Geometry of adversarial robustness of deep networks: methods and applications

Thèse N° 9579

Présentée le 11 novembre 2019

à la Faculté des sciences et techniques de l'ingénieur
Laboratoire de traitement des signaux 4
Programme doctoral en informatique et communications

pour l'obtention du grade de Docteur ès Sciences

par

Seyed Mohsen MOOSAVI DEZFOOLI

Acceptée sur proposition du jury

Prof. P. Thiran, président du jury
Prof. P. Frossard, directeur de thèse
Prof. B. Schiele, rapporteur
Prof. D. Fleet, rapporteur
Prof. A. Krause, rapporteur

2019

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

There is no capital more useful than intellect and wisdom,
and there is no indigence more injurious than ignorance and unawareness.
— Ali bin Abi Talib (a.s)

Acknowledgements

Thank God that I could finally finish my PhD thesis!

I am delighted to have the opportunity to work under the supervision of Pascal Frossard. I would like to thank him for hosting me during the years of my PhD, and for the trust and the freedom he gave me in exploring research directions that I was interested in. I also greatly thank him for providing thorough and detailed feedbacks. I admire his patience with me, even in the saddest moments of my PhD life.

I would like to extend my sincerest thanks to the jury members of my thesis defense, Bernt Schiele, David Fleet, Andreas Krause, and Patrick Thiran. I appreciate their comments on the draft of this thesis and the fruitful discussion during the exam session.

I am extremely glad to have met and have worked with Alhussein Fawzi since the beginning of my PhD. Hussein's profound knowledge made him an ideal collaborator for me. I am extremely enthusiastic to discuss different problems with him and he is always ready to challenge me. I am not sure how much of this thesis would exist without his constant encouragement.

An exciting internship at Apple in California offered the opportunity to work with great researchers and engineers at Special Project Group. Especially, I would like to thank Oncel Tuzel and Ashish Shrivistava for such a wonderful time at Apple.

Many thanks to the former and the current LTS4 labmates Ana, Apostolos, Bastien, Beril, Chenglin, Clément, Clémentine, Dorina, Eda, Ersi, Francesca, Giuseppe, Guillermo, Hermina, Isabela, Laura, Mattia, Mireille, Pinar, Renata, Roberto, Sofia, Stefano, Sunok. My special thanks go to Mattia for sharing the office with me during my PhD years. It was really a pleasure to have a super neat and organized officemate. Also, I thank Bastien for helping me to write the French abstract of this thesis.

I also want to thank our wonderful secretary Anne De Witte for all the paperworks and her handling of administrative issues. Having such an amazing secretary like Anne is one of my best memories of LTS4. I also want to thank our former secretary Ms. Rosie De Pietro.

During my PhD, I was fortunate to supervise great internal and external students, Can Christos, Eric, Gu, Melika, Shi, and Yujia. They were a great source of motivation for me and I learned many things from them.

The Iranian community in Lausanne was a blessing to me during all these years away from home. I am thankful to such great people who never let me feel the absence of my family. There is a long list of people to name but if I want to name just one person, Fazel

was really like a brother to me and has helped me since day one in Switzerland.
I am indebted to my family and my family-in-law for always being there for me, and for their unconditional love and support. Last but not the least, I am thankful to Mojgan for her support, sacrifices and her patience with me during all these years. I am really grateful to her while I cannot give back all she has offered me.

Lausanne, 20 June 2019 Seyed-Mohsen Moosavi-Dezfooli

Abstract

We are witnessing a rise in the popularity of using artificial neural networks in many fields of science and technology. Deep neural networks in particular have shown impressive classification performance on a number of challenging benchmarks, generally in well controlled settings. However it is equally important that these classifiers satisfy robustness guarantees when they are deployed in uncontrolled (noise-prone) and possibly hostile environments. In other words, small perturbations applied to the samples should not yield significant loss to the performance of the classifier. Unfortunately, deep neural network classifiers are shown to be intriguingly vulnerable to perturbations and it is relatively easy to design noise that can change the estimated label of the classifier. The study of this high-dimensional phenomenon is a challenging task, and requires the development of new algorithmic tools, as well as theoretical and experimental analysis in order to identify the key factors driving the robustness properties of deep networks. This is exactly the focus of this PhD thesis.

First, we propose a computationally efficient yet accurate method to generate minimal perturbations that fool deep neural networks. It permits to reliably quantify the robustness of classifiers and compare different architectures. We further propose a systematic algorithm for computing universal (image-agnostic) and very small perturbation vectors that cause natural images to be misclassified with high probability. The vulnerability to universal perturbations is particularly important in security-critical applications of deep neural networks, and our algorithm shows that these systems are quite vulnerable to noise that is designed with only limited knowledge about test samples or classification architectures.

Next, we study the geometry of the classifier's decision boundary in order to explain the adversarial vulnerability of deep networks. Specifically, we establish precise theoretical bounds on the robustness of classifiers in a novel semi-random noise regime that generalizes both the adversarial and the random perturbation regimes. We show in particular that the robustness of deep networks to universal perturbations is driven by a key property of the curvature of their decision boundaries. Our analysis therefore suggests ways to improve the robustness properties of these classifiers to adversarial perturbations.

Finally, we build on the geometric insights derived in this thesis in order to improve the robustness properties of state-of-the-art image classifiers. We leverage a fundamental property in the curvature of the decision boundary of deep networks, and propose a method to detect small adversarial perturbations in images, and to recover the labels

of perturbed images. To achieve inherently robust classifiers, we further propose an alternative to the common adversarial training strategy, where we directly minimize the curvature of the classifier. This leads to adversarial robustness that is on par with adversarial training. Our proposed regularizer is thus an important step towards designing robust image classification systems.
In summary, we demonstrate in this thesis a new geometric approach to the problem of the adversarial vulnerability of deep networks, and provide novel quantitative and qualitative results that precisely describe the behavior of classifiers in adversarial settings. Our results in this thesis contribute to the understanding of the fundamental properties of state-of-the-art image classifiers that eventually will bring important benefits in safety-critical applications such as in self-driving cars, autonomous robots, and medical imaging.

Keywords: image classification, robustness, adversarial examples, decision boundary, deep neural networks, universal perturbations, curvature regularization, autonomous systems.

Résumé

Nous assistons à une augmentation de la popularité des réseaux de neurones artificiels dans de nombreux domaines de la science et de la technologie. Les réseaux profonds en particulier ont montré des performances de classification impressionnantes pour un certain nombre de tâches difficiles, généralement dans des environnements bien contrôlés. Cependant, il est également important que ces systèmes satisfassent des garanties de robustesse lorsqu'ils sont déployés dans des environnements non contrôlés (sujets au bruit) et éventuellement hostiles. En d'autres termes, de petites perturbations appliquées aux échantillons ne doivent pas entraîner de réduction significative des performances du système. Malheureusement, il a été démontré que les réseaux profonds sont extrêmement vulnérables aux perturbations, et il est relativement facile de concevoir un bruit pouvant modifier la classe estimée par ces méthodes. L'étude de ce phénomène en hautes dimensions est une tâche difficile qui nécessite le développement de nouveaux outils algorithmiques, ainsi que des analyses théoriques et expérimentales, afin d'identifier les facteurs clés qui déterminent les propriétés de robustesse des réseaux profonds. C'est exactement l'objet de cette thèse.

Premièrement, nous proposons une méthode peu coûteuse en temps de calcul, mais précise, pour générer des perturbations minimales qui trompent les réseaux de neurones profonds. Cette méthode permet de quantifier de manière fiable la robustesse des classifieurs et de comparer différentes architectures. Nous proposons de plus un algorithme pour calculer de manière systématique des perturbations universelles (pas directement liées aux images à disposition) et de très petite magnitude, qui entraînent une classification erronée des images naturelles avec une forte probabilité. La vulnérabilité aux perturbations universelles est particulièrement importante pour les applications des réseaux de neurones profonds pour lesquelles la sécurité est un aspect critique. Notre algorithme montre que ces systèmes sont très vulnérables à un bruit conçu avec une connaissance limitée des échantillons de test ou des architectures de classification.

Ensuite, nous étudions la géométrie de la frontière de décision du classifieur afin d'expliquer la vulnérabilité des réseaux profonds aux échantillons adverses. Plus précisément, nous établissons des bornes théoriques précises sur la robustesse des classifieurs dans un nouveau régime de bruit semi-aléatoire généralisant à la fois les régimes de perturbations adverses et aléatoires. Nous montrons en particulier que la robustesse des réseaux profonds face aux perturbations universelles est déterminée par une propriété clé de la courbure de leurs frontières de décision. Notre analyse suggère donc des moyens d'améliorer les propriétés

de robustesse de ces classifieurs aux perturbations adverses.
Enfin, nous exploitons les connaissances géométriques présentées dans cette thèse afin d'améliorer les propriétés de robustesse des classifieurs de l'état de l'art. Nous exploitons une propriété fondamentale de la courbure de la frontière de décision des réseaux profonds et proposons une méthode permettant de détecter les petites perturbations adverses dans les images et de retrouver les classes des images perturbées. Pour réaliser des classifieurs intrinsèquement robustes, nous proposons une alternative à la stratégie répandue d'entraînement adverse, dans laquelle nous minimisons directement la courbure du classifieur. Cela mène à une robustesse aux échantillons adverses comparable à un entraînement adverse qui constitue l'état de l'art. Le régularisateur que nous proposons est donc une étape importante dans la conception de systèmes robustes pour la classification d'images.
En résumé, nous présentons dans cette thèse une nouvelle approche géométrique du problème de la vulnérabilité des réseaux profonds aux échantillons adverses. Nous fournissons de nouveaux résultats quantitatifs et qualitatifs qui décrivent précisément le comportement des classifieurs dans des contextes adverses. Nos résultats dans cette thèse contribuent à la compréhension des propriétés fondamentales des classifieurs d'images de l'état de l'art, et contribueront de manière importante aux applications où la sécurité est un élément critique, teles que les voitures et robots autonomes, et l'imagerie médicale.

Mots-clés : classification d'images, robustesse, échantillons adverses, frontières de décision, réseaux de neurones profonds, perturbations universelles, régularisation de courbure, systèmes autonomes.

Contents

List of Figures

List of Tables

1 Introduction

"I don't know what's the matter with people: they don't learn by understanding; they learn by some other way - by rote, or something. Their knowledge is so fragile!"
— Richard Feynman

In practice, the generalization performance of a classifier is determined by measuring the so-called *test accuracy*, that is the classifier's performance on a set of held-out data samples. While the performance of the classifier in the presence of noise is a highly desired property in many real-world applications, the test accuracy does not fully reflect the classifier's performance in noisy and possibly hostile environments.

In particular, nowadays, image classifiers are widely used in safety-critical applications such as autonomous driving and face identification, hence it is extremely important to ensure that they show a robust performance in the worst-case scenarios. These scenarios can be due to either rare events or adversarial manipulations caused by malicious agents. To achieve robust image classification, image classifiers should ideally learn the underlying "concepts" which distinguish different classes. For example, given an image of a cat, corrupting a few pixels should not alter the decision of the classifier, as the image still represents a cat. Furthermore, we would generally want an image classifier to maintain its output if the changes are not perceptible to human eye, and hence, to be at least as robust as human visual system. Moreover, supervised image classification can be used to identify important features to explain the relationship between a class label and the data distribution. However, without being robust to small modifications, it can be misleading to employ image classifiers to learn the distinguishing factors between different classes. Such important problem can be an obstacle in achieving "interpretable machine learning" algorithms.

1.1 Adversarial robustness

In the recent years, deep neural networks have shown impressive classification performance (i.e., the test accuracy) on a diverse set of visual tasks. When deployed in real-world (noise-prone) environments, it is equally important that these classifiers maintain their performance. Intriguingly, it has been shown that these classifiers are indeed quite vulnerable to small and imperceptible manipulations of their input. In particular, they are shown to be extremely vulnerable to the so-called *adversarial examples*. As it is first introduced in the seminal work of [86], an adversarial example for a network can be informally defined as "...an imperceptible non-random perturbation to a test image, [which] arbitrarily changes the network's prediction...". Imperceptibility, as one of the key elements of this definition, is extremely difficult to be mathematically defined. In addition to being subjective, the imperceptibility can be interpreted in different ways for different types of data. It therefore results in a level of arbitrariness in the definition of adversarial examples in the literature. Despite of that, one common choice in the literature is to constrain the ℓ_p-norm of perturbations as a proxy for the perceptibility. In that, to find imperceptible perturbations, one should seek perturbations with the smallest ℓ_p-norm. Minimizing the ℓ_p-norm to find adversarial perturbations has the advantage of providing some degree of flexibility to cover a wide range of perturbations, thanks to the parameter p. For example, sparse and uniformly bounded perturbations can be defined using ℓ_1 and ℓ_∞ norms respectively. Furthermore, minimal ℓ_p-norm definition of adversarial perturbations allow us to perform theoretical analysis such as finding bounds on the magnitude of these perturbations via the concentration of measure inequalities.

Among many possible adversarial manipulations of the data, we primarily focus on additive perturbations as the simplest form of adversarial manipulations. As we will see, a geometric interpretation of additive adversarial perturbations permits us to study the local geometry of the decision boundary of deep neural network classifiers. Formally, for an image $\boldsymbol{x} \in \mathbb{R}^d$ and a given classifier with C classes $f : \mathbb{R}^d \to \{1, 2, \ldots, C\}$, one can define the ℓ_p-norm adversarial perturbation as

$$\begin{aligned} &\underset{\boldsymbol{r}}{\operatorname{argmin}} \ \|\boldsymbol{r}\|_p \\ &\text{s.t. } f(\boldsymbol{x}) \neq f(\boldsymbol{x} + \boldsymbol{r}). \end{aligned} \tag{1.1}$$

We choose this definition of adversarial perturbations throughout this manuscript unless it is stated otherwise.

There are many controversial open questions pertaining to the adversarial robustness of deep neural networks. Some authors argue in favour of the simplicity of deep learning models as the reason behind adversarial vulnerability [35]. On the contrary, some suggest that deep networks' excessive complexity harms their robustness to adversarial manipulations [86]. Furthermore, there is no established relation between the architecture

of deep neural networks and their robustness properties. Even more, there are some evidence suggesting that there is a fundamental limit on the robustness of deep networks, and due to high dimensionality, one cannot achieve an arbitrary robust classifier with the current architectures while still maintaining a high accuracy. On the practical side, building robust classifiers seems to be an extremely challenging problem [2]. In this thesis, we provide a novel geometric perspective on the problem of the adversarial vulnerability of deep neural networks to address some of these questions.

1.2 A geometric perspective on the robustness

The main theme of this thesis is to give a geometric perspective on the problem of the adversarial vulnerability of deep neural networks trained to solve image classification tasks. Our general goal is to exploit the connection between the adversarial robustness and the geometry of the decision boundary of deep networks, first to study various properties of adversarial perturbations, and secondly to propose novel geometry-inspired methods to evaluate and/or improve the robustness properties of deep neural networks. Our contribution in this thesis is a step towards improving the reliability of image classifiers and achieving better robustness properties in various classification tasks. In addition, our analysis contributes to a better understanding of the behaviour of deep neural networks, which are often seen as "black-box" models.

We are broadly interested in three aspects of adversarial perturbations:

- Efficiently computing them in high-dimensional classification problems in order to be able to compare robustness of different classifiers.
- Using these perturbations to study geometric properties of deep image classifiers.
- Developing methods to detect adversarial perturbations and/or to improve the robustness properties of deep networks.

Fast yet accurate methods to compute adversarial perturbations are needed to reliably evaluate the classifiers robustness properties. Such methods can also facilitate the study of the geometric properties of deep image classifiers. To this end, we provide computationally efficient yet accurate algorithms to find minimal ℓ_p-norm adversarial examples. We also propose a novel image-agnostic noise regime which exploits the geometric properties of deep image classifiers. Image-agnostic perturbations can be a real threat for security-critical applications.

The adversarial perturbations are not always harmful! They can be used as a powerful tool to study various geometric and topological properties of the decision regions of deep networks. In this thesis, we will demonstrate how effective adversarial perturbations are

in identifying the geometric properties of the decision boundary of deep image classifiers. Despite the fact that the analysis of the "shape" of the decision regions of these high-dimensional classifiers is generally intractable, we , for the first time, develop algorithms using adversarial perturbations to empirically study some of the topological properties of the decision regions of state-of-the-art deep networks.

Based on the developed algorithms and the geometric analysis, we propose efficient methods to improve the robustness properties of state-of-the-art image classifiers. In particular, we propose a geometric regularization technique which significantly improves the adversarial robustness to constrained the ℓ_p-norm perturbations. Moreover, we show that the geometric properties of the decision boundary of classifiers can be successfully used to detect the minimal ℓ_p-norm perturbations.

We should note that, besides image classifiers, the problem of the robustness to adversarial manipulations is equally important for other vision tasks such as semantic segmentation, object tracking, depth estimation, etc. As a first step towards understanding the robustness properties of deep networks, we however focus on the analysis and the quantification of the image classifiers' robustness to adversarial perturbations. It should be noted that although our focus is on images, our geometric approach can be easily extended to other modalities of data.

1.3 Thesis outline

The thesis is organized into three main parts: 1) Algorithms and techniques, 2) Geometric analysis, 3) Applications. We start, in Part I, by introducing the necessary algorithmic tools to study the robustness properties of deep image classifiers. Our proposed algorithms can be employed to efficiently evaluate and benchmark the robustness of classifiers in multiple settings. They further constitute our tools to analyse the local geometry of state-of-the-art classifiers. In Part II, we provide theoretical analysis of different types of adversarial perturbations. Based on the methods developed in Part I, we support these theoretical findings with the experiments on state-of-the-art deep networks. Our theoretical analysis sheds light on some of intriguing properties of deep neural networks, and suggest ways to improve their robustness to adversarial perturbations. Finally, Part III will be dedicated to demonstrate some applications of the methods and the analysis developed in the first two parts of this thesis.

In Chapter 2, we review some prior works related to the problem of adversarial robustness of image classifiers.

In Chapter 3, we study the problem of evaluating the robustness of image classifiers and in particular state-of-the-art deep neural networks. We show that despite the non-convexity of Eq. (1.1) that defines adversarial perturbations, it can be efficiently and

accurately solved using an iterative linearization of the classifier's decision function. Extensive experimental evaluations show the effectiveness of our method for computation of perturbations on high-dimensional image classification tasks. Furthermore, to perform data augmentation, adversarial examples generated using our accurate method are shown to be more effective compared to less accurate methods. We conclude this chapter by demonstrating the flexibility of the proposed algorithm in exploring the space of adversarial examples to generate structured perturbations.

Next, in Chapter 4, we introduce image-agnostic adversarial perturbations, coined "universal perturbations", for deep image classifiers. In contrast with typical adversarial perturbations defined in Eq. (1.1), universal perturbations do not depend on individual input images. We propose a greedy algorithm to generate these perturbations to fool state-of-the-art natural image classifiers. Though they are originally generated to transfer across different images, they surprisingly generalize well across different architectures.

The general focus of Part II is to study and quantify the local geometry of deep classifiers in order to explain the existence of adversarial perturbations. In Chapter 5, we study the *hypothesis* that "having a flat decision boundary causes deep networks to become vulnerable to adversarial perturbations". To do so, we introduce a novel *semi-random noise* regime, which generalizes random and adversarial perturbations. Semi-random noise regime permits us to explore the space of adversarial examples in the vicinity of a given datapoint, and to study the effect of flatness of decision boundary on the existence of adversarial perturbations. We particularly establish precise bounds on the robustness of classifiers to semi-random noise that depend on the curvature of the classifier's decision boundary. We conclude Chapter 5 with the experimental results showing that our theoretical estimates are very accurately satisfied by state-of-the-art deep image classifiers, which provides evidence in favour of the aforementioned "flatness" hypothesis.

In Chapter 6, we analyze theoretically the robustness of classifiers to universal perturbations, under two geometrical decision boundary models: locally flat and curved. The former corresponds to the previously mentioned "flatness" hypothesis, and it guarantees the existence of universal perturbations, provided the normal vectors in the vicinity of datapoints are correlated. The curved model instead relates the robustness to universal perturbations, to the existence of a shared subspace along which the decision boundary is highly curved. We empirically verify both assumptions for deep networks, and we show that the curved model better explains the vulnerability of deep networks to universal perturbations.

Chapter 7 is focused on studying the geometry and topology of the decision regions of deep image classifiers. We demonstrate how adversarial perturbations can be used to examine the local geometry of classifiers. Specifically, we resolve the seeming contradiction between the observations in Chapter 5 and 6 regarding the geometry of state-of-the-art networks by a thorough analysis of the curvature of their decision boundary, and thus

provide a more complete picture of the geometry of decision boundary in the vicinity of images. In particular, we empirically show that the decision boundaries learned by deep image classifiers are flat along most directions, and that some curved directions are shared across datapoints. Furthermore, we provide a method to study topological properties of classifiers in particular path-connectedness of decision regions. We finally leverage an observation on the asymmetry in the local curvature of image classifiers, and propose an effective method for detecting adversarially perturbed samples. This shows that the study of the geometry of state-of-the-art deep networks is not only the key from an analysis perspective, but it can also lead to the methods to improve their robustness properties.

