Networks and GANs,* International Conference on Learning Representations (2019).

[244] Y. Bai, X. Yan, Y. Jiang, S.-T. Xia, and Y. Wang, *Clustering effect of adversarial robust models,* Thirty-fifth Annual Conference on Neural Information Processing Systems (2021).