Finally in Chapter 8, we demonstrate the advantage of our geometric analysis in developing efficient regularizers for improving robustness of deep networks. In particular, we provide theoretical and empirical evidence showing the existence of a strong link between the curvature of classifiers, in the vicinity of datapoints, and their robustness to adversarial perturbations. More specifically, we analyse the geometry of *adversarial training*, i.e., data augmentation with adversarial examples, and show that adversarial training significantly decreases the curvature of the loss landscape of deep networks with respect to the input. On the other hand, encouraging small curvatures significantly improves the robustness of deep networks and even achieves performance on par with adversarial training. Contrary to prior works attributing the adversarial vulnerability of deep classifiers to their "excessive linearity", our result somewhat surprisingly shows that one needs to decrease curvature (or increase the "linearity") to improve the robustness.

In summary, we develop a set of scalable yet accurate algorithmic tools to experimentally and theoretically analyse the geometric characteristics of the decision boundaries induced by state-of-the-art classifiers. This analysis does not only highlights the important geometric factors pertaining to the robustness properties of classifiers, but more importantly, it also leads to more robust classification systems.

2 Related work

In this chapter, we review some of the relevant works from the literature that are linked to the problems studied in this thesis. We start in Section 2.1 by giving a brief summary of the significant works that triggered the fundamental research questions related to adversarial robustness of image classifiers. In Section 2.2, we particularly focus on the methods to evaluate the robustness properties of deep neural networks. Next, we review related works that study the geometric properties of these classifiers in Section 2.3. Finally, in Section 2.4, we focus on the major efforts in improving the robustness properties of image classifiers.

2.1 Adversarial robustness of deep networks

Deep neural network architectures have achieved state-of-the-art performance in solving complex learning tasks. In particular, they have shown an astounding performance on challenging visual classification benchmarks [38, 49, 88, 51]. Despite this success, many fundamental, and often intriguing, properties of these data representation learning algorithms are still not understood. One of such intriguing properties of deep networks is their robustness to various forms of perturbations. In the seminal work of [86], the phenomenon of adversarial vulnerability of deep neural networks was first introduced. The authors particularly showed that deep network image classifiers are susceptible to small well-sought additive perturbations in the data, coined *adversarial perturbations*, even if they achieve a high generalization performance. Surprisingly, the resulting perturbed data , called *adversarial examples*, are indistinguishable from the original data to the human eye. In [86], for a given classifier $\hat{k}$ and a given input image $\boldsymbol{x} \in [0,1]^d$, an adversarial example is found by solving a series of the following optimization problems – for different values of C – using a quasi-Newton optimization algorithm [10]:

$$\begin{aligned} \min_{\boldsymbol{r}} \; & J(\boldsymbol{x}+\boldsymbol{r}, t) + C\|\boldsymbol{r}\|_2 \\ \text{s.t. } & \boldsymbol{x}+\boldsymbol{r} \in [0,1]^d \end{aligned}$$

where $J(\boldsymbol{x}, t)$ is the cost used to train the classifier, and t is a target label different than the original label of $\boldsymbol{x}$. Despite the surprising effectiveness of this method in finding imperceptible adversarial perturbations, it is not scalable to high dimensional image classification tasks, and hence not a suitable method to study the robustness properties of state-of-the-art image classifiers in general. More importantly, such practical difficulty may give the false impression that adversarial examples are of zero-measure in the input space, as it is speculated in [86].

The introduction of *Fast Gradient Sign* method (FGSM) in [35] showed that adversarial examples can be found almost effortlessly for state-of-the-art image classifiers. This method computes adversarial examples by going in the direction of the sign of the gradient of the loss function J:

$$\epsilon \operatorname{sign}\left(\nabla_{\boldsymbol{x}} \left(\operatorname{loss}(\boldsymbol{x}, k(\boldsymbol{x}))\right)\right), \tag{2.1}$$

where $k(\boldsymbol{x})$ is the target label associated with $\boldsymbol{x}$, and ϵ is the ℓ_∞-norm of the perturbation. Despite its efficiency, this one-step algorithm provides only a coarse approximation of the minimal perturbation vectors, therefore it is not a reliable tool in studying the worst-case robustness properties of deep neural networks.

We finally note that while adversarial attacks for Support Vector Machine (SVM) classifiers has been introduced in [7], the study of the robustness properties of deep neural network classifiers has become an active area of research mostly since the seminal work of [86].

2.2 Evaluating robustness of deep networks

Additive perturbations. Robustness of classifiers to additive perturbations, either random or adversarial noise, is highly desired in many real-world applications of deep neural networks. Therefore, computationally efficient yet accurate methods are essential in order to study and evaluate the robustness properties of these networks. The methods proposed in [86, 35] to find adversarial examples either are computationally expensive or provide an inaccurate estimation of the robustness of classifiers.

Besides the methods mentioned in the previous section, many other methods have been developed to generate structured and unstructured additive adversarial perturbations. Such carefully crafted perturbations are generally estimated by solving an optimization problem. In [5], finding adversarial perturbations for ReLU networks is cast as a linear program. The authors of [50] propose an iterative version of the algorithm presented in [35]. In [13], adversarial perturbations are found by converting the constrained optimization problem defined in (1.1) to an unconstrained one. Moreover, additive adversarial perturbations for deep neural networks can be generated to also manipulate their internal representations [81]. While such an adversarial example is visually similar to one image, its corresponding internal representation is extremely similar to another

image, from a different class. In [4], a generative network is used to generate additive adversarial perturbations.

In some applications, evaluating the robustness of classifiers to perturbations with certain structures, such as sparsity, may be required. Sparse additive perturbations have thus been studied in the literature. In [75], a saliency-based method is proposed to perturb a few pixels of the input to fool an image classifier. Evolutionary algorithms, in [84], are also shown to be effective in generating sparse perturbations. However, neither of these methods are scalable to high-dimensional classification tasks. Recently, a geometry-inspired and computationally efficient method to find sparse perturbations is proposed in [66].

Band-limited perturbations, as another type of structured perturbations, are first studied in [29]. Such perturbations shed light on the robustness of deep neural networks in different frequency bands. In particular, in [100], it has been shown that the low-frequency perturbations computed for one network transfer better than their high-frequency counterparts to a different network. Such property concerns the black-box robustness of deep networks, as we will see later in this chapter.

Beside the challenge of evaluating the robustness of image classifiers to additive perturbations, evaluating the robustness properties can become even more challenging in more general settings.

Non-additive perturbations. Adversarial examples can be generated in more sophisticated ways rather than by merely adding a perturbation vector to the datapoints. Such non-additive perturbations, such as geometric transformations, are more likely to occur in real-world scenarios. Most of such manipulations cannot be modelled by additive perturbations, and their efficient computation requires more sophisticated optimization techniques. Furthermore, ℓ_p-norms become ineffective in measuring the robustness of deep networks to non-additive perturbations. In particular, the problem of finding the minimal *geometric* transformation that fools image classifiers is studied in [27, 46]. They provided quantitative measure of the robustness of classifiers to geometric transformations. Some have assessed the robustness of deep neural networks to other perturbation regimes such occlusions [82, 28, 25], and deformations [28, 45, 96]. Compared to additive perturbations, invariance to these natural nuisances play a more important role in building classifiers used in real-world applications.

Black-box perturbations. Evaluating the robustness of classifiers in black-box settings, where an adversary has only a partial knowledge of the classifiers, is extremely important from the security point of view. The robustness of classifiers when the adversary has only access to a surrogate model is examined in [7]. One of the intriguing

properties of adversarial examples is their transferability across different models [86]. In that, an adversarial example computed for one network can cause misclassification in another network. Transferability of adversarial examples can be exploited to devise efficient black-box attacks [57, 100]. Even worse, when the adversary has only access to the classifier's decision, query-based methods can be used to craft adversarial examples [8]. Though query-based methods to attack classifiers are generally computationally expensive, some prior information can be exploited to reduce the number of the queries in order to attack deep neural networks [41, 58].

Finally note that while we focus in this thesis on adversarial examples for image classification, such examples exist for other visual tasks such as object detection and semantic segmentation [17], and other modalities of data such as text [23] and speech [11].

2.3 Geometric analysis of image classifiers

Understanding the causes of the adversarial vulnerability of deep neural networks has been an active and controversial area of research. The seminal work of [86] attributed the adversarial instability to the large Lipschitz constant associated to the internal representations of deep neural networks which creates "blind-spots" in the classifier. On the contrary, in [87], the authors empirically show that adversarial examples are not isolated points, but rather occupy dense regions of the input space. As opposed to [86], where adversarial vulnerability is attributed to "excessive complexity" of deep networks, in [35], the "excessive linearity" of these networks are thought to be responsible for such vulnerabilities.

In an attempt to theoretically analyse the geometric properties pertaining to the adversarial vulnerability, the authors in [26] studied the problem of adversarial perturbations on some simple families of classifiers, and provided upper bounds on the robustness of these classifiers. They concluded that even linear classifiers trained on high-dimensional data are highly susceptible to adversarial perturbations. By pursuing a geometric approach, in [89], the authors provide a new explanation for the existence of adversarial examples based on the tilting of the decision boundary with respect to the data manifold.

While the geometry of classification regions and decision functions induced by traditional classifiers (such as linear and kernel SVM) is fairly well understood, the same fundamental geometric properties are to a large extent unknown for state-of-the-art deep neural networks. Yet, to understand the recent success of deep neural networks and potentially address their weaknesses (such as their instability to perturbations), an understanding of these geometric properties remains primordial.

The geometric and topological properties of deep neural networks can be studied either in *the input space* or in *the weight space* of these networks. In [16, 21, 15, 22], the geometry

of the *optimization landscape* in the weight space is studied; in particular, generalization of deep networks is shown to be intimately related to geometric properties of the optimization landscape (e.g., width of a minima). Related to our work, in [34], the authors study the optimization landscape of deep neural networks, where the connectedness of solutions with low error is shown in the weight space. An algorithm is provided to assess the nature of this connection in the weight space; empirical evidence supports the existence of "easy" paths between trained models. We follow here a similar goal to that of [34], but are interested instead in the connectivity of deep networks in the *input space* (and not the weight space). Finally, we note that graph-based techniques have been proposed in [64, 3] to analyze the classification regions of shallow neural networks; we rather focus in this thesis on the new generation of deep neural networks, which have shown remarkable performance.

The geometric properties of the decision boundary and classification regions of deep networks have comparatively received little attention. Closer to our work, in [76], the authors employ tools from Riemannian geometry to study the expressivity of random deep neural networks. In particular, the largest principal curvatures are shown to increase exponentially with the depth; the decision boundaries hence become more complex with depth. We however provide in this thesis a complementary analysis of the decision boundary, where the curvature of the decision boundary along *all* directions are analyzed (and not only in the direction of largest curvature). One of the contributions of this thesis is to provide both theoretical and empirical analyses to characterize the adversarial robustness of deep networks using the geometry of their decision boundary, and leverage them to improve the robustness of deep neural networks to adversarial perturbations.

2.4 Improving robustness of deep networks

In general, robust classification can be achieved either by detecting adversarial examples or by making data samples further from the decision boundary of the classifier. There has been a large body of work on designing more robust deep classifiers. These works can be broadly categorized as follows.

Adversarial training. One of the earliest attempts to build more robust classifiers is called "adversarial training" where, naively, the training data is augmented with adversarial examples [86, 35]. Different versions of adversarial training have been developed since then. The connection between adversarial training and Robust Optimization (RO) has been highlighted in [1]. One of the risks of using adversarial training is that it can cause the classifier to overfit to specific types of adversarial examples. However, in [70], it is shown that adversarial training using minimal ℓ_p perturbations (as defined in (1.1)) is more effective than using FGSM samples [35]. As another solution to avoid overfitting, adversarial training using attacks on an ensemble of networks is suggested in [90]. At

the time of writing this thesis, surprisingly, the most effective scheme to improve the robustness is shown to be the adversarial training proposed in [62], where the adversarial examples generated by the method of [50] are used.

Denoising-based methods. The simple idea behind denoising-based methods is to mitigate the effect of adversarial perturbations by removing the noise from the input image either by applying input transformations such as foeviation mechanism [60], JPEG compression [19], and quantization [37] or by learning a denoising operator [72]. Similarly to the input image, denoising techniques can also be applied to the internal representations of the network [55].

Regularization-based methods. Increasing the stability of the classifier has been one of the favourite methods to improve the adversarial robustness. In [36], the authors introduced a smoothness penalty in the training procedure that tries to boost the robustness of the classifier via penalizing the norm of the gradient in each layer. Many other works have attempted to improve the robustness using the gradient or the full Jacobian regularization [61, 78, 42]. In [18, 97], regularizing the spectral properties of individual convolutional filters has been proposed to improve robustness. Inspired by SVMs, large-margin deep neural networks have been introduced in [24]. It is worth to note that most of these methods penalize the classifier to have a small Lipschitz constant, which, generally, might significantly harm its generalization performance. In this thesis, we however introduce a promising second-order regularization technique that can perform as well as adversarial training.

Detection-based methods. In parallel, for some applications of image classification, it might be enough to detect adversarial examples instead of improving the robustness of the classifier. To do so, detector networks can be trained to distinguish adversarial examples from clean images [65]. Using Bayesian uncertainty estimation, model confidence can be investigated to detect adversarial examples [32]. However, in [12], it has been shown that many detection methods can be easily by-passed, and even if they successfully detect all the known adversarial attacks, they still remain susceptible to new unknown attacks.

It should be noted that many of the proposed methods to improve the robustness obscure the model rather than make the model truly robust against all attacks [92, 2]. One method however stands out, adversarial training, which has shown to be empirically robust against all designed attacks. One of our contributions in this thesis is to provide an analysis of this phenomenon, and propose a regularization strategy, which mimics the effect of adversarial training. Our method, while being computationally less expensive, can achieve close to state-of-the-art adversarial performance in some standard classification benchmarks.

2.5 Summary

We summarize the main points of this chapter, in the light of the contributions of this thesis and upcoming challenges:

- Deep neural networks are shown to be extremely vulnerable to adversarial manipulations. Different methods have been developed to assess their robustness properties in various adversarial settings; however, many of these methods lack either the scalability or the accuracy. One of the goals of this thesis is to provide scalable methods to accurately evaluate the robustness of deep networks to adversarial perturbations.
- Due to the complexity and the high-dimensionality of deep neural networks, their geometric properties have rather received little attention. We here provide a thorough study on the geometry of deep networks to shed light on their robustness properties.
- Designing methods to increase the robustness of deep networks is an active area of research. However, despite all the efforts so far, no satisfactory solution yet exists to achieve sufficiently robust image classifiers for many image classification benchmarks. In this thesis, we take a step towards understanding the key elements contributing to the success of some of the best known methods to improve robustness properties of deep networks.

Part I Tools and algorithms

3 Adversarial perturbations

"Truth is ever to be found in simplicity, and not in the multiplicity and confusion of things."
— Isaac Newton

3.1 Introduction

In this chapter , we propose a fast yet accurate algorithm to estimate the robustness of deep image classifiers to additive adversarial perturbations. Additive adversarial perturbations are small perturbations sought to change the estimated label of the classifier. Deep neural networks are shown to be unstable to very small and often imperceptible additive adversarial perturbations [86]. However, the method originally provided in [86] does not scale well to high-dimensional datasets. An accurate method for finding the adversarial perturbations is thus useful to study and compare the robustness properties of different image classifiers. Furthermore, as we will see later in this chapter, thanks to the proximity of datapoints to the decision boundary of deep neural networks, an accurate algorithm to find adversarial perturbations can be employed to study the geometric properties of the decision boundary of deep networks in the vicinity of datapoints.

We propose an efficient algorithm called DeepFool to compute minimal perturbations of deep image classifiers. Since one can find such perturbations in closed-form for linear classifiers, the main idea behind DeepFool is to iteratively linearize the classifier's decision function, and therefore reduce the problem of finding minimal perturbations to a series of closed-form update rules. Through extensive experimental comparisons, we show that 1) our method computes adversarial perturbations more reliably and efficiently than some of

Part of this chapter has been published in

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016,

Alhussein Fawzi*, Seyed Moosavi-Dezfooli*, and Pascal Frossard. Robustness of classifiers: from adversarial to random noise. In *Neural Information Processing Systems (NIPS)*, 2016. (*: Equal contribution)

the existing methods 2) augmenting training data with adversarial examples significantly increases the robustness to adversarial perturbations. Furthermore, we demonstrate the flexibility of the DeepFool algorithm to study the robustness of deep neural networks in more general settings.

The rest of this chapter is organised as follows. In Section 3.2, we introduce our computationally efficient algorithm, called DeepFool, to find minimal adversarial perturbations, and provide experimental results to compare the proposed optimisation method, for computing the robustness, to other existing methods. In Section 3.3, we provide an efficient and accurate way to enhance the robustness performance of deep networks by proper fine-tuning. Finally in Section 3.4, we provide variants of DeepFool to study adversarial perturbations confined in a subspace.

3.2 DeepFool

We recall that, for a given classifier, we define an adversarial perturbation as the *minimal* perturbation $\boldsymbol{r}$, in the sense of the ℓ_p-norm, that is sufficient to change the estimated label[1] $\hat{k}(\boldsymbol{x})$:

$$\Delta(\boldsymbol{x}; \hat{k}) := \min_{\boldsymbol{r}} \|\boldsymbol{r}\|_p \text{ subject to } \hat{k}(\boldsymbol{x} + \boldsymbol{r}) \neq \hat{k}(\boldsymbol{x}), \tag{3.1}$$

where $\boldsymbol{x}$ is an image and $\hat{k}(\boldsymbol{x})$ is the estimated label. We call $\Delta(\boldsymbol{x}; \hat{k})$ the robustness of $\hat{k}$ at point $\boldsymbol{x}$.

Since a multiclass classifier can be viewed as an aggregation of binary classifiers, we first propose the algorithm for binary classifiers. That is, we assume here $\hat{k}(\boldsymbol{x}) = \text{sign}(f(\boldsymbol{x}))$, where f is an arbitrary scalar-valued image classification function $f : \mathbb{R}^n \to \mathbb{R}$. We also denote by $\mathscr{F} \triangleq \{\boldsymbol{x} : f(\boldsymbol{x}) = 0\}$ the level set at zero of f. We begin by analyzing the case where f is an affine classifier $f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b$, and then derive the general algorithm, which can be applied to any differentiable binary classifier f.

3.2.1 Binary classifiers

In the case where the classifier f is affine, it can easily be seen that the robustness of f at point $\boldsymbol{x}_0$, $\Delta(\boldsymbol{x}_0; f)$[2], is equal to the distance from $\boldsymbol{x}_0$ to the separating affine hyperplane $\mathscr{F} = \{\boldsymbol{x} : \boldsymbol{w}^T\boldsymbol{x} + b = 0\}$ (Fig. 3.1). The minimal perturbation to change the classifier's decision corresponds to the orthogonal projection of $\boldsymbol{x}_0$ onto $\mathscr{F}$. It is given by

[1] Throughout this thesis, perturbation vectors sending a datapoint exactly to the boundary are assumed to change the estimated label of the classifier.

[2] From now on, we refer to a classifier either by f or its corresponding discrete mapping $\hat{k}$. Therefore, $\rho_{\text{adv}}(\hat{k}) = \rho_{\text{adv}}(f)$ and $\Delta(\boldsymbol{x}; \hat{k}) = \Delta(\boldsymbol{x}; f)$.

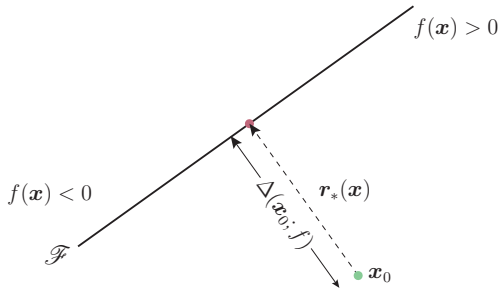


Figure 3.1 – Adversarial examples for a linear binary classifier.

the closed-form formula:

$$\begin{aligned} \boldsymbol{r}_*(\boldsymbol{x}_0) &:= \arg\min \|\boldsymbol{r}\|_2 \text{ subject to } \operatorname{sign}(f(\boldsymbol{x}_0 + \boldsymbol{r})) \neq \operatorname{sign}(f(\boldsymbol{x}_0)) \\ &= -\frac{f(\boldsymbol{x}_0)}{\|\boldsymbol{w}\|_2^2}\boldsymbol{w}. \end{aligned} \tag{3.2}$$

Assuming now that f is a general differentiable binary classifier, we adopt an iterative procedure to estimate the robustness $\Delta(\boldsymbol{x}_0; f)$. Specifically, at each iteration, f is linearized around the current point $\boldsymbol{x}_i$ and the minimal perturbation of the linearized classifier is computed as

$$\arg\min_{\boldsymbol{r}_i} \|\boldsymbol{r}_i\|_2 \text{ subject to } f(\boldsymbol{x}_i) + \nabla f(\boldsymbol{x}_i)^T \boldsymbol{r}_i = 0. \tag{3.3}$$

The perturbation $\boldsymbol{r}_i$ at iteration i of the algorithm is computed using the closed form solution in Eq. (3.3), and the next iterate $\boldsymbol{x}_{i+1}$ is updated. The algorithm stops when $\boldsymbol{x}_{i+1}$ changes sign of the classifier. The DeepFool algorithm for binary classifiers is summarized in Algorithm 1 and a geometric illustration of the method is shown in Fig. 3.2.

Algorithm 1 DeepFool for binary classifiers

input: Image $\boldsymbol{x}$, classifier f.
output: Perturbation $\hat{\boldsymbol{r}}$.
Initialize $\boldsymbol{x}_0 \leftarrow \boldsymbol{x}$, $i \leftarrow 0$.
while $\operatorname{sign}(f(\boldsymbol{x}_i)) = \operatorname{sign}(f(\boldsymbol{x}_0))$ **do**
    $\boldsymbol{r}_i \leftarrow -\frac{f(\boldsymbol{x}_i)}{\|\nabla f(\boldsymbol{x}_i)\|_2^2}\nabla f(\boldsymbol{x}_i)$,
    $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \boldsymbol{r}_i$,
    $i \leftarrow i + 1$.
end while
return $\hat{\boldsymbol{r}} = \sum_i \boldsymbol{r}_i$.

In practice, the above algorithm can often converge to a point on the zero level set $\mathscr{F}$. In order to reach the other side of the classification boundary, the final perturbation vector $\hat{\boldsymbol{r}}$ is multiplied by a constant $1 + \eta$, with $\eta \ll 1$. In our experiments, we have used $\eta = 0.02$.

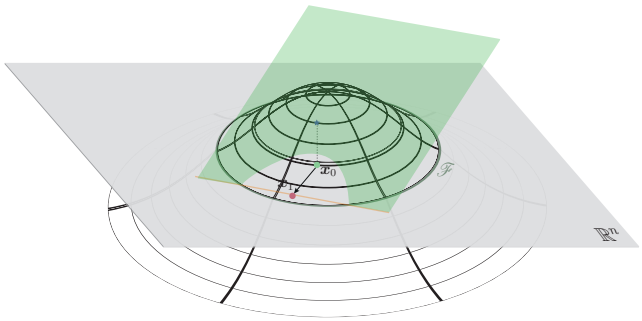


Figure 3.2 – Illustration of Algorithm 1 for $n = 2$. Assume $\boldsymbol{x}_0 \in \mathbb{R}^n$. The green plane is the graph of $\boldsymbol{x} \mapsto f(\boldsymbol{x}_0) + \nabla f(\boldsymbol{x}_0)^T(\boldsymbol{x} - \boldsymbol{x}_0)$, which is tangent to the classifier function (wire-framed graph) $\boldsymbol{x} \mapsto f(\boldsymbol{x})$. The orange line indicates where $f(\boldsymbol{x}_0) + \nabla f(\boldsymbol{x}_0)^T(\boldsymbol{x} - \boldsymbol{x}_0) = 0$. $\boldsymbol{x}_1$ is obtained from $\boldsymbol{x}_0$ by projecting $\boldsymbol{x}_0$ on the orange hyperplane of $\mathbb{R}^n$.

3.2.2 Multi-class classifiers

We now extend the DeepFool method to the multiclass case. The most common used scheme for multiclass classifiers is one-vs-all. Hence, we also propose our method based on this classification scheme. In this scheme, the classifier has c outputs where c is the number of classes. Therefore, a classifier can be defined as $f : \mathbb{R}^n \to \mathbb{R}^c$ and the classification is done by the following mapping:

$$\hat{k}(\boldsymbol{x}) = \arg\max_k f_k(\boldsymbol{x}), \tag{3.4}$$

where $f_k(\boldsymbol{x})$ is the output of $f(\boldsymbol{x})$ that corresponds to the k^{th} class. Similarly to the binary case, we first present the proposed approach for the linear case and then we generalize it to other classifiers.

Affine multiclass classifier

Let $f(\boldsymbol{x})$ be an affine classifier, i.e., $f(\boldsymbol{x}) = \mathbf{W}^\top \boldsymbol{x} + \boldsymbol{b}$ for a given $\mathbf{W}$ and $\boldsymbol{b}$. Since the mapping $\hat{k}$ is the outcome of a one-vs-all classification scheme, the minimal perturbation to fool the classifier can be rewritten as follows

$$\begin{aligned} &\arg\min_{\boldsymbol{r}} \|\boldsymbol{r}\|_2 \\ &\text{s.t. } \exists k : \boldsymbol{w}_k^\top(\boldsymbol{x}_0 + \boldsymbol{r}) + b_k \geq \boldsymbol{w}_{\hat{k}(\boldsymbol{x}_0)}^\top(\boldsymbol{x}_0 + \boldsymbol{r}) + b_{\hat{k}(\boldsymbol{x}_0)}, \end{aligned} \tag{3.5}$$

where $\boldsymbol{w}_k$ is the k^{th} column of $\mathbf{W}$. Geometrically, the above problem corresponds to the computation of the distance between $\boldsymbol{x}_0$ and the *complement* of the convex polyhedron P,

$$P = \bigcap_{k=1}^{c} \{\boldsymbol{x} : f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}) \geq f_k(\boldsymbol{x})\}, \tag{3.6}$$

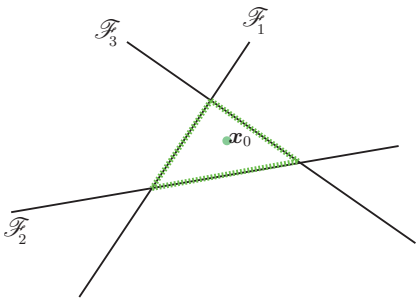


Figure 3.3 – For $\boldsymbol{x}_0$ belonging to class 4, let $\mathscr{F}_k = \{\boldsymbol{x} : f_k(\boldsymbol{x}) - f_4(\boldsymbol{x}) = 0\}$. These hyperplanes are depicted in solid lines and the boundary of P is shown in green dotted line.

where $\boldsymbol{x}_0$ is located inside P. We denote this distance by $\mathbf{dist}(\boldsymbol{x}_0, P^c)$. The polyhedron P defines the region of the space where f outputs the label $\hat{k}(\boldsymbol{x}_0)$. This setting is depicted in Fig. 3.3. The solution to the problem in Eq. (3.5) can be computed in closed form as follows. Define $\hat{l}(\boldsymbol{x}_0)$ to be the closest hyperplane of the boundary of P (e.g. $\hat{l}(\boldsymbol{x}_0) = 3$ in Fig. 3.3). Formally, $\hat{l}(\boldsymbol{x}_0)$ can be computed as follows

$$\hat{l}(\boldsymbol{x}_0) = \underset{k \neq \hat{k}(\boldsymbol{x}_0)}{\arg\min} \frac{\left| f_k(\boldsymbol{x}_0) - f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_0) \right|}{\|\boldsymbol{w}_k - \boldsymbol{w}_{\hat{k}(\boldsymbol{x}_0)}\|_2}. \tag{3.7}$$

The minimum perturbation $\boldsymbol{r}_*(\boldsymbol{x}_0)$ is the vector that projects $\boldsymbol{x}_0$ on the hyperplane indexed by $\hat{l}(\boldsymbol{x}_0)$, i.e.,

$$\boldsymbol{r}_*(\boldsymbol{x}_0) = \frac{\left| f_{\hat{l}(\boldsymbol{x}_0)}(\boldsymbol{x}_0) - f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_0) \right|}{\|\boldsymbol{w}_{\hat{l}(\boldsymbol{x}_0)} - \boldsymbol{w}_{\hat{k}(\boldsymbol{x}_0)}\|_2^2} (\boldsymbol{w}_{\hat{l}(\boldsymbol{x}_0)} - \boldsymbol{w}_{\hat{k}(\boldsymbol{x}_0)}). \tag{3.8}$$

In other words, we find the closest projection of $\boldsymbol{x}_0$ on faces of P.

General classifier

We now extend the DeepFool algorithm to the general case of multiclass differentiable classifiers. For general non-linear classifiers, the set P in Eq. (3.6) that describes the region of the space where the classifier outputs label $\hat{k}(\boldsymbol{x}_0)$ is no longer a polyhedron. Following the explained iterative linearization procedure in the binary case, we approximate the set P at iteration i by a polyhedron $\tilde{P}_i$

$$\tilde{P}_i = \bigcap_{k=1}^{c} \Big\{ \boldsymbol{x} : f_k(\boldsymbol{x}_i) - f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_i) + \nabla f_k(\boldsymbol{x}_i)^\top \boldsymbol{x} - \nabla f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_i)^\top \boldsymbol{x} \leq 0 \Big\}. \tag{3.9}$$

We then approximate, at iteration i, the distance between $\boldsymbol{x}_i$ and the complement of P, $\mathbf{dist}(\boldsymbol{x}_i, P^c)$, by $\mathbf{dist}(\boldsymbol{x}_i, \tilde{P}_i^c)$. Specifically, at each iteration of the algorithm, the perturbation vector that reaches the boundary of the polyhedron $\tilde{P}_i$ is computed, and the

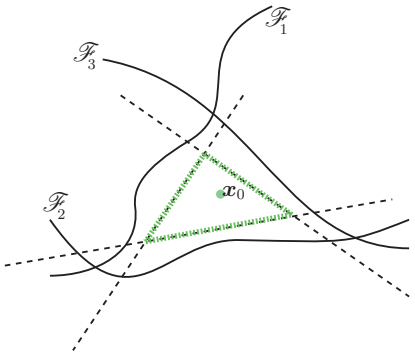


Figure 3.4 – For $\boldsymbol{x}_0$ belonging to class 4, let $\mathscr{F}_k = \{\boldsymbol{x} : f_k(\boldsymbol{x}) - f_4(\boldsymbol{x}) = 0\}$. The linearized zero level sets are shown in dashed lines and the boundary of the polyhedron $\tilde{P}_0$ in green.

current estimate is updated. A schematic representation of the linearization of the decision boundary is shown in Fig. 3.4. The method is given in Algorithm 2. It should be noted that the proposed algorithm operates in a greedy way and is not guaranteed to converge to the optimal perturbation in (3.1). However, we have observed in practice that our algorithm yields very small perturbations which are believed to be good approximations of the minimal perturbation.

Algorithm 2 DeepFool: multi-class case

input: Image $\boldsymbol{x}$, classifier f.
output: Perturbation $\hat{\boldsymbol{r}}$.

Initialize $\boldsymbol{x}_0 \leftarrow \boldsymbol{x}$, $i \leftarrow 0$.
while $\hat{k}(\boldsymbol{x}_i) = \hat{k}(\boldsymbol{x}_0)$ **do**
    for $k \neq \hat{k}(\boldsymbol{x}_0)$ **do**
        $\boldsymbol{w}'_k \leftarrow \nabla f_k(\boldsymbol{x}_i) - \nabla f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_i)$
        $f'_k \leftarrow f_k(\boldsymbol{x}_i) - f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_i)$
    end for
    $\hat{l} \leftarrow \arg\min_{k \neq \hat{k}(\boldsymbol{x}_0)} \frac{|f'_k|}{\|\boldsymbol{w}'_k\|_2}$
    $\boldsymbol{r}_i \leftarrow \frac{|f'_{\hat{l}}|}{\|\boldsymbol{w}'_{\hat{l}}\|_2^2} \boldsymbol{w}'_{\hat{l}}$
    $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \boldsymbol{r}_i$
    $i \leftarrow i + 1$
end while
return $\hat{\boldsymbol{r}} = \sum_i \boldsymbol{r}_i$

It should be noted that the optimization strategy of DeepFool is strongly tied to existing optimization techniques. In the binary case, it can be seen as Newton's iterative algorithm for finding roots of a nonlinear system of equations in the underdetermined case [80]. This algorithm is known as the normal flow method. The convergence analysis of this optimization technique can be found for example in [94]. Our algorithm in the binary case can alternatively be seen as a gradient descent algorithm with an adaptive step size

that is automatically chosen at each iteration. The linearization in Algorithm 2 is also similar to a sequential convex programming where the constraints are linearized at each step.

3.2.3 Experiments

Setup

We now test our DeepFool algorithm on deep convolutional neural networks architectures applied to MNIST, CIFAR-10, and ImageNet image classification datasets. We consider the following deep neural network architectures:

- **MNIST:** A two-layer fully connected network, and a two-layer LeNet convolutional neural network architecture [53]. Both networks are trained with SGD with momentum using the MatConvNet [93] package.
- **CIFAR-10:** We trained a three-layer LeNet architecture, as well as a Network In Network (NIN) architecture [56].
- **ILSVRC 2012:** We used CaffeNet [44] and GoogLeNet [85] pre-trained models.

In order to evaluate the robustness to adversarial perturbations of a classifier f, we compute the average robustness $\hat{\rho}_{\text{adv}}(f)$, defined by

$$\hat{\rho}_{\text{adv}}(f) = \frac{1}{|\mathscr{D}|} \sum_{\boldsymbol{x} \in \mathscr{D}} \frac{\|\hat{\boldsymbol{r}}(\boldsymbol{x})\|_2}{\|\boldsymbol{x}\|_2}, \tag{3.10}$$

where $\hat{\boldsymbol{r}}(\boldsymbol{x})$ is the estimated minimal perturbation obtained using DeepFool, and $\mathscr{D}$ denotes the test set[3].

We compare the proposed DeepFool approach to state-of-the-art techniques to compute adversarial perturbations in [86] and [35]. The method in [86] solves a series of penalized optimization problems to find the minimal perturbation, whereas [35] estimates the minimal perturbation by taking the sign of the gradient

$$\hat{\boldsymbol{r}}(\boldsymbol{x}) = \epsilon \operatorname{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right),$$

with J the cost used to train the neural network, $\boldsymbol{\theta}$ is the model parameters, and y is the label of $\boldsymbol{x}$. The method is called Fast Gradient Sign Method (FGSM). In practice, in the absence of general rules to choose the parameter ϵ, we chose the smallest ϵ such that 90% of the data are misclassified after perturbation.[4]

[3] For ILSVRC2012, we used the validation data.

[4] Using this method, we observed empirically that one cannot reach 100% misclassification rate on

Table 3.1 – The adversarial robustness of different classifiers on different datasets. The time required to compute one sample for each method is given in the time columns. The times are computed on a Mid-2015 MacBook Pro without CUDA support. The asterisk marks determines the values computed using a GTX 750 Ti GPU.

Classifier	Test error	$\hat{\rho}_{\text{adv}}$ (Ours)	time	$\hat{\rho}_{\text{adv}}$ [35]	time	$\hat{\rho}_{\text{adv}}$ [86]	time
MNIST							
LeNet	1%	2.0×10^{-1}	110 ms	1.0	20 ms	2.5×10^{-1}	> 4 s
FC500-150-10	1.7%	1.1×10^{-1}	50 ms	3.9×10^{-1}	10 ms	1.2×10^{-1}	> 2 s
CIFAR-10							
NIN	11.5%	2.3×10^{-2}	1100 ms	1.2×10^{-1}	180 ms	2.4×10^{-2}	>50 s
LeNet	22.6%	3.0×10^{-2}	220 ms	1.3×10^{-1}	50 ms	3.9×10^{-2}	>7 s
ImageNet							
CaffeNet	42.6%	2.7×10^{-3}	510 ms*	3.5×10^{-2}	50 ms*	-	-
GoogLeNet	31.3%	1.9×10^{-3}	800 ms*	4.7×10^{-2}	80 ms*	-	-

Results

We report in Table 3.1 the accuracy and average robustness $\hat{\rho}_{\text{adv}}$ of each classifier computed using different methods. We also show the running time required for each method to compute *one* adversarial sample.

It can be seen that DeepFool estimates smaller perturbations (hence closer to minimal perturbation defined in (3.1)) than the ones computed using the competitive approaches. For example, the average perturbation obtained using DeepFool is 5 times lower than the one estimated with [35]. On the ILSVRC2012 challenge dataset, the average perturbation is one order of magnitude smaller compared to the fast gradient method. It should be noted moreover that the proposed approach also yields slightly smaller perturbation vectors than the method in [86]. The proposed approach is hence more accurate in detecting directions that can potentially fool neural networks. As a result, DeepFool can be used as a valuable tool to accurately assess the robustness of classifiers. On the cost aspect, the proposed approach is substantially faster than the standard method proposed in [86]. In fact, while the approach [86] involves a costly minimization of a series of objective functions, we observed empirically that DeepFool converges in a few iterations (i.e., less than 3) to a perturbation vector that fools the classifier. Hence, the proposed approach reaches a more accurate perturbation vector compared to state-of-the-art methods, while being computationally efficient. This makes it readily suitable to be used as a baseline method to estimate the robustness of very deep neural networks on large-scale datasets. In that context, we provide the first quantitative evaluation of the

some datasets. In fact, even by increasing ϵ to be very large, this method can fail in misclassifying all samples.

robustness of state-of-the-art classifiers on the large-scale ImageNet dataset. It can be seen that despite their very good test accuracy, these methods are extremely unstable to adversarial perturbations: a perturbation that is 1000 smaller in magnitude than the original image is sufficient to fool state-of-the-art deep neural networks.

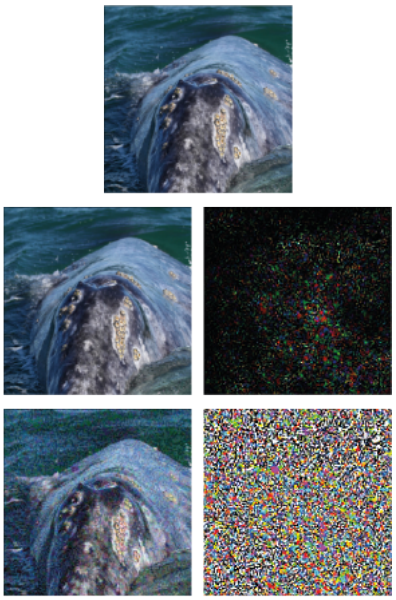


Figure 3.5 – An example of adversarial perturbations. First row: the original image $\boldsymbol{x}$ that is classified as $\hat{k}(\boldsymbol{x})$="whale". Second row: the image $\boldsymbol{x}+\boldsymbol{r}$ classified as $\hat{k}(\boldsymbol{x}+\boldsymbol{r})$="turtle" and the corresponding perturbation $\boldsymbol{r}$ computed by DeepFool. Third row: the image classified as "turtle" and the corresponding perturbation computed by the fast gradient sign method (FGMS) [35]. DeepFool leads to a smaller perturbation.

We illustrate in Fig. 3.5 perturbed images generated by the fast gradient sign and DeepFool. It can be observed that the proposed method generates adversarial perturbations which are hardly perceptible, while the fast gradient sign method outputs a perturbation image with higher norm.

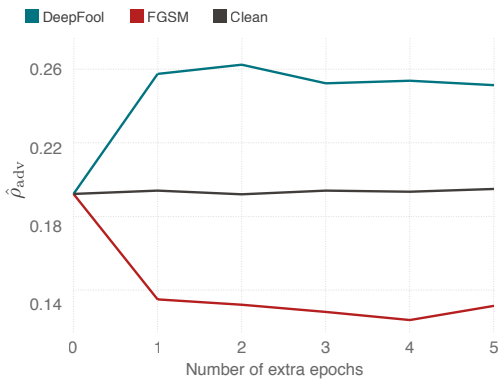
It should be noted that, when perturbations are measured using the ℓ_∞ norm, the above conclusions remain unchanged: DeepFool yields adversarial perturbations that are smaller (hence closer to the optimum) compared to other methods for computing adversarial examples. Table 3.2 reports the ℓ_∞ robustness to adversarial perturbations measured by $\hat{\rho}^{\infty}_{\text{adv}}(f) = \frac{1}{|\mathscr{D}|}\sum_{\boldsymbol{x}\in\mathscr{D}} \frac{\|\hat{\boldsymbol{r}}(\boldsymbol{x})\|_\infty}{\|\boldsymbol{x}\|_\infty}$, where $\hat{\boldsymbol{r}}(\boldsymbol{x})$ is computed respectively using DeepFool (with $p=\infty$, see Section 3.4.1), and the Fast gradient sign method for MNIST and CIFAR-10 tasks.

Table 3.2 – Values of $\hat{\rho}^{\infty}_{\text{adv}}$ for four different networks based on DeepFool (smallest l_∞ perturbation) and fast gradient sign method with 90% of misclassification.

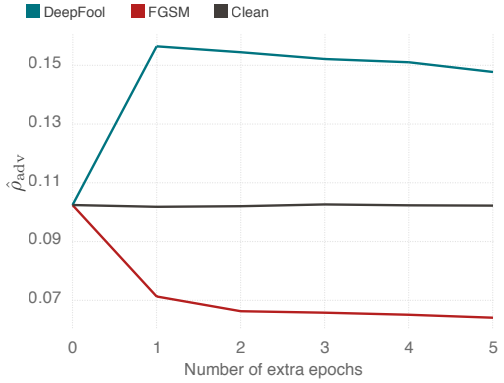
Classifier	DeepFool	FGSM [35]
MNIST		
LeNet	0.10	0.26
FC500-150-10	0.04	0.11
CIFAR-10		
NiN	0.008	0.024
LeNet	0.015	0.028

3.3 Adversarial training using DeepFool

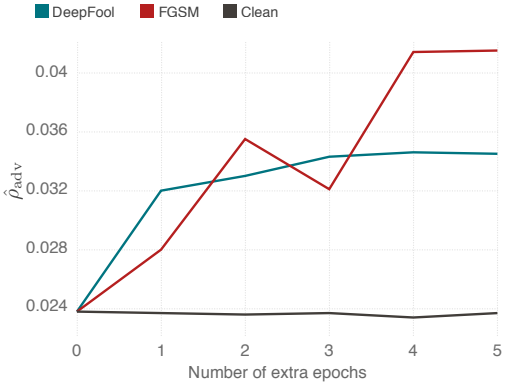
In this section, we fine-tune the networks of Table 3.1 on adversarial examples to build more robust classifiers for the MNIST and CIFAR-10 tasks. Specifically, for each network, we performed two experiments: (i) Fine-tuning the network on DeepFool's adversarial examples, (ii) Fine-tuning the network on the fast gradient sign adversarial examples. We fine-tune the networks by performing 5 additional epochs, with a 50% decreased learning rate only on the perturbed training set. For each experiment, the same training data was used through all 5 extra epochs. For the sake of completeness, we also performed 5 extra epochs on the original data. The evolution of $\hat{\rho}_{\text{adv}}$ for the different fine-tuning strategies is shown in Figures 3.6a to 3.6d, *where the robustness $\hat{\rho}_{adv}$ is estimated using DeepFool*, since this is the most accurate method, as shown in Table 3.1. Observe that fine-tuning with DeepFool adversarial examples significantly increases the robustness of the networks to adversarial perturbations even after one extra epoch. For example, the robustness of the networks on MNIST is improved by 50% and NIN's robustness is increased by about 40%. On the other hand, quite surprisingly, the method in [35] can lead to *a decreased* robustness to adversarial perturbations of the network. We hypothesize that this behavior is due to the fact that perturbations estimated using the fast gradient sign method are much larger than minimal adversarial perturbations. Fine-tuning the network with overly perturbed images decreases the robustness of the networks to adversarial perturbations. To verify this hypothesis, we compare in Fig. 3.7 the adversarial robustness of a network that is fine-tuned with the adversarial examples obtained using DeepFool, where norms of perturbations have been deliberately multiplied by $\alpha = 1, 2, 3$. Interestingly, we see that by magnifying the norms of the adversarial perturbations, the robustness of the fine-tuned network is *decreased*. This might explain why overly perturbed images decrease the robustness of MNIST networks: these perturbations can really change the class of the digits, hence fine-tuning based on these examples can lead to a drop of the robustness (for an illustration, see Fig. 3.8). This lends credence to our hypothesis, and further shows



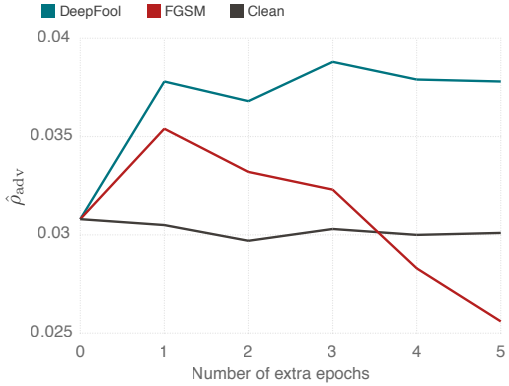
(a) Effect of fine-tuning on adversarial examples computed by two different methods for LeNet on MNIST.



(b) Effect of fine-tuning on adversarial examples computed by two different methods for a fully-connected network on MNIST.



(c) Effect of fine-tuning on adversarial examples computed by two different methods for NIN on CIFAR-10.



(d) Effect of fine-tuning on adversarial examples computed by two different methods for LeNet on CIFAR-10.

Figure 3.6

the importance of designing accurate methods to compute minimal perturbations.

Table 3.3 lists the accuracies of the fine-tuned networks. It can be seen that fine-tuning with DeepFool can improve the accuracy of the networks. Conversely, fine-tuning with the approach in [35] has led to a *decrease* of the test accuracy in all our experiments. This confirms the explanation that the fast gradient sign method outputs *overly perturbed* images that lead to images that are unlikely to occur in the test data. Hence, it *decreases* the performance of the method as it acts as a regularizer that does not represent the distribution of the original data. This effect is analogous to geometric data augmentation schemes, where *large* transformations of the original samples have a counter-productive effect on generalization.[5]

[5]While the authors of [35] reported an *increased* generalization performance on the MNIST task (from 0.94% to 0.84%) using adversarial regularization, it should be noted that the their experimental setup is significantly different as [35] trained the network based on a modified cost function, while we performed straightforward fine-tuning.

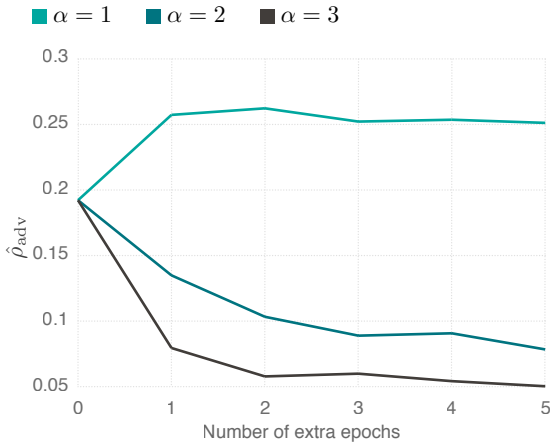


Figure 3.7 – Fine-tuning based on magnified DeepFool's adversarial perturbations.

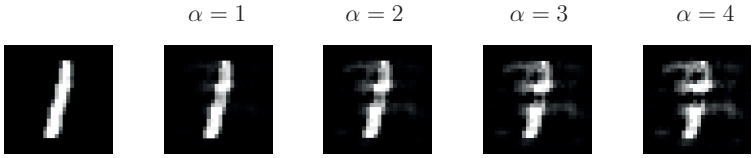


Figure 3.8 – From "1" to "7" : original image classified as "1" and the DeepFool perturbed images classified as "7" using different values of α.

Table 3.3 – The test error of networks after the fine-tuning on adversarial examples (after five epochs). Each columns correspond to a different type of augmented perturbation.

Classifier	DeepFool	FGSM [35]	Baseline
MNIST			
LeNet	0.8%	4.4%	1%
FC500-150-10	1.5%	4.9%	1.7%
CIFAR-10			
NIN	11.2%	21.2%	11.5%
LeNet	20.0%	28.6%	22.6%

The importance of minimal perturbations

To emphasize the importance of an accurate estimation of the minimal perturbation, we now show that using approximate methods can lead to wrong conclusions regarding the adversarial robustness of networks. We fine-tune the NiN classifier on the fast gradient sign adversarial examples. We follow the procedure described earlier but this time, we decreased the learning rate by 90%. We have evaluated the adversarial robustness of this

network at different extra epochs using DeepFool and the *fast gradient sign method*. As one can see in Fig. 3.9, the red plot exaggerates the effect of training on the adversarial examples. Moreover, it is not sensitive enough to demonstrate the loss of robustness at the first extra epoch. These observations confirm that using an *accurate* tool to measure the robustness of classifiers is crucial to derive conclusions about the robustness of deep networks.

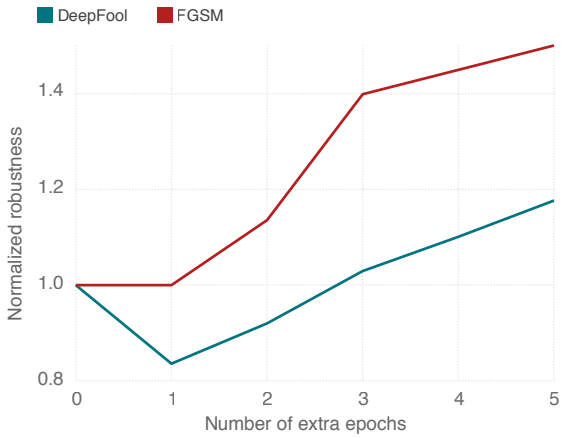


Figure 3.9 – How the adversarial robustness is judged by different methods. Both plots correspond to a network fine-tuned on DeepFool adversarial examples, however, its robustness is evaluated differently using DeepFool and FGSM. The values are normalized by the corresponding $\hat{\rho}_{\text{adv}}$s of the original network.

3.4 Variants of DeepFool

3.4.1 Extending DeepFool to ℓ_p norm

In this chapter, we have measured the perturbations using the ℓ_2 norm. Our framework is however not limited to this choice, and the proposed algorithm can simply be adapted to find minimal adversarial perturbations for any ℓ_p norm ($p \in [1, \infty)$). To do so, the update steps in line 10 and 11 in Algorithm 2 must be respectively substituted by the following updates

$$\hat{l} \leftarrow \arg\min_{k \neq \hat{k}(\boldsymbol{x}_0)} \frac{|f'_k|}{\|\boldsymbol{w}'_k\|_q}, \tag{3.11}$$

$$\boldsymbol{r}_i \leftarrow \frac{|f'_{\hat{l}}|}{\|\boldsymbol{w}'_{\hat{l}}\|_q^q} |\boldsymbol{w}'_{\hat{l}}|^{q-1} \odot \text{sign}(\boldsymbol{w}'_{\hat{l}}), \tag{3.12}$$

where $\odot$ is the pointwise product and $q = \frac{p}{p-1}$.[6] In particular, when $p = \infty$ (i.e., the supremum norm ℓ_∞), these update steps become

$$\hat{l} \leftarrow \arg\min_{k \neq \hat{k}(\boldsymbol{x}_0)} \frac{|f'_k|}{\|\boldsymbol{w}'_k\|_1}, \tag{3.13}$$

$$\boldsymbol{r}_i \leftarrow \frac{|f'_{\hat{l}}|}{\|\boldsymbol{w}'_{\hat{l}}\|_1} \text{sign}(\boldsymbol{w}'_{\hat{l}}). \tag{3.14}$$

3.4.2 Subspace-constrained DeepFool

The DeepFool algorithm developed in the previous sections can be used to find adversarial perturbations confined in a low dimensional subspace. Let $\mathcal{S}$ be an arbitrary subspace of $\mathbb{R}^d$ of dimension m. Define $\boldsymbol{r}^*_{\mathcal{S}}$ to be the perturbation in $\mathcal{S}$ of minimal ℓ_2-norm that is required to change the estimated label of k at $\boldsymbol{x}_0$:

$$r^*_{\mathcal{S}}(\boldsymbol{x}_0) = \arg\min_{\boldsymbol{r} \in \mathcal{S}} \|\boldsymbol{r}\|_2 \text{ subject to } \hat{k}(\boldsymbol{x}_0 + \boldsymbol{r}) \neq \hat{k}(\boldsymbol{x}_0). \tag{3.15}$$

We recall that Algorithm 2 iteratively linearizes the classifier's decision function. Therefore, one can extend it, in the case where the perturbations are confined in a subspace $\mathcal{S}$ of dimension m, by modifying Eq. 3.7 and 3.8 into

$$\boldsymbol{r}^*_{\mathcal{S}}(\boldsymbol{x}_0) = \frac{\left|f_{k^*}(\boldsymbol{x}_0) - f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_0)\right|}{\left\|\mathbf{P}_{\mathcal{S}}\boldsymbol{w}_{k^*} - \mathbf{P}_{\mathcal{S}}\boldsymbol{w}_{\hat{k}(\boldsymbol{x}_0)}\right\|_2^2} \left(\mathbf{P}_{\mathcal{S}}\boldsymbol{w}_{k^*} - \mathbf{P}_{\mathcal{S}}\boldsymbol{w}_{\hat{k}(\boldsymbol{x}_0)}\right), \tag{3.16}$$

where k^* satisfies

$$k^* = \arg\min_{k} \frac{\left|f_k(\boldsymbol{x}_0) - f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_0)\right|}{\left\|\mathbf{P}_{\mathcal{S}}\boldsymbol{w}_k - \mathbf{P}_{\mathcal{S}}\boldsymbol{w}_{\hat{k}(\boldsymbol{x}_0)}\right\|_2}. \tag{3.17}$$

Here, $\mathbf{P}_{\mathcal{S}}$ is the orthogonal projector onto the subspace $\mathcal{S}$. The resulting algorithm is provided in Algorithm 3. Similarly to Eq. (3.10), one can define the robustness in a given subspace as:

$$\hat{\rho}^{\mathcal{S}}_{\text{adv}} = \frac{1}{|\mathscr{D}|} \sum_{\boldsymbol{x} \in \mathscr{D}} \frac{\|\hat{\boldsymbol{r}}_{\mathcal{S}}(\boldsymbol{x})\|_2}{\|\boldsymbol{x}\|_2}. \tag{3.18}$$

We note that, while the full gradients of the classification functions f_k with respect to the input are required in order to compute the unconstrained perturbation (namely, when $\mathcal{S} = \mathbb{R}^d$), we only require the projections of the gradients onto the subspace $\mathcal{S}$ for estimating

[6] To see this, one can apply Holder's inequality to obtain a lower bound on the ℓ_p norm of the perturbation.

Algorithm 3 Computing minimal perturbation in a subspace $\mathcal{S}$

input: Image $\boldsymbol{x}$, classifier f, orthogonal projector $\mathbf{P}_{\mathcal{S}}$ onto $\mathcal{S}$.
output: Perturbation $\hat{\boldsymbol{r}}_{\mathcal{S}}$.
Initialize $\boldsymbol{x}_0 \leftarrow \boldsymbol{x}$, $i \leftarrow 0$.
while $\hat{k}(\boldsymbol{x}_i) = \hat{k}(\boldsymbol{x}_0)$ **do**
    for $k \neq \hat{k}(\boldsymbol{x}_0)$ **do**
        $\boldsymbol{w}'_k \leftarrow \nabla f_k(\boldsymbol{x}_i) - \nabla f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_i)$
        $f'_k \leftarrow f_k(\boldsymbol{x}_i) - f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_i)$
    end for
    $k^* \leftarrow \arg\min_{k \neq \hat{k}(\boldsymbol{x}_0)} \frac{|f'_k|}{\|\mathbf{P}_{\mathcal{S}} \boldsymbol{w}'_k\|_2}$
    $\boldsymbol{r}_i \leftarrow \frac{|f'_{k^*}|}{\|\mathbf{P}_{\mathcal{S}} \boldsymbol{w}'_{k^*}\|_2^2} \mathbf{P}_{\mathcal{S}} \boldsymbol{w}'_{k^*}$
    $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \boldsymbol{r}_i$
    $i \leftarrow i + 1$
end while
return $\hat{\boldsymbol{r}}_{\mathcal{S}} = \sum_i \boldsymbol{r}_i$

the subspace perturbation in Algorithm 3. Informally speaking, the computation of the subspace constrained perturbation requires less information about the classifier. This property, especially when $m \ll d$, can be exploited to design computationally efficient black-box attacks.

Application I: spectral robustness

Subspace constrained perturbations can be used to study the robustness properties of deep networks to band-limited adversarial perturbations, where the subspace constitutes of 2-dimensional Discrete Cosine Transform (DCT) basis vectors. Let $\boldsymbol{s}_{ij}$ denote the vectorized version of the inverse DCT transform of an image of a delta function located at the position (i, j) of the image. The span of a subset of $\boldsymbol{s}_{ij}$s (for diffferent values of i and j) forms a subspace $\mathcal{S}$. For an image of dimensions 224×224, in order to study the robustness of deep networks in different frequency bands, we choose subspaces $\mathcal{S}_k$ as:

$$\mathcal{S}_k = \text{span}\left\{\boldsymbol{s}_{ij} : \left\lfloor \frac{i}{10} \right\rfloor, \left\lfloor \frac{j}{10} \right\rfloor = k; i, j \leq 224\right\}, \tag{3.19}$$

where each $\mathcal{S}_q$ corresponds to a frequency band, and higher q indicates higher frequencies. An example of the resulted perturbations for three different subspaces $\mathcal{S}_0$, $\mathcal{S}_{10}$, and $\mathcal{S}_{20}$ are depicted in Fig. 3.10. For each case, the algorithm misclassified the original image (Polaroid Camera) as a Dial Phone. We observe that for the low frequency subspace $\mathcal{S}_0$, the perturbation is imperceptible, and thus the robustness seems to be low. For the case of the middle frequency subspace $\mathcal{S}_{10}$, we see a slight difference in the texture of the picture, which indicates that the robustness is increasing while we move away from the low frequencies. This observation is justified by the last, high frequency subspace $\mathcal{S}_{20}$,

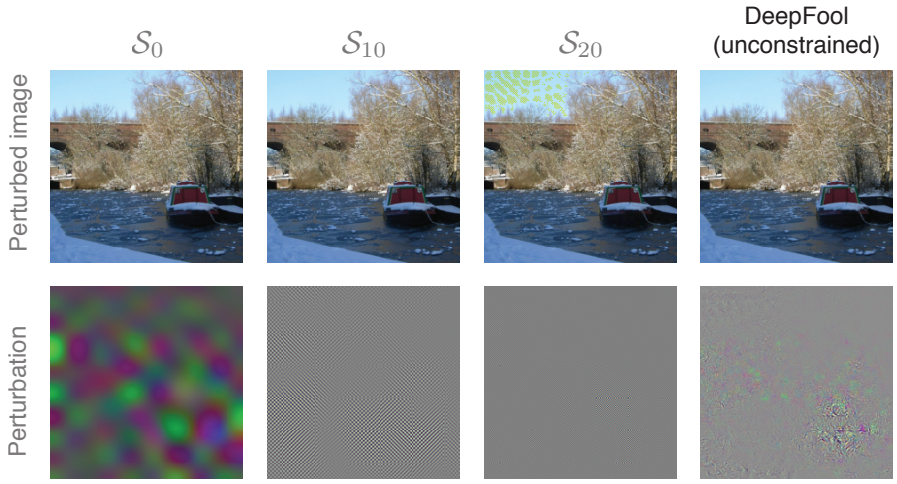


Figure 3.10 – The original image is classified as "boat house" while all three perturbed images in the first row are classified as "jeep". The second row shows the corresponding perturbations.

where the perturbation is perceptible and thus the robustness is higher. One can quantify the previous observation by computing the average robustness (over multiple images) for each $\mathcal{S}_q$. To do so, we define the following measure:

$$\gamma = \sqrt{\frac{m}{d}} \frac{\hat{\rho}_{\text{adv}}^{\mathcal{S}}}{\hat{\rho}_{\text{adv}}}. \tag{3.20}$$

For a randomly chosen subspace $\mathcal{S}$ (see Chapter 5), the value of γ should be close to 1. We compute γ for 1000 randomly selected images from the ImageNet validation set on a pre-trained ResNet-50 network. As seen in Fig. 3.11, the average robustness for low frequency perturbations is quite low, while it increases as the perturbations live in higher frequency bands, which justifies our previous qualitative observation. Therefore, surprisingly, deep networks trained on natural images seem to be really robust to high frequency perturbations. This behaviour is in contrast with the defense mechanisms which try to remove adversarial perturbations by treating them as high frequency noise [19, 37].

Application II: watermarking

Algorithm 3 can be used to generate structured additive adversarial perturbations. As an example, we now show a simple demonstration of the vulnerability of classifiers to subspace adversarial perturbations in Fig. 3.12, where a structured message is hidden in the image and causes data misclassification. Specifically, we consider $\mathcal{S}$ to be the span of random translated and scaled versions of words "NIPS", "SPAIN" and "2016" in an image, such that $\lfloor d/m \rfloor = 228$. The resulting perturbations in the subspace are therefore linear combinations of these words with different intensities. The perturbed image $\boldsymbol{x}_0 + \boldsymbol{r}_{\mathcal{S}}^*$ shown in Fig. 3.12 (c) is clearly indistinguishable from Fig. 3.12 (a). This

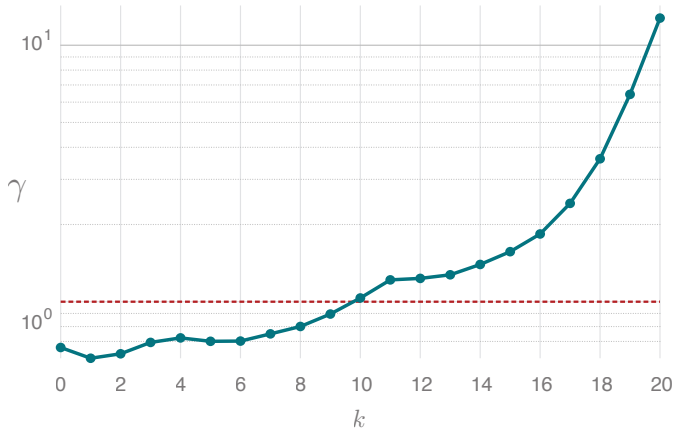


Figure 3.11 – The values of γ with respect to the subspaces $\mathcal{S}_q$ for a ResNet-50 network trained on the ImageNet dataset. The dashed line corresponds to the expected value of γ for a random subspace of dimension q (see Chapter 5).

(a) Classified as "Potflower" (b) Perturbation (c) Classified as "Pineapple"

Figure 3.12 – A fooling hidden message. $\mathcal{S}$ is the span of random translations and scales of the words "NIPS", "SPAIN", and "2016".

shows that imperceptibly small structured messages can be added to an image to cause misclassification.

The potential of Algorithm 3 extends beyond crafting adversarial attacks. In Chapter 5, by means of the so-called *semi-random noise* regime, we show how this algorithm can be employed to study the geometry of the decision boundary of deep neural networks.

3.5 Conclusion

In this chapter, we proposed an algorithm, DeepFool, to compute adversarial examples that fool state-of-the-art image classifiers. It is based on an iterative linearization of the classifier to generate minimal perturbations that are sufficient to change classification labels. We provided extensive experimental evidence on three datasets and eight classifiers, showing the superiority of the proposed method over state-of-the-art methods to compute adversarial perturbations, as well as the efficiency of the proposed approach. Thanks to

its accurate estimation of the adversarial perturbations, the proposed DeepFool algorithm provides an efficient and accurate way to evaluate the robustness of classifiers and to enhance their performance by proper fine-tuning.

We demonstrated the flexibility of our proposed algorithm by introducing two simple yet efficient modifications of DeepFool: ℓ_p-DeepFool and subspace-DeepFool algorithms. Using subspace-DeepFool, we particularly showed that deep networks are surprisingly more vulnerable to lower frequency adversarial perturbations. Furthermore, DeepFool can successfully be used in generating highly structured perturbation vectors. The accuracy and the computational efficiency of DeepFool permits us to use it, in the rest of this thesis, as an essential tool to characterise the geometric properties of the robustness of deep networks.

We finally note that DeepFool has widely been used as a method to evaluate the robustness of classifiers. Furthermore, according to independent benchmarks [77], it has been shown to outperform other adversarial attack methods in computing minimal ℓ_2-norm adversarial perturbations on large-scale datasets.

One intriguing feature of adversarial perturbations computed using DeepFool and other adversarial attack methods is that they can to some extent be transferred across different architectures [86]. However, these input-dependent perturbations do not necessarily transfer well across different images. The question of the existence of input-independent perturbations thus remains open. Hence in the next chapter, we investigate this question and provide an iterative algorithm based on DeepFool to find image-independent adversarial perturbations, which significantly generalise across different images and even different architectures.

4 Universal adversarial perturbations

"Real knowledge is to know the extent of one's ignorance."
— Confucius

4.1 Introduction

In the previous chapter, we proposed proposed a fast but computationally efficient method to generate minimal adversarial perturbations fooling state-of-the-art classifiers. While such perturbations are principally sought for a specific classifier, they can surprisingly fool other classifiers trained on the same dataset [86]. In this chapter, we are however interested to see to what extent adversarial perturbations can become independent from the input. In other words, can we find a *single* small image perturbation that fools a state-of-the-art deep neural network classifier on all natural images?

We show here the existence of quasi-imperceptible *universal* perturbation vectors that lead to misclassify natural images with high probability. Specifically, by adding such a *quasi-imperceptible* perturbation to natural images, the label estimated by the deep neural network is changed with high probability (see Fig. 4.1). Such perturbations are dubbed *universal*, as they are image-agnostic. The existence of these perturbations is problematic when the classifier is deployed in real-world (and possibly hostile) environments, as they can be exploited by adversaries to break the classifier. Indeed, the perturbation process involves the mere addition of one very small perturbation to all natural images, and can be relatively straightforward to implement by adversaries in real-world environments, while being relatively difficult to detect as such perturbations are very small and thus do not significantly affect data distributions.

To find image-agnostic perturbations, we propose an algorithm which seeks a universal

Part of this chapter has been published in
Seyed-Mohsen Moosavi-Dezfooli*, Alhussein Fawzi*, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. (*: Equal contribution)

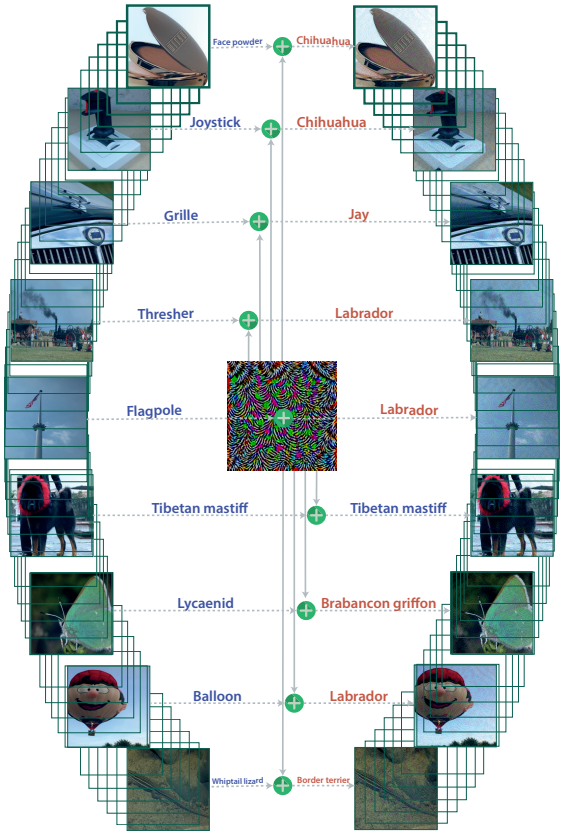


Figure 4.1 – When added to a natural image, a universal perturbation image causes the image to be misclassified by the deep neural network with high probability. *Left images:* Original natural images. The labels are shown on top of each arrow. *Central image:* Universal perturbation. *Right images:* Perturbed images. The estimated labels of the perturbed images are shown on top of each arrow.

perturbation for a set of training points, and proceeds by aggregating atomic perturbation vectors that send successive datapoints to the decision boundary of the classifier. We show their remarkable generalization property, as perturbations computed for a rather small set of training points fool new images with high probability. Furthermore, such perturbations are not only universal across images, but also generalize well across deep neural networks. As a result, these perturbations are therefore *doubly* universal, both with respect to the data and the network architectures.

Universal vs. adversarial perturbations. A fundamental property of adversarial perturbations is their intrinsic dependence on datapoints: the perturbations are specifically crafted for each data point independently. As a result, the computation of an adversarial perturbation for a new data point requires solving a data-dependent optimization problem from scratch, which uses the full knowledge of the classification model. This is different from the universal perturbation considered in this chapter, as we seek a single perturbation

vector that fools the network on most natural images. Perturbing a new datapoint then only involves the mere addition of the universal perturbation to the image (and does not require solving an optimization problem/gradient computation). Finally, we emphasize that our notion of universal perturbation differs from the generalization of adversarial perturbations studied in [86], where perturbations computed on the MNIST task were shown to generalize well across different models. Instead, we examine the existence of universal perturbations that are common to most data points belonging to the data distribution.

The organization of this chapter is as follows: in Section 4.2, we introduce a naive iterative algorithm to find universal perturbations for deep neural networks using only a small subset of training images. In Section 4.3, we empirically analyse some of the key properties of these universal perturbations. Finally in Section 4.5, we study the effectiveness of a simple adversarial training scheme to improve the robustness of classifiers to universal perturbations.

4.2 Universal perturbations for deep networks

We formalize in this section the notion of universal perturbations, and propose a method for estimating such perturbations. Let μ denote a distribution of images in $\mathbb{R}^d$, and $\hat{k}$ define a classification function that outputs for each image $\boldsymbol{x} \in \mathbb{R}^d$ an estimated label $\hat{k}(\boldsymbol{x})$. The main focus is to seek perturbation vectors $\boldsymbol{v} \in \mathbb{R}^d$ that fool the classifier $\hat{k}$ on *almost all* datapoints sampled from μ. That is, we seek a vector $\boldsymbol{v}$ such that

$$\hat{k}(\boldsymbol{x} + \boldsymbol{v}) \neq \hat{k}(\boldsymbol{x}) \text{ for "most" } \boldsymbol{x} \sim \mu.$$

We coin such a perturbation *universal*, as it represents a fixed image-agnostic perturbation that causes label change for most images sampled from the data distribution μ. We focus here on the case where the distribution μ represents the set of natural images, hence containing a huge amount of variability. In that context, we examine the existence of small universal perturbations (in terms of the ℓ_p norm with $p \in [1, \infty)$) that misclassify most images. The goal is therefore to find $\boldsymbol{v}$ that satisfies the following two constraints:

1. $\|\boldsymbol{v}\|_p \leq \xi$,
2. $\underset{x \sim \mu}{\mathbb{P}} \left(\hat{k}(\boldsymbol{x} + \boldsymbol{v}) \neq \hat{k}(\boldsymbol{x}) \right) \geq 1 - \delta$.

The parameter ξ controls the magnitude of the perturbation vector $\boldsymbol{v}$, and δ quantifies the desired fooling rate for all images sampled from the distribution μ.

Algorithm. Let $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$ be a set of images sampled from the distribution μ. Our proposed algorithm seeks a universal perturbation $\boldsymbol{v}$, such that $\|\boldsymbol{v}\|_p \leq \xi$, while fooling most images in X. The algorithm proceeds iteratively over the data in X and

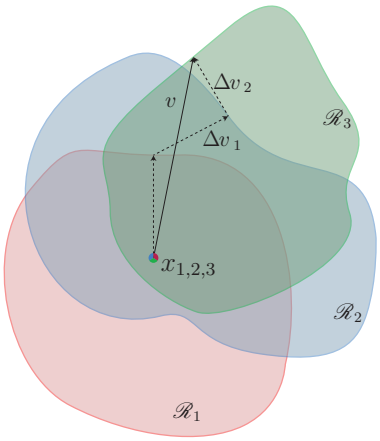


Figure 4.2 – Schematic representation of the proposed algorithm used to compute universal perturbations. In this illustration, data points $\boldsymbol{x}_1, \boldsymbol{x}_2$ and $\boldsymbol{x}_3$ are super-imposed, and the classification regions $\mathscr{R}_i$ (i.e., regions of constant estimated label) are shown in different colors. Our algorithm proceeds by aggregating sequentially the minimal perturbations sending the current perturbed points $\boldsymbol{x}_i + \boldsymbol{v}$ outside of the corresponding classification region $\mathscr{R}_i$.

gradually builds the universal perturbation (see Fig. 4.2). At each iteration, the minimal perturbation $\Delta \boldsymbol{v}_i$ that sends the current perturbed point, $\boldsymbol{x}_i + \boldsymbol{v}$, to the decision boundary of the classifier is computed, and aggregated to the current instance of the universal perturbation. In more details, provided the current universal perturbation $\boldsymbol{v}$ does not fool data point $\boldsymbol{x}_i$, we seek the extra perturbation $\Delta \boldsymbol{v}_i$ with minimal norm that allows to fool data point $\boldsymbol{x}_i$ by solving the following optimization problem:

$$\Delta \boldsymbol{v}_i \leftarrow \arg\min_{\boldsymbol{r}} \|\boldsymbol{r}\|_2 \text{ s.t. } \hat{k}(\boldsymbol{x}_i + \boldsymbol{v} + \boldsymbol{r}) \neq \hat{k}(\boldsymbol{x}_i). \tag{4.1}$$

To ensure that the constraint $\|\boldsymbol{v}\|_p \leq \xi$ is satisfied, the updated universal perturbation is further projected on the ℓ_p ball of radius ξ and centered at 0. That is, let $\mathcal{P}_{p,\xi}$ be the projection operator defined as follows:

$$\mathcal{P}_{p,\xi}(\boldsymbol{v}) = \arg\min_{\boldsymbol{v}'} \|\boldsymbol{v} - \boldsymbol{v}'\|_2 \text{ subject to } \|\boldsymbol{v}'\|_p \leq \xi.$$

Then, our update rule is given by $\boldsymbol{v} \leftarrow \mathcal{P}_{p,\xi}(\boldsymbol{v} + \Delta \boldsymbol{v}_i)$. Several passes on the data set X are performed to improve the quality of the universal perturbation. The algorithm is terminated when the empirical "fooling rate" on the perturbed data set $X_{\boldsymbol{v}} := \{\boldsymbol{x}_1 + \boldsymbol{v}, \ldots, \boldsymbol{x}_m + \boldsymbol{v}\}$ exceeds the target threshold $1 - \delta$. That is, we stop the algorithm whenever $\text{Err}(X_{\boldsymbol{v}}) := \frac{1}{m} \sum_{i=1}^{m} 1_{\hat{k}(\boldsymbol{x}_i + \boldsymbol{v}) \neq \hat{k}(\boldsymbol{x}_i)} \geq 1 - \delta$. The detailed algorithm is provided in Algorithm 4. Interestingly, in practice, the number of data points m in X need not be large to compute a universal perturbation that is valid for the whole distribution μ. In particular, we can set m to be much smaller than the number of training points (see Section 4.3).

Algorithm 4 Computation of universal perturbations.

input: Data points X, classifier $\hat{k}$, desired ℓ_p norm of the perturbation ξ, desired accuracy on perturbed samples δ.
output: Universal perturbation vector $\boldsymbol{v}$.
Initialize $\boldsymbol{v} \leftarrow 0$.
while $\mathrm{Err}(X_{\boldsymbol{v}}) \leq 1 - \delta$ **do**
    for each datapoint $\boldsymbol{x}_i \in X$ **do**
        if $\hat{k}(\boldsymbol{x}_i + \boldsymbol{v}) = \hat{k}(\boldsymbol{x}_i)$ **then**
            Compute the *minimal* perturbation that sends $\boldsymbol{x}_i + \boldsymbol{v}$ to the decision boundary:

$$\Delta \boldsymbol{v}_i \leftarrow \arg\min_{\boldsymbol{r}} \|r\|_2 \text{ s.t. } \hat{k}(\boldsymbol{x}_i + \boldsymbol{v} + \boldsymbol{r}) \neq \hat{k}(\boldsymbol{x}_i).$$

            Update the perturbation:

$$\boldsymbol{v} \leftarrow \mathcal{P}_{p,\xi}(\boldsymbol{v} + \Delta \boldsymbol{v}_i).$$

        end if
    end for
end while

The proposed algorithm involves solving at most m instances of the optimization problem in Eq. (4.1) for each pass. While this optimization problem is not convex when $\hat{k}$ is a standard classifier (e.g., a deep neural network), several efficient approximate methods have been devised for solving this problem [86, 70, 40]. We use in the following the approach in [70] for its efficency. It should further be noticed that the objective of Algorithm 4 is *not* to find the smallest universal perturbation that fools most data points sampled from the distribution, but rather to find one such perturbation with sufficiently small norm. In particular, different random shufflings of the set X naturally lead to a diverse set of universal perturbations $\boldsymbol{v}$ satisfying the required constraints. The proposed algorithm can therefore be leveraged to generate multiple universal perturbations for a deep neural network (see next section for visual examples).

Note that Algorithm 4 can be seen as a stochastic algorithm with a mini-batch size of 1. In general, such algorithms can have a very high variance and they might never converge to a solution. However, as we see in Table 4.1, our proposed algorithm manages to find solutions for state-of-the-art image classifiers.

We now analyze the robustness of state-of-the-art deep neural network classifiers to universal perturbations using Algorithm 4. We assess the estimated universal perturbations for different recent deep neural networks on the ILSVRC 2012 [79] validation set (50,000 images), and report the *fooling ratio*, that is the proportion of images that change labels when perturbed by our universal perturbation. Results are reported for $p = 2$ and $p = \infty$, where we respectively set $\xi = 2000$ and $\xi = 10$. These numerical values were chosen

Table 4.1 – Fooling ratios on the set X, and the validation set.

	CaffeNet [44]	VGG-F [14]	VGG-16 [83]	VGG-19 [83]	GoogLeNet [85]	RN-152 [38]
$p = 2$						
X	85.4%	85.9%	90.7%	86.9%	82.9%	89.7%
Val.	85.6%	87.0%	90.3%	84.5%	82.0%	88.5%
$p = \infty$						
X	93.1%	93.8%	78.5%	77.8%	80.8%	85.4%
Val.	93.3%	93.7%	78.3%	77.8%	78.9%	84.0%

in order to obtain a perturbation whose norm is significantly smaller than the image norms, such that the perturbation is quasi-imperceptible when added to natural images[1]. Results are listed in Table 4.1. Each result is reported on the set X, which is used to compute the perturbation, as well as on the validation set (that is *not* used in the process of the computation of the universal perturbation). Observe that for all networks, the universal perturbation achieves very high fooling rates on the validation set. Specifically, the universal perturbations computed for CaffeNet and VGG-F fool more than 90% of the validation set (for $p = \infty$). In other words, for any natural image in the validation set, the mere addition of our universal perturbation fools the classifier more than 9 times out of 10. This result is moreover not specific to such architectures, as we can also find universal perturbations that cause VGG, GoogLeNet and ResNet classifiers to be fooled on natural images with probability edging 80%.

These results have an element of surprise, as they show the existence of *single* universal perturbation vectors that cause natural images to be misclassified with high probability, albeit being quasi-imperceptible to humans. To verify this latter claim, we show visual examples of perturbed images in Fig. 4.3a, where the GoogLeNet architecture is used. These images are either taken from the ILSVRC 2012 validation set, or captured using a mobile phone camera. Observe that in most cases, the universal perturbation is *quasi-imperceptible*, yet this powerful image-agnostic perturbation is able to misclassify any image with high probability for state-of-the-art classifiers. We visualize the universal perturbations corresponding to different networks in Fig. 4.4.

4.3 Properties of universal perturbations

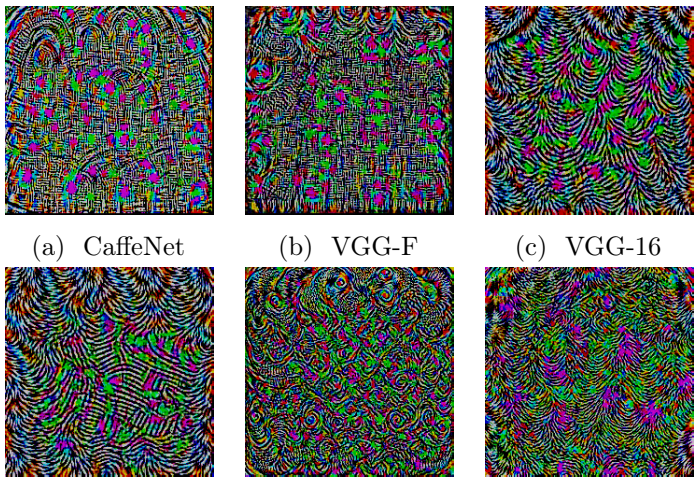
We now analyse some of the properties of universal perturbations generated using Algorithm 4.

[1] For comparison, the average ℓ_2 and ℓ_∞ norm of an image in the validation set is respectively $\approx 5 \times 10^4$ and ≈ 250.

(a) Examples of perturbed images and their corresponding labels. The first 8 images belong to the ILSVRC 2012 validation set, and the last 4 are images taken by a mobile phone camera.

(b) Original images. The first 8 images belong to the ILSVRC 2012 validation set, and the last 4 are images taken by a mobile phone camera.

Figure 4.3



(a) CaffeNet (b) VGG-F (c) VGG-16

(d) VGG-19 (e) GoogLeNet (f) ResNet-152

Figure 4.4 – Universal perturbations computed for different deep neural network architectures. Images generated with $p = \infty$, $\xi = 10$. The pixel values are scaled for visibility.

4.3.1 Diversity of universal perturbations

It should be noted that universal perturbations are not unique, as many different universal perturbations (all satisfying the two required constraints) can be generated for the same network. In Fig. 4.5, we visualize five different universal perturbations obtained by using different random shufflings in X. Observe that such universal perturbations are different, although they exhibit a similar pattern. This is moreover confirmed by computing the normalized inner products between two pairs of perturbation images, as the normalized inner products do not exceed 0.1, which shows that one can find diverse universal perturbations.

Figure 4.5 – Diversity of universal perturbations for the GoogLeNet architecture. The five perturbations are generated using different random shufflings of the set X. Note that the normalized inner products for any pair of universal perturbations does not exceed 0.1, which highlights the diversity of such perturbations.

4.3.2 Effect of the size of training set X

While the above universal perturbations are computed for a set X of 10,000 images from the training set (i.e., in average 10 images per class), we now examine the influence of the size of X on the quality of the universal perturbation. We show in Fig. 4.6 the fooling rates obtained on the validation set for different sizes of X for GoogLeNet. Note for example that with a set X containing only 500 images, we can fool more than 30% of the images on the validation set. This result is significant when compared to the number of classes in ImageNet (1000), as it shows that we can fool a large set of unseen images, even when using a set X containing less than one image per class! The universal perturbations computed using Algorithm 4 have therefore a remarkable generalization power over unseen data points, and can be computed on a very small set of training images.

4.3.3 Cross-model universality

While the computed perturbations are universal across unseen data points, we now examine their *cross-model* universality. That is, we study to which extent universal perturbations computed for a specific architecture (e.g., VGG-19) are also valid for another architecture (e.g., GoogLeNet). Table 4.2 displays a matrix summarizing the universality of such perturbations across six different architectures. For each architecture, we compute a

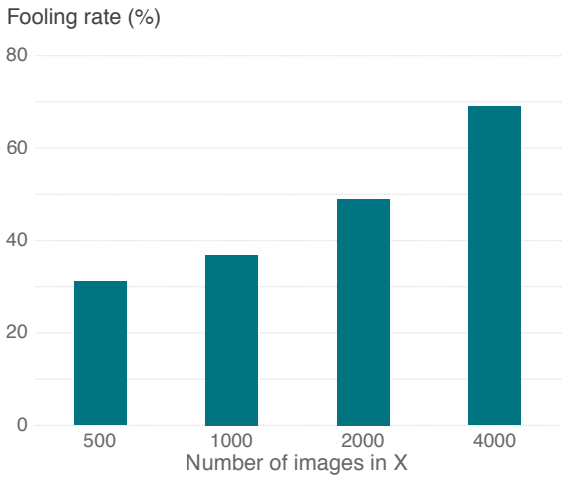


Figure 4.6 – Fooling ratio on the validation set versus the size of X. Note that even when the universal perturbation is computed on a very small set X (compared to training and validation sets), the fooling ratio on validation set is large.

universal perturbation and report the fooling ratios on all other architectures; we report these in the rows of Table 4.2. Observe that, for some architectures, the universal perturbations generalize very well across other architectures. For example, universal perturbations computed for the VGG-19 network have a fooling ratio above 53% for all other tested architectures. This result shows that our universal perturbations are, to some extent, *doubly-universal* as they generalize well across data points *and* very different architectures. It should be noted that, in [86], adversarial perturbations were previously shown to generalize well, to some extent, across different neural networks on the MNIST problem. Our results are however different, as we show the generalizability of universal perturbations across different architectures on the ImageNet data set. This result shows that such perturbations are of practical relevance, as they generalize well across data points and architectures. In particular, in order to fool a new image on an unknown neural network, a simple addition of a universal perturbation computed on the VGG-19 architecture is likely to misclassify the data point.

4.3.4 Visualization of the effect of universal perturbations

To gain insights on the effect of universal perturbations on natural images, we now visualize the distribution of labels on the ImageNet validation set. Specifically, we build a directed graph $G = (V, E)$, whose vertices denote the labels, and directed edges $e = (i \to j)$ indicate that the majority of images of class i are fooled into label j when applying the universal perturbation. The existence of edges $i \to j$ therefore suggests that the preferred fooling label for images of class i is j. We construct this graph for GoogLeNet, and visualize the full graph in Fig. 4.7. The visualization of this graph shows a very peculiar topology. In particular, the graph is a union of disjoint components, where all edges in

Table 4.2 – Generalizability of the universal perturbations across different networks. The percentages indicate the fooling rates. The rows indicate the architecture for which the universal perturbations is computed, and the columns indicate the architecture for which the fooling rate is reported.

	VGG-F	CaffeNet	GoogLeNet	VGG-16	VGG-19	ResNet-152
VGG-F	**93.7%**	71.8%	48.4%	42.1%	42.1%	47.4 %
CaffeNet	74.0%	**93.3%**	47.7%	39.9%	39.9%	48.0%
GoogLeNet	46.2%	43.8%	**78.9%**	39.2%	39.8%	45.5%
VGG-16	63.4%	55.8%	56.5%	**78.3%**	73.1%	63.4%
VGG-19	64.0%	57.2%	53.6%	73.5%	**77.8%**	58.0%
ResNet-152	46.3%	46.3%	50.5%	47.0%	45.5%	**84.0%**

one component mostly connect to one target label. See Fig. 4.8 for an illustration of two connected components. This visualization clearly shows the existence of several *dominant labels*, and that universal perturbations mostly make natural images classified with such labels. We hypothesize that these dominant labels occupy large regions in the image space, and therefore represent good candidate labels for fooling most natural images. Note that these dominant labels are automatically found and are not imposed a priori in the computation of perturbations.

4.4 Comparison with other types of perturbations

The goal of this section is to analyze and explain the high vulnerability of deep neural network classifiers to universal perturbations. To understand the unique characteristics of universal perturbations, we first compare such perturbations with other types of perturbations, namely i) *random* perturbation, ii) *adversarial* perturbation computed for a randomly picked sample (computed using the DF and FGS methods respectively in [70] and [35]), iii) *sum* of adversarial perturbations over X, and iv) mean of the images (or *ImageNet bias*). For each perturbation, we depict a phase transition graph in Fig. 4.9 showing the fooling rate on the validation set with respect to the ℓ_2 norm of the perturbation. Different perturbation norms are achieved by scaling accordingly each perturbation with a multiplicative factor to have the target norm. Note that the universal perturbation is computed for $\xi = 2000$, and also scaled accordingly.

Observe that the proposed universal perturbation quickly reaches very high fooling rates, even when the perturbation is constrained to be of small norm. For example, the universal perturbation computed using Algorithm 4 achieves a fooling rate of 85% when the ℓ_2 norm is constrained to $\xi = 2000$, while other perturbations (e.g., adversarial perturbations) achieve much smaller ratios for comparable norms. In particular, random vectors sampled

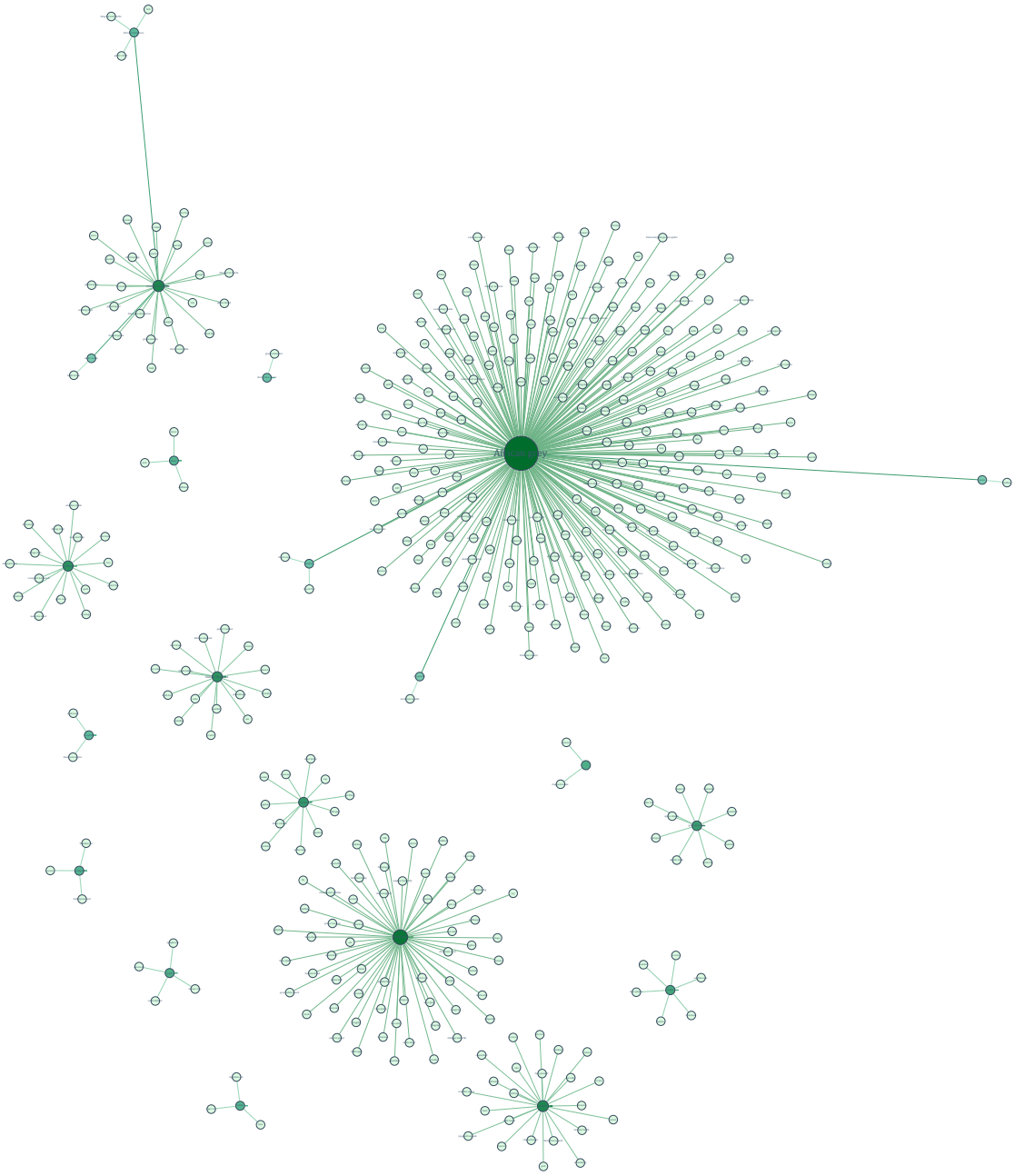


Figure 4.7 – Graph representing the relation between original and perturbed labels. Note that "dominant labels" appear systematically. Please zoom for readability. Isolated nodes are removed from this visualization for readability.

uniformly from the sphere of radius of 2000 only fool 10% of the validation set. The large difference between universal and random perturbations suggests that the universal perturbation exploits some *geometric correlations* between different parts of the decision boundary of the classifier. In fact, if the orientations of the decision boundary in the neighborhood of different data points were completely uncorrelated (and independent

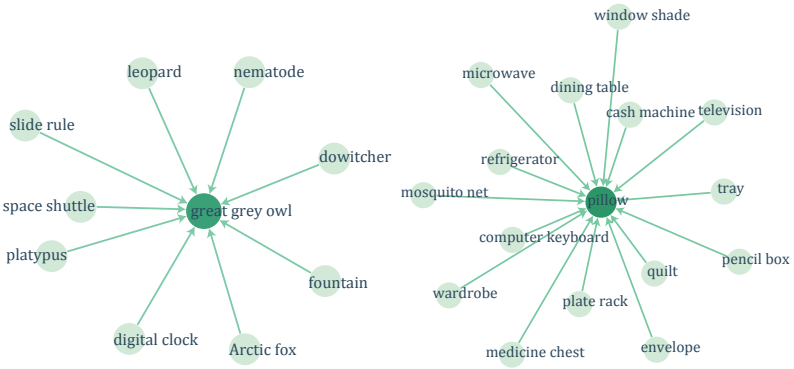


Figure 4.8 – Two connected components of the graph $G = (V, E)$, where the vertices are the set of labels, and directed edges $i \to j$ indicate that most images of class i are fooled into class j.

of the distance to the decision boundary), the norm of the best universal perturbation would be comparable to that of a random perturbation. Note that the latter quantity is well understood (see [29]), as the norm of the random perturbation required to fool a specific data point precisely behaves as $\Theta(\sqrt{d}\|\boldsymbol{r}\|_2)$, where d is the dimension of the input space, and $\|\boldsymbol{r}\|_2$ is the distance between the data point and the decision boundary (or equivalently, the norm of the smallest adversarial perturbation). For the considered ImageNet classification task, this quantity is equal to $\sqrt{d}\|\boldsymbol{r}\|_2 \approx 2 \times 10^4$, for most data points, which is at least one order of magnitude larger than the universal perturbation ($\xi = 2000$). This substantial difference between *random* and *universal* perturbations thereby suggests redundancies in the geometry of the decision boundaries that we further explore in Chapter 6.

4.5 Defense against universal perturbations

We now examine the effect of adversarial training the networks with *universally* perturbed images. We use the VGG-F architecture, and fine-tune the network based on a modified training set where universal perturbations are added to a fraction of (clean) training samples: for each training point, a universal perturbation is added with probability 0.5, and the original sample is preserved with probability 0.5.

We use a slightly modified notion of universal perturbations, where the *direction* of the universal vector $\boldsymbol{v}$ is fixed for all data points, while its *magnitude* is adaptive. That is, for each data point $\boldsymbol{x}$, we consider the perturbed point $\boldsymbol{x} + \alpha\boldsymbol{v}$, where α is the smallest coefficient that fools the classifier. We observed that this feed-backing strategy is less prone to overfitting than the strategy where the universal perturbation is simply added to all training points. To account for the diversity of universal perturbations, we pre-compute a pool of 10 different universal perturbations and add perturbations to the

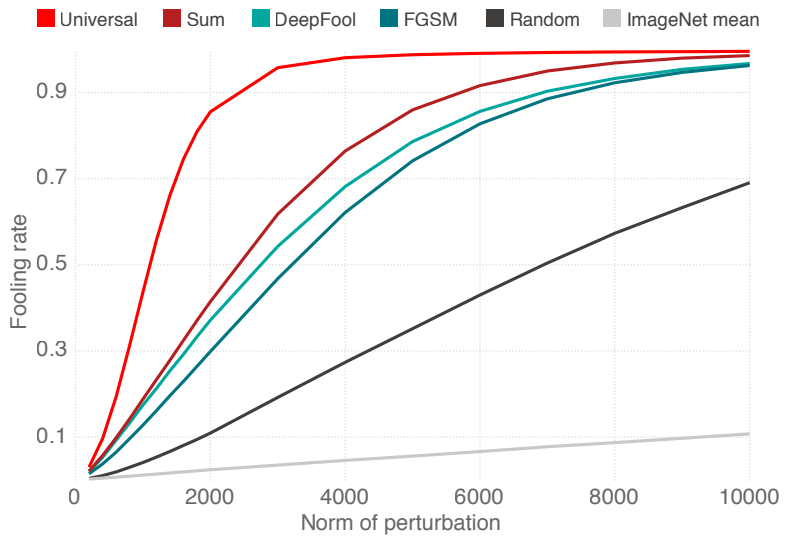


Figure 4.9 – Comparison between fooling rates of different perturbations. Experiments performed on the CaffeNet architecture.

training samples randomly from this pool. The network is fine-tuned by training 5 extra epochs on the modified training set.

To assess the effect of fine-tuning on the robustness of the network, we compute a new universal perturbation for the fine-tuned network (with $p = \infty$ and $\xi = 10$), and report the fooling rate of the network. After 5 extra epochs, the fooling rate on the validation set is 76.2%, which shows an improvement with respect to the original network (93.7%, see Table 4.1).[2] Despite this improvement, the fine-tuned network remains largely vulnerable to small universal perturbations. We therefore repeated the above procedure (i.e., computation of a pool of 10 universal perturbations for the fine-tuned network, fine-tuning of the new network based on the modified training set for 5 epochs), and we obtained a new fooling ratio of 80.0%. In general, the repetition of this procedure for a fixed number of times did *not* yield any improvement over the 76.2% ratio obtained after one step of fine-tuning. Hence, while fine-tuning the network leads to an improvement in the robustness, this simple solution does not fully immune against universal perturbations.

4.6 Conclusions

We showed the existence of small universal perturbations that can fool state-of-the-art classifiers on natural images. We proposed an iterative algorithm to generate universal perturbations, and highlighted several properties of such perturbations. In particular, we showed that universal perturbations generalize well across different classification models,

[2]This fine-tuning procedure moreover led to a minor increase in the error rate on the validation set, which might be due to a slight overfitting of the perturbed data.

resulting in doubly-universal perturbations (image-agnostic, network-agnostic).

Similar to normal adversarial perturbations, universal perturbations are not limited to image classification tasks, and they can be an issue for other visual recognition tasks, such as semantic segmentation [39], and even for non-visual modalities such as textual data [6]. Furthermore, there exist efficient methods to find universal perturbations without requiring any training datapoints [73, 47]. These and other features of such image-independent perturbations make them a real threat for security- and safety-critical applications of deep neural networks.

The existence of universal perturbations contests our understanding of the geometry of the decision boundary of deep networks, and contributes to a better understanding of such systems. Specifically, in Chapter 6, we will provide sufficient geometric conditions for the existence of such image-agnostic perturbations. In particular, we show that the correlation between different regions of the decision boundary partially explains the vulnerability of deep networks to universal perturbations. Furthermore, deep networks with a curved decision boundary can be even more susceptible to such perturbations.

Geometric analysis Part II

5 Geometric analysis with adversarial perturbations

"It is through science that we prove, but through intuition that we discover."
— Henri Poincaré

5.1 Introduction

The existence of extremely small adversarial perturbations causing misclassification of input data, and that they can potentially be transferred well across different models and datapoints, reveal important properties of the decision boundary of deep neural networks. Studying the geometry of classifiers thus helps us understand and explain the underlying reasons of the adversarial vulnerability of deep classifiers, and can lead to building more robust classifiers. Furthermore, the geometric properties of the decision boundary of deep network can be exploited to develop scalable methods in order to assess the adversarial vulnerability of deep classifiers.

In this chapter and the next, we quantify the link between robustness properties of deep networks and the geometry of their decision boundary in the vicinity of data samples.We first point to the inherent connection between adversarial perturbations and the local geometry of the decision boundary of deep networks. We then study the hypothesis that the decision boundary is almost flat in the vicinity of data samples, and how this hypothesis leads to vulnerability to adversarial perturbations. In particular, we support the "flatness" hypothesis by introducing the *semi-random* adversarial perturbations, and derive bounds on the adversarial perturbations of low curvature (locally flat) classifiers.

The chapter is organized as follows. In Section 5.2, we explain the intuitive link between robustness to adversarial perturbations and the local geometry of the decision boundary of classifiers. We introduce the notion of robustness to semi-random noise in Section 5.3. In

Part of this chapter has been published in
Alhussein Fawzi*, Seyed Moosavi-Dezfooli*, and Pascal Frossard. Robustness of classifiers: from adversarial to random noise. In *Neural Information Processing Systems (NIPS)*, 2016.

Section 5.4, we characterize the robustness of linear classifiers to random and semi-random noise, and we then extend it to general non-linear classifiers, provided the curvature of the decision boundary is kept small, in Section 5.5. In Section 5.6, experimental results are presented, where our theoretical results are shown to be accurately satisfied by state-of-the-art deep neural networks on various sets of data.

5.2 Adversarial robustness and the geometry of classifiers

The study of robustness allows us to derive insights about the classifiers, and more precisely about the geometry of the classification function acting on the high dimensional input space. We recall that $f : \mathbb{R}^d \to \mathbb{R}^L$ denotes our L-class classifier, and we denote by $f_1, \ldots, f_L$ the L probabilities associated to each class by the classifier. Specifically, for a given datapoint $x \in \mathbb{R}^d$, the estimated label is obtained by $\hat{k}(\boldsymbol{x}) = \arg\max_k f_k(\boldsymbol{x})$, where $f_k(\boldsymbol{x})$ is the k^{th} component of $f(\boldsymbol{x})$ that corresponds to the k^{th} class. For deep neural networks, the functions f_i represent the outputs of the last layer in the network (generally the softmax layer). Note that the classifier f can be seen as a mapping that partitions the input space $\mathbb{R}^d$ into classification regions, each of which has a constant estimated label (i.e., $\hat{k}(\cdot)$ is constant for each such region). The decision boundary $\mathcal{B}$ of the classifier is defined as the union of the boundaries of such classification regions. *Additive* adversarial perturbations are inherently related with the geometry of the decision boundary. This link relies on the following simple observation

Observation 1 (Geometric interpretation of adversarial perturbation)**.** *Let $\boldsymbol{x} \in \mathbb{R}^d$, and $\boldsymbol{r}^*_{adv}(\boldsymbol{x})$ be the adversarial perturbation, defined as the minimizer of*

$$\boldsymbol{r}^*(\boldsymbol{x}) = \arg\min_{\boldsymbol{r}} \|\boldsymbol{r}\|_p \text{ s.t. } \hat{k}(\boldsymbol{x} + \boldsymbol{r}) \neq \hat{k}(\boldsymbol{x}), \tag{5.1}$$

with $p = 2$. Then, we have:

1. *$\|\boldsymbol{r}^*_{adv}(\boldsymbol{x})\|_2$ measures the Euclidean distance from x to the closest point on the decision boundary $\mathcal{B}$.*
2. *The vector $\boldsymbol{r}^*_{adv}(\boldsymbol{x})$ is orthogonal to the decision boundary of the classifier, at $\boldsymbol{x} + \boldsymbol{r}^*_{adv}(\boldsymbol{x})$.*

These two geometric properties are illustrated in Fig. 5.1. Note that these geometric properties are specific to the ℓ_2 norm. The high instability of classifiers to adversarial perturbations, which we highlighted in the previous chapters, shows that natural images lie very close to the classifier's decision boundary. While this result is key to understanding

Section 5.2 has been published in

Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. The robustness of deep networks: A geometrical perspective. *IEEE Signal Processing Magazine*, 34(6):50–62, 2017.

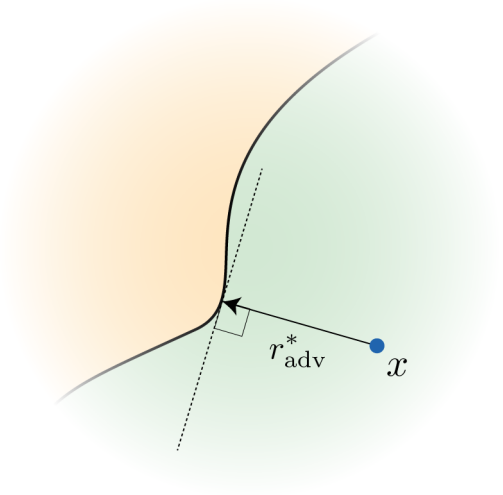


Figure 5.1 – $\boldsymbol{r}^*_{\text{adv}}$ denotes the adversarial perturbation of $\boldsymbol{x}$ (with $p = 2$). Note that $\boldsymbol{r}^*_{\text{adv}}$ is orthogonal to the decision boundary $\mathcal{B}$ and $\|\boldsymbol{r}^*_{\text{adv}}\|_2 = \text{dist}(\boldsymbol{x}, \mathcal{B})$.

the geometry of the datapoints with regards to the classifier's decision boundary, it does not provide any insights on the *shape* of the decision boundary. A local geometric description of the decision boundary (in the vicinity of $\boldsymbol{x}$) is rather captured by the *direction* of $\boldsymbol{r}^*_{\text{adv}}(\boldsymbol{x})$, due to the orthogonality property of adversarial perturbations highlighted in the previous observation. In the rest of this chapter, we introduce a new set of perturbations, called semi-random noise, to study the shape of decision boundary in the vicinity of datapoints. Specifically, the magnitude of such perturbations allow us to infer the average curvature of the decision boundary in a neighborhood of data samples. Moreover, the existence of semi-random perturbations is a strong indication that the space of adversarial examples in the vicinity of datapoints are quite rich, and unlike previous speculations (e.g., in [86]) they are not "blind-spots" of the decision regions of classifiers.

5.3 Semi-random noise regime

In this chapter, we are interested in quantifying the robustness of f with respect to adversarial perturbations confined in a low-dimensional subspace $\mathcal{S}$, as defined in Section 3.4.2 of Chapter 3. It allows us to explore the space of possible adversarial perturbations for a given datapoint. To do so, we define $\boldsymbol{r}^*_{\mathcal{S}}$ to be the perturbation in $\mathcal{S}$ of minimal norm that is required to change the estimated label of f at $\boldsymbol{x}_0$.[1]

$$\boldsymbol{r}^*_{\mathcal{S}}(\boldsymbol{x}_0) = \underset{\boldsymbol{r} \in \mathcal{S}}{\arg\min} \|\boldsymbol{r}\|_2 \text{ s.t. } \hat{k}(\boldsymbol{x}_0 + \boldsymbol{r}) \neq \hat{k}(\boldsymbol{x}_0). \tag{5.2}$$

[1] Perturbation vectors sending a datapoint exactly to the boundary are assumed to change the estimated label of the classifier.

Note that $\boldsymbol{r}^*_{\mathcal{S}}(\boldsymbol{x}_0)$ can be equivalently written

$$\boldsymbol{r}^*_{\mathcal{S}}(\boldsymbol{x}_0) = \arg\min_{\boldsymbol{r}\in\mathcal{S}} \|\boldsymbol{r}\|_2 \text{ s.t. } \exists k \neq \hat{k}(\boldsymbol{x}_0) : f_k(\boldsymbol{x}_0+\boldsymbol{r}) \geq f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_0+\boldsymbol{r}). \tag{5.3}$$

When $\mathcal{S} = \mathbb{R}^d$, $\boldsymbol{r}^*(\boldsymbol{x}_0) := \boldsymbol{r}^*_{\mathbb{R}^d}(\boldsymbol{x}_0)$ is the *adversarial (or worst-case) perturbation* defined in [86], which corresponds to the (unconstrained) perturbation of minimal norm that changes the label of the datapoint $\boldsymbol{x}_0$. In other words, $\|\boldsymbol{r}^*(\boldsymbol{x}_0)\|_2$ corresponds to the minimal distance from $\boldsymbol{x}_0$ to the classifier boundary. In the case where $\mathcal{S} \subset \mathbb{R}^d$, only perturbations along $\mathcal{S}$ are allowed. The robustness of f at $\boldsymbol{x}_0$ along $\mathcal{S}$ is naturally measured by the norm $\|\boldsymbol{r}^*_{\mathcal{S}}(\boldsymbol{x}_0)\|_2$. Different choices for $\mathcal{S}$ permit to study the robustness of f in two different regimes:

- **Random noise regime**: This corresponds to the case where $\mathcal{S}$ is a *one-dimensional subspace* ($m = 1$) with direction $\boldsymbol{v}$, where $\boldsymbol{v}$ is a *random vector* sampled uniformly from the unit sphere $\mathbb{S}^{d-1}$. Writing it explicitly, we study in this regime the robustness quantity defined by $\min_t |t|$ s.t. $\exists k \neq \hat{k}(\boldsymbol{x}_0), f_k(\boldsymbol{x}_0+t\boldsymbol{v}) \geq f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_0+t\boldsymbol{v})$, where $\boldsymbol{v}$ is a vector sampled uniformly at random from the unit sphere $\mathbb{S}^{d-1}$.
- **Semi-random noise regime**: In this case, the subspace $\mathcal{S}$ is chosen *randomly*, but can be of arbitrary dimension m.[2] We use the *semi*-random terminology as the subspace is chosen randomly, and the smallest vector that causes misclassification is then sought in the subspace. It should be noted that the random noise regime is a special case of the semi-random regime with a subspace of dimension $m = 1$. We differentiate nevertheless between these two regimes for clarity.

We first establish relations between the robustness in the random and semi-random regimes on the one hand, and the robustness to adversarial perturbations $\|\boldsymbol{r}^*(\boldsymbol{x}_0)\|_2$ on the other hand. We recall that the latter quantity captures the distance from $\boldsymbol{x}_0$ to the classifier boundary, and is therefore a key quantity in the analysis of robustness.

In the following analysis, we fix $\boldsymbol{x}_0$ to be a datapoint classified as $\hat{k}(\boldsymbol{x}_0)$. To simplify the notation, we remove the explicit dependence on $\boldsymbol{x}_0$ in our notations (e.g., we use $\boldsymbol{r}^*_{\mathcal{S}}$ instead of $\boldsymbol{r}^*_{\mathcal{S}}(\boldsymbol{x}_0)$ and $\hat{k}$ instead of $\hat{k}(\boldsymbol{x}_0)$), and it should be implicitly understood that all our quantities pertain to the fixed datapoint $\boldsymbol{x}_0$.

5.4 Robustness of affine classifiers

We first assume that f is an affine classifier, i.e., $f(\boldsymbol{x}) = \mathbf{W}^\top \boldsymbol{x} + \boldsymbol{b}$ for a given $\mathbf{W} = [\boldsymbol{w}_1 \dots \boldsymbol{w}_L]$ and $\boldsymbol{b} \in \mathbb{R}^L$.

[2] A random subspace is defined as the span of m statistically independent vectors drawn uniformly at random from $\mathbb{S}^{d-1}$.

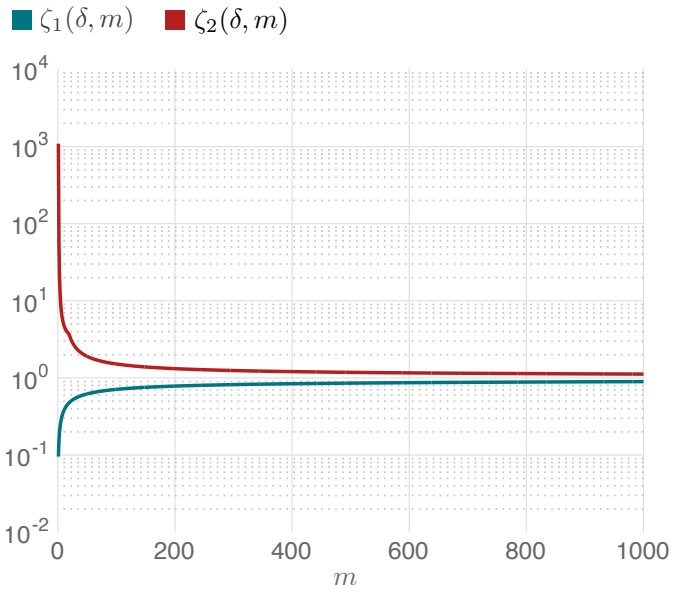


Figure 5.2 – $\zeta_1(m, \delta)$ and $\zeta_2(m, \delta)$ in function of m $[\delta = 0.05]$.

The following result shows a precise relation between the robustness to semi-random noise, $\|\boldsymbol{r}^*_{\mathcal{S}}\|_2$ and the robustness to adversarial perturbations, $\|\boldsymbol{r}^*\|_2$.

Theorem 1. *Let $\delta > 0$, $\mathcal{S}$ be a random m-dimensional subspace of $\mathbb{R}^d$, and f be a L-class affine classifier. Let*

$$\zeta_1(m, \delta) = \left(1 + 2\sqrt{\frac{\ln(1/\delta)}{m}} + \frac{2\ln(1/\delta)}{m}\right)^{-1}, \tag{5.4}$$

$$\zeta_2(m, \delta) = \left(\max\left((1/e)\delta^{2/m}, 1 - \sqrt{2(1 - \delta^{2/m})}\right)\right)^{-1}. \tag{5.5}$$

The following inequalities hold between the robustness to semi-random noise $\|\boldsymbol{r}^*_{\mathcal{S}}\|_2$, and the robustness to adversarial perturbations $\|\boldsymbol{r}^*\|_2$:

$$\sqrt{\zeta_1(m, \delta)}\sqrt{\frac{d}{m}}\|\boldsymbol{r}^*\|_2 \leq \|\boldsymbol{r}^*_{\mathcal{S}}\|_2 \leq \sqrt{\zeta_2(m, \delta)}\sqrt{\frac{d}{m}}\|\boldsymbol{r}^*\|_2, \tag{5.6}$$

with probability exceeding $1 - 2(L + 1)\delta$.

The proof can be found in the appendix. Our upper and lower bounds depend on the functions $\zeta_1(m, \delta)$ and $\zeta_2(m, \delta)$ that control the inequality constants (for m, δ fixed). It should be noted that $\zeta_1(m, \delta)$ and $\zeta_2(m, \delta)$ are independent of the data dimension d. Fig. 5.2 shows the plots of $\zeta_1(m, \delta)$ and $\zeta_2(m, \delta)$ as functions of m, for a fixed δ. It should be noted that for sufficiently large m, $\zeta_1(m, \delta)$ and $\zeta_2(m, \delta)$ are very close to 1 (e.g., $\zeta_1(m, \delta)$ and $\zeta_2(m, \delta)$ belong to the interval $[0.8, 1.3]$ for $m \geq 250$ in the settings of Fig. 5.2). The interval $[\zeta_1(m, \delta), \zeta_2(m, \delta)]$ is however (unavoidably) larger when $m = 1$.

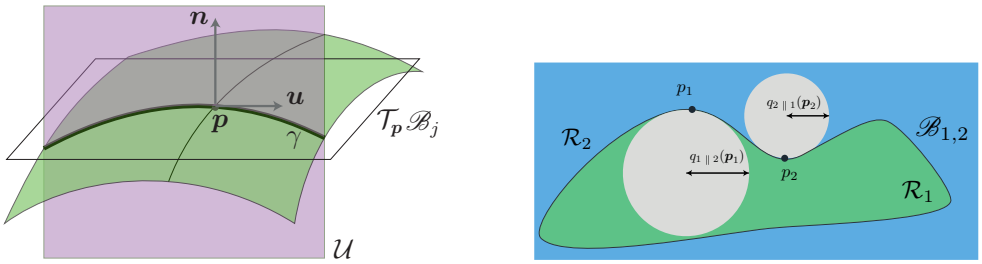
The result in Theorem 1 shows that in the random and semi-random noise regimes, the robustness to noise is precisely related to $\|\boldsymbol{r}^*\|_2$ by a factor of $\sqrt{d/m}$. Specifically, in the random noise regime ($m = 1$), the magnitude of the noise required to misclassify the datapoint behaves as $\Theta(\sqrt{d}\|\boldsymbol{r}^*\|_2)$ with high probability, with constants in the interval $[\zeta_1(1,\delta), \zeta_2(1,\delta)]$. Our results therefore show that, in high dimensional classification settings, affine classifiers can be robust to random noise, even if the datapoint lies very close to the decision boundary (i.e., $\|\boldsymbol{r}^*\|_2$ is small). In the semi-random noise regime with m sufficiently large (e.g., $m \geq 250$), we have $\|\boldsymbol{r}^*_{\mathcal{S}}\|_2 \approx \sqrt{d/m}\|\boldsymbol{r}^*\|_2$ with high probability, as the constants $\zeta_1(m,\delta) \approx \zeta_2(m,\delta) \approx 1$ for sufficiently large m. Our bounds therefore "interpolate" between the random noise regime, which behaves as $\sqrt{d}\|\boldsymbol{r}^*\|_2$, and the worst-case noise $\|\boldsymbol{r}^*\|_2$. More importantly, the square root dependence is also notable here, as it shows that the semi-random robustness can remain small even in regimes where m is chosen to be a very small fraction of d. For example, choosing a small subspace of dimension $m = 0.01d$ results in semi-random robustness of $10\|\boldsymbol{r}^*\|_2$ with high probability, which might still not be perceptible in complex visual tasks. Hence, for semi-random noise that is mostly random and only mildly adversarial (i.e., the subspace dimension is small), affine classifiers remain vulnerable to such noise.

5.5 Robustness of general classifiers

5.5.1 Curvature of the decision boundary

We now consider the general case where f is a nonlinear classifier. We derive relations between the random and semi-random robustness $\|\boldsymbol{r}^*_{\mathcal{S}}\|_2$ and worst-case robustness $\|\boldsymbol{r}^*\|_2$ using properties of the classifier's *boundary*. Let i and j be two arbitrary classes; we define the pairwise boundary $\mathscr{B}_{i,j}$ as the boundary of the *binary* classifier where only classes i and j are considered. Formally, the decision boundary is given by $\mathscr{B}_{i,j} := \{\boldsymbol{x} \in \mathbb{R}^d : f_i(\boldsymbol{x}) - f_j(\boldsymbol{x}) = 0\}$. The boundary $\mathscr{B}_{i,j}$ separates between two regions of $\mathbb{R}^d$, namely $\mathcal{R}_i$ and $\mathcal{R}_j$, where the estimated label of the binary classifier is respectively i and j.

We assume for the purpose of this analysis that the boundary $\mathscr{B}_{i,j}$ is smooth. We are now interested in the geometric properties of the boundary, namely its curvature. Many notions of curvature can be defined on hypersurfaces [54]. In the simple case of a curve in a two-dimensional space, the curvature is defined as the inverse of the radius of the so-called oscullating circle. One way to define curvature for high-dimensional hypersurfaces is by taking *normal* sections of the hypersurface, and measuring the curvature of the resulting planar curve (see Fig. 5.3). We however introduce a notion of curvature that is specifically suited to the analysis of the decision boundary of a classifier. Informally, our curvature captures the *global* bending of the decision boundary by inscribing balls in the regions separated by the decision boundary. For a given $\boldsymbol{p} \in \mathscr{B}_{i,j}$, we define $q_{i\,\|\,j}(\boldsymbol{p})$ to be the radius of the largest open ball included in the region $\mathcal{R}_i$ that intersects with $\mathscr{B}_{i,j}$ at $\boldsymbol{p}$;



(a) (b)

Figure 5.3 – (a) Normal section of the boundary $\mathscr{B}_{i,j}$ with respect to plane $\mathcal{U} = \text{span}(\boldsymbol{n}, \boldsymbol{u})$, where $\boldsymbol{n}$ is the normal to the boundary at $\boldsymbol{p}$, and $\boldsymbol{u}$ is an arbitrary in the tangent space $\mathcal{T}_{\boldsymbol{p}}(\mathscr{B}_{i,j})$. (b) Illustration of the quantities introduced for the definition of the curvature of the decision boundary.

i.e.,

$$q_{i\,\|\,j}(\boldsymbol{p}) = \sup_{\boldsymbol{z}\in\mathbb{R}^d} \{\|\boldsymbol{z}-\boldsymbol{p}\|_2 : B(\boldsymbol{z}, \|\boldsymbol{z}-\boldsymbol{p}\|_2) \subseteq \mathcal{R}_i\}\,, \tag{5.7}$$

where $B(\boldsymbol{z}, \|\boldsymbol{z}-\boldsymbol{p}\|_2)$ is the open ball in $\mathbb{R}^d$ of center $\boldsymbol{z}$ and radius $\|\boldsymbol{z}-\boldsymbol{p}\|_2$. An illustration of this quantity in two dimensions is provided in Fig. 5.3 (b). It is not hard to see that any ball $B(\boldsymbol{z}^*, \|\boldsymbol{z}^*-\boldsymbol{p}\|_2)$ centered in $\boldsymbol{z}^*$ and included in $\mathcal{R}_i$ will have its tangent space at $\boldsymbol{p}$ coincide with the tangent of the decision boundary at the same point.

It should further be noted that the definition in Eq. (5.7) is not symmetric in i and j. We therefore define the following symmetric quantity $q_{i,j}(\boldsymbol{p})$, where the worst-case ball inscribed in any of the two regions $\mathcal{R}_i$ and $\mathcal{R}_j$ is considered:

$$q_{i,j}(\boldsymbol{p}) = \min\left(q_{i\,\|\,j}(\boldsymbol{p}), q_{j\,\|\,i}(\boldsymbol{p})\right).$$

To measure the global curvature, the worst-case radius is taken over all points on the decision boundary, i.e., $q(\mathscr{B}_{i,j}) = \inf_{\boldsymbol{p}\in\mathscr{B}_{i,j}} q_{i,j}(\boldsymbol{p})$. The curvature $\kappa(\mathscr{B}_{i,j})$ is then defined as the inverse of the worst-case radius: $\kappa(\mathscr{B}_{i,j}) = 1/q(\mathscr{B}_{i,j})$.

In the case of affine classifiers, we have $\kappa(\mathscr{B}_{i,j}) = 0$, as it is possible to inscribe balls of infinite radius inside each region of the space. When the classification boundary is a union of (sufficiently distant) spheres with equal radius R, the curvature $\kappa(\mathscr{B}_{i,j}) = 1/R$. In general, the quantity $\kappa(\mathscr{B}_{i,j})$ provides an intuitive way of describing the nonlinearity of the decision boundary by fitting balls inside the classification regions.

5.5.2 Robustness to random and semi-random noise

We now establish bounds on the robustness to random and semi-random noise in the binary classification case. Let $\boldsymbol{x}_0$ be a datapoint classified as $\hat{k} = \hat{k}(\boldsymbol{x}_0)$. We first study the

binary classification problem, where only classes $\hat{k}$ and $k \in \{1, \dots, L\} \backslash \{\hat{k}\}$ are considered. To simplify the notation, we let $\mathscr{B}_k := \mathscr{B}_{k,\hat{k}}$ be the decision boundary between classes k and $\hat{k}$. In the case of the binary classification problem where classes k and $\hat{k}$ are considered, the semi-random perturbation defined in Eq. (5.3) can be re-written as follows:

$$\boldsymbol{r}_{\mathcal{S}}^k = \arg\min_{\boldsymbol{r} \in \mathcal{S}} \|\boldsymbol{r}\|_2 \text{ s.t. } f_k(\boldsymbol{x}_0 + \boldsymbol{r}) \geq f_{\hat{k}}(\boldsymbol{x}_0 + \boldsymbol{r}). \tag{5.8}$$

The worst case perturbation (obtained with $\mathcal{S} = \mathbb{R}^d$) is denoted by $\boldsymbol{r}^k$. It should be noted that the global quantities $\boldsymbol{r}_{\mathcal{S}}^*$ and $\boldsymbol{r}^*$ are obtained from $\boldsymbol{r}_{\mathcal{S}}^k$ and $\boldsymbol{r}^k$ by taking the vectors with minimum norm over all classes k.

The following result gives upper and lower bounds on the ratio $\frac{\|\boldsymbol{r}_{\mathcal{S}}^k\|_2}{\|\boldsymbol{r}^k\|_2}$ in function of the curvature of the boundary separating class k and $\hat{k}$.

Theorem 2. *Let $\mathcal{S}$ be a random m-dimensional subspace of $\mathbb{R}^d$. Let $\kappa := \kappa(\mathscr{B}_k)$. Assuming that the curvature satisfies*

$$\kappa \leq \frac{C}{\zeta_2(m, \delta)\|\boldsymbol{r}^k\|_2} \frac{m}{d}, \tag{5.9}$$

the following inequality holds between the semi-random robustness $\|\boldsymbol{r}_{\mathcal{S}}^k\|_2$ and the adversarial robustness $\|\boldsymbol{r}^k\|_2$:

$$\left(1 - C_1\|\boldsymbol{r}^k\|_2 \kappa \zeta_2 \frac{d}{m}\right) \sqrt{\zeta_1}\sqrt{\frac{d}{m}} \leq \frac{\|\boldsymbol{r}_{\mathcal{S}}^k\|_2}{\|\boldsymbol{r}^k\|_2} \leq \left(1 + C_2\|\boldsymbol{r}^k\|_2 \kappa \zeta_2 \frac{d}{m}\right) \sqrt{\zeta_2}\sqrt{\frac{d}{m}} \tag{5.10}$$

with probability larger than $1 - 4\delta$. We recall that $\zeta_1 = \zeta_1(m, \delta)$ and $\zeta_2 = \zeta_2(m, \delta)$ are defined in Eq. (5.4, 5.5). The constants are $C = 0.2, C_1 = 0.625, C_2 = 2.25$.

The proof can be found in the appendix. This result shows that the bounds relating the robustness to random and semi-random noise to the worst-case robustness can be extended to nonlinear classifiers, provided the curvature of the boundary $\kappa(\mathscr{B}_k)$ is sufficiently small. In the case of linear classifiers, we have $\kappa(\mathscr{B}_k) = 0$, and we recover the result for affine classifiers from Theorem 1.

To extend this result to multi-class classification, special care has to be taken. In particular, if k denotes a class that has no boundary with class $\hat{k}$, $\|\boldsymbol{r}^k\|_2$ can be very large and the previous curvature condition is not satisfied. It is therefore crucial to *exclude* such classes that have no boundary in common with class $\hat{k}$, or more generally, boundaries that are far from class $\hat{k}$. We define the set A of excluded classes k where $\|\boldsymbol{r}^k\|_2$ is large

$$A = \{k : \|\boldsymbol{r}^k\|_2 \geq 1.45\sqrt{\zeta_2(m, \delta)}\sqrt{\frac{d}{m}}\|\boldsymbol{r}^*\|_2\}. \tag{5.11}$$

Note that A is independent of $\mathcal{S}$, and depends only on d, m and δ. Moreover, the constants in (5.11) were chosen for simplicity of exposition.

Assuming a curvature constraint *only on the close enough classes*, the following result establishes a simplified relation between $\|\boldsymbol{r}^*_{\mathcal{S}}\|_2$ and $\|\boldsymbol{r}^*\|_2$.

Corollary 1. *Let $\mathcal{S}$ be a random m-dimensional subspace of $\mathbb{R}^d$. Assume that, for all $k \notin A$, the curvature condition in Eq. (5.9) holds. Then, we have*

$$0.875\sqrt{\zeta_1(m,\delta)}\sqrt{\frac{d}{m}}\|\boldsymbol{r}^*\|_2 \leq \|\boldsymbol{r}^*_{\mathcal{S}}\|_2 \leq 1.45\sqrt{\zeta_2(m,\delta)}\sqrt{\frac{d}{m}}\|\boldsymbol{r}^*\|_2 \tag{5.12}$$

with probability larger than $1 - 4(L+2)\delta$.

Under the curvature condition in (5.9) on the boundaries between $\hat{k}$ and classes in $\{1, 2, \ldots, L\} - A$, our result shows that the robustness to random and semi-random noise exhibits the same behavior that has been observed earlier for linear classifiers in Theorem 1. In particular, $\|\boldsymbol{r}^*_{\mathcal{S}}\|_2$ is precisely related to the adversarial robustness $\|\boldsymbol{r}^*\|_2$ by a factor of $\sqrt{d/m}$. In the random regime ($m = 1$), this factor becomes $\sqrt{d}$, and shows that in high dimensional classification problems, classifiers with sufficiently flat boundaries are much more robust to random noise than to adversarial noise. However, in the semi-random, the factor is $\sqrt{d/m}$ and shows that robustness to semi-random noise might not be achieved even if m is chosen to be a tiny fraction of d. In other words, if a classifier is highly vulnerable to adversarial perturbations, then it is also vulnerable to noise that is overwhelmingly random and only mildly adversarial.

It is important to note that the curvature condition in Corollary 1 is *not* an assumption on the curvature of the global decision boundary, but rather an assumption on the decision boundaries between pairs of classes. The distinction here is significant, as junction points where two decision boundaries meet might actually have a very large (or infinite) curvature (even in linear classification settings), and the curvature condition in Corollary 1 typically does not hold for this global curvature definition. We refer to our experimental section for a visualization of this phenomenon.

5.6 Experiments

We now evaluate the robustness of different image classifiers to random and semi-random perturbations, and assess the accuracy of our bounds on various datasets and state-of-the-art classifiers. Specifically, our theoretical results show that the robustness $\|\boldsymbol{r}^*_{\mathcal{S}}(\boldsymbol{x})\|_2$ of classifiers satisfying the curvature property precisely behaves as $\sqrt{d/m}\|\boldsymbol{r}^*(\boldsymbol{x})\|_2$. We first check the accuracy of these results in different classification settings. For a given

Table 5.1 – $\beta(f; m)$ for different classifiers f and different subspace dimensions m. The VGG-F and VGG-19 are respectively introduced in [14, 83].

	m/d				
	$1/4$	$1/16$	$1/36$	$1/64$	$1/100$
MNIST					
LeNet	1.00 ± 0.06	1.01 ± 0.12	1.03 ± 0.20	1.01 ± 0.26	1.05 ± 0.34
CIFAR-10					
LeNet	1.01 ± 0.03	1.02 ± 0.07	1.04 ± 0.10	1.06 ± 0.14	1.10 ± 0.19
ImageNet					
VGG-F	1.00 ± 0.01	1.02 ± 0.02	1.03 ± 0.04	1.03 ± 0.05	1.04 ± 0.06
VGG-19	1.00 ± 0.01	1.02 ± 0.03	1.02 ± 0.05	1.03 ± 0.06	1.04 ± 0.08

classifier f and subspace dimension m, we define

$$\beta(f; m) = \sqrt{m/d} \frac{1}{|\mathscr{D}|} \sum_{\boldsymbol{x} \in \mathscr{D}} \frac{\|\boldsymbol{r}^*_{\mathcal{S}}(\boldsymbol{x})\|_2}{\|\boldsymbol{r}^*(\boldsymbol{x})\|_2},$$

where $\mathcal{S}$ is chosen randomly for each sample $\boldsymbol{x}$ and $\mathscr{D}$ denotes the test set. This quantity provides indication to the accuracy of our $\sqrt{d/m}\|\boldsymbol{r}^*(\boldsymbol{x})\|_2$ estimate of the robustness, and should ideally be equal to 1 (for sufficiently large m). Since β is a random quantity (because of $\mathcal{S}$), we report both its mean and standard deviation for different networks in Table 5.1.

It should be noted that finding $\|\boldsymbol{r}^*_{\mathcal{S}}\|_2$ and $\|\boldsymbol{r}^*\|_2$ involves solving the optimization problem in (5.2). We use Algorithm 3 proposed in Chapter 3 to find subspace minimal perturbations. For each network, we estimate the expectation by averaging $\beta(f; m)$ on 1000 random samples, with $\mathcal{S}$ also chosen randomly for each sample.

Observe that β is suprisingly close to 1, even when m is a small fraction of d. This shows that our quantitative analysis provide very accurate estimates of the robustness to semi-random noise. We visualize the robustness to random noise, semi-random noise (with $m = 10$) and worst-case perturbations on a sample image in Fig. 5.4. While random noise is clearly perceptible due to the $\sqrt{d} \approx 400$ factor, semi-random noise becomes much less perceptible even with a relatively small value of $m = 10$, thanks to the $1/\sqrt{m}$ factor that attenuates the required noise to misclassify the datapoint. It should be noted that the robustness of neural networks to adversarial perturbations has previously been observed empirically in [86], but we provide here a quantitative and generic explanation for this phenomenon.

The high accuracy of our bounds for different state-of-the-art classifiers, and different

(a) (b) (c) (d)

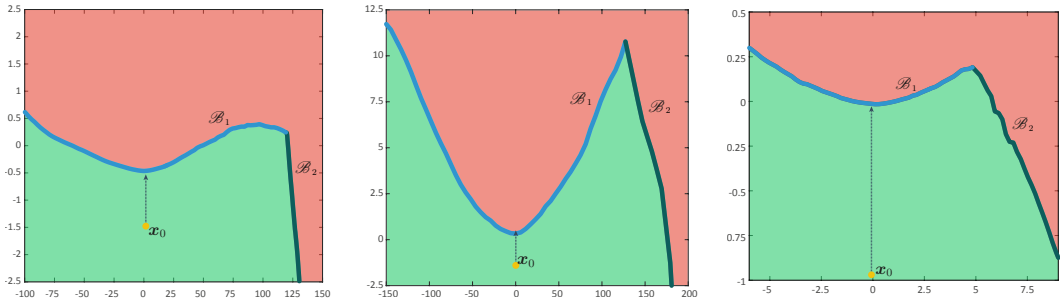
Figure 5.4 – (a) Original image classified as "Cauliflower". Fooling perturbations for VGG-F network: (b) Random noise, (c) Semi-random perturbation with $m = 10$, (d) Worst-case perturbation, all wrongly classified as "Artichoke".

datasets suggest that the decision boundaries of these classifiers have limited curvature $\kappa(\mathscr{B}_k)$, as this is a key assumption of our theoretical findings. To support the validity of this curvature hypothesis in practice, we visualize two-dimensional sections of the classifiers' boundary in Fig. 5.5 in three different settings. Note that we have opted here for a visualization strategy rather than the numerical estimation of $\kappa(\mathscr{B})$, as the latter quantity is difficult to approximate in practice in high dimensional problems. In Fig. 5.5, $\boldsymbol{x}_0$ is chosen randomly from the test set for each data set, and the decision boundaries are shown in the plane spanned by $\boldsymbol{r}^*$ and $\boldsymbol{r}^*_{\mathcal{S}}$, where $\mathcal{S}$ is a random *direction* (i.e., $m = 1$). Different colors on the boundary correspond to boundaries with different classes. It can be observed that the curvature of the boundary is very small except at "junction" points where the boundary of two different classes intersect. Our curvature assumption, which only assumes a bound on the curvature of the decision boundary between pairs of classes $\hat{k}(\boldsymbol{x}_0)$ and k (but not on the *global* decision boundary that contains junctions with high curvature) is therefore adequate to the decision boundaries of state-of-the-art classifiers according to Fig. 5.5. Interestingly, the assumption in Corollary 1 is satisfied by taking κ to be an empirical estimate of the curvature of the planar curves in Fig. 5.5 (a) for the dimension of the subspace being a *very* small fraction of d; e.g., $m = 10^{-3}d$. While not reflecting the curvature $\kappa(\mathscr{B}_k)$ that drives the assumption of our theoretical analysis, this result still seems to suggest that the curvature assumption holds in practice.

5.7 A note on the flatness of the decision boundary

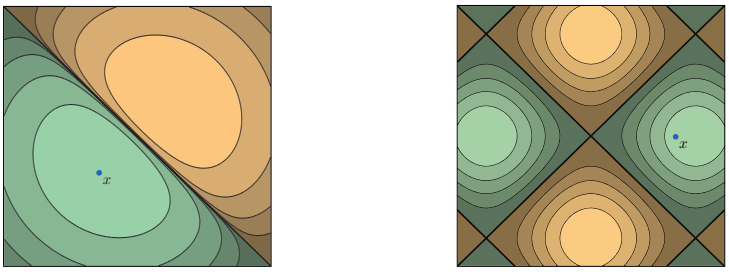
It can be observed that the decision boundaries of state-of-the-art deep neural networks have a very low curvature on these two dimensional *random* cross-sections in Fig. 5.5. In other words, these plots suggest that the decision boundary at the vicinity of $\boldsymbol{x}$ can be locally well approximated by a hyperplane passing through $\boldsymbol{x} + \boldsymbol{r}^*_{\text{adv}}(\boldsymbol{x})$ with the normal vector $\boldsymbol{r}^*_{\text{adv}}(\boldsymbol{x})$. A related observation was qualitatively reported in [95].

In [35], it is hypothesized that state-of-the-art classifiers are "too linear", leading to decision boundaries with very small curvature, and further explaining the high instability



(a) VGG-F (ImageNet) (b) LeNet (CIFAR) (c) LeNet (MNIST)

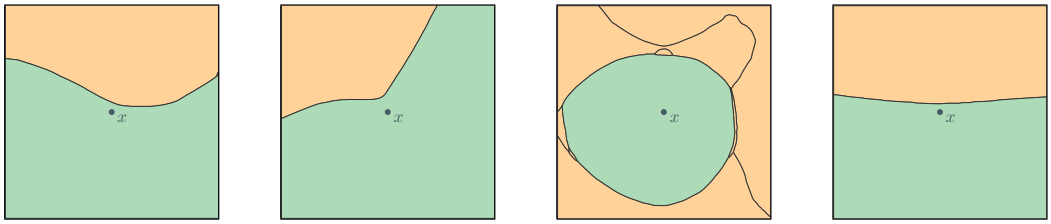
Figure 5.5 – Boundaries of three classifiers near randomly chosen samples. Axes are normalized by the corresponding $\|\boldsymbol{r}^*\|_2$ as our assumption in the theoretical bound depends on the product of $\|\boldsymbol{r}^*\|_2\kappa$. Note the difference in range between x and y axes. Note also that the range of horizontal axis in (c) is much smaller than the other two, hence the illustrated boundary is more curved.



(a) (b)

Figure 5.6 – The contours of two highly non-linear functions with linear boundaries. Specifically, the contours in the green and yellow regions represent the different (positive and negative) level sets of $g(x)$ (where $g(x) = g_1(x) - g_2(x)$, the difference between class 1 and class 2 score). The decision boundary is defined as the region of the space where $g(x) = 0$, and is indicated with a solid black line. Note that, although g is a highly nonlinear function in these examples, the decision boundaries are flat.

of such classifiers to adversarial perturbations. To motivate the linearity hypothesis of deep networks, the success of the Fast Gradient Sign method (which is exact for linear classifiers) in finding adversarial perturbations is invoked. However, some recent works challenge this linearity hypothesis; for example, in [81], the authors show that there exist adversarial perturbations that cannot be explained with this hypothesis, and in [89], the authors provide a new explanation based on the tilting of the decision boundary with respect to the data manifold. We stress here that the low curvature of the decision boundary does not, in general, imply that the function learned by the deep neural network (as a function of the input image) is linear, or even approximately linear. Fig. 5.6 shows illustrative examples of highly nonlinear functions resulting in flat decision boundaries. Moreover, it should be noted that, while the decision boundary of deep networks are very flat on *random* two dimensional cross-sections, these boundaries might not be flat



(a) (b) (c) (d)

Figure 5.7 – Cross-sections of the decision boundary in the vicinity of data point x. (a), (b), and (c) show decision boundaries with high curvature, while (d) shows the decision boundary along a random normal section (with very small curvature). The correct class and the neighboring classes are colored in green and orange respectively. The boundaries between different classes are shown in solid black lines. x and y axes have the same scale.

on *all* cross-sections. That is, there exist directions in which the boundary are very curved. Fig. 5.7 provides some illustrations of such cross-sections, where the decision boundary has large curvature, and therefore significantly departs from the first order linear approximation, suggested by the flatness of the decision boundary on random sections in Fig. 5.5. Hence, these visualizations of the decision boundary strongly suggest that the curvature along a small set of directions can be very large, and that the curvature is relatively small along random directions in the input space. In Chapter 7, using a numerical computation of the curvature, we empirically verify the *sparsity* of the curvature profile for deep neural networks.

5.8 Conclusion

In this chapter, we precisely characterized the robustness of classifiers in a novel semi-random noise regime that generalizes the random noise regime. Specifically, our bounds relate the robustness in this regime to the robustness to adversarial perturbations. Our bounds depend on the *curvature* of the decision boundary, the data dimension, and the dimension of the subspace to which the perturbation belongs. Our results show, in particular, that when the decision boundary has a small curvature, classifiers are robust to random noise in high dimensional classification problems (even if the robustness to adversarial perturbations is relatively small).

Moreover, for semi-random noise that is mostly random and only mildly adversarial (i.e., the subspace dimension is small), our results show that state-of-the-art classifiers remain vulnerable to such perturbations as they have extremely small curvature along random directions. However, such observation does not fully explain the vulnerability of classifiers to universal perturbations introduced in Chapter 4. In fact, the directions where the decision boundary is curved play a major role in explaining the robustness properties of classifiers to universal perturbations. In the next chapter, we particularly show that

in spite of having small curvatures in most of directions, there exist low-dimensional subspaces where state-of-the-art deep networks have a rather high curvature. In other words, though the decision boundary of deep classifiers is almost flat, there exist highly curved directions along which universal perturbations can be sought.

We finally note that the result presented in this chapter for ℓ_2-norm perturbations has similarly been extended to other ℓ_p-norm perturbations in [33].

6 Geometric analysis with universal perturbations

"Everyone knows what a curve is, until he has studied enough mathematics to become confused through the countless number of possible exceptions."
— Felix Klein

6.1 Introduction

In Chapter 4, we empirically showed that state-of-the-art classifiers are vulnerable to universal perturbations: there exist very small *image-agnostic* perturbations that cause most natural images to be misclassified. To recall, universal perturbations fundamentally differ from the semi-random noise regime introduced in the previous chapter, and exploit essential properties of deep networks to misclassify most natural images with perturbations of very small magnitude. Why are state-of-the-art classifiers highly vulnerable to these specific directions in the input space? What do these directions represent? To answer these questions, we follow a theoretical approach and find the causes of this vulnerability in the geometry of the decision boundaries induced by deep neural networks. For deep networks, we show that the key to answering these questions lies in the existence of shared directions (across different datapoints) along which the decision boundary is highly curved. This establishes fundamental connections between geometry and robustness to universal perturbations, and thereby reveals new properties of the decision boundaries induced by deep networks.

Our aim in this chapter is to derive an analysis of the vulnerability to universal perturbations in terms of the geometric properties of the decision boundary. To this end, we

Part of this chapter has been published in

Seyed-Mohsen Moosavi-Dezfooli*, Alhussein Fawzi*, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017,

Seyed-Mohsen Moosavi-Dezfooli*, Alhussein Fawzi*, Omar Fawzi, Pascal Frossard, and Stefano Soatto. Robustness of classifiers to universal perturbations: A geometric perspective. In *Sixth International Conference on Learning Representations*, 2018. (*: Equal contribution)

introduce two decision boundary models: 1) the *locally flat* model assumes that the first order linear approximation of the decision boundary holds locally in the vicinity of the natural images, and 2) the *locally curved* model provides a second order local description of the decision boundary, and takes into account the curvature information.

Under the *locally flat* decision boundary model, we show that classifiers are vulnerable to universal directions as long as the normals to the decision boundaries in the vicinity of natural images are correlated (i.e., they approximately span a low dimensional space). This result formalizes and proves some of the empirical observations made in Chapter 4. On the other hand, under the locally curved decision boundary model, the robustness to universal perturbations is instead driven by the *curvature* of the decision boundary; we show that the existence of *shared* directions along which the decision boundary is positively[1] curved implies the existence of very small universal perturbations.

For state-of-the-art deep networks, we show that the assumption of our theorem derived for the locally curved model is satisfied, that is there actually exist shared directions along which the decision boundary of deep neural networks are positively curved. Our theoretical result consequently captures the large vulnerability of state-of-the-art deep networks to universal perturbations. We finally show that the developed theoretical framework provides a novel (geometric) method for computing universal perturbations, and further explains some of the properties observed in Chapter 4 (e.g., diversity, transferability) regarding the robustness to universal perturbations.

The rest of this chapter is organized as follows: we start by introducing the necessary notations and definitions in Section 6.2. The flat model for the decision boundary of classifiers is introduced in Section 6.3, and its validity to explain the existence of universal perturbations in deep networks is studied in Section 6.3.1. In Section 6.4, we propose a curved model for the decision boundary of classifiers, and we empirically show it better explains the vulnerability of deep networks to universal perturbations.

6.2 Definitions and notations

Consider an L-class classifier $f : \mathbb{R}^d \to \mathbb{R}^L$. Given a datapoint $\boldsymbol{x} \in \mathbb{R}^d$, we define the estimated label $\hat{k}(\boldsymbol{x}) = \arg\max_k f_k(\boldsymbol{x})$, where $f_k(\boldsymbol{x})$ is the kth component of $f(\boldsymbol{x})$ that corresponds to the k^{th} class. We define by μ a distribution over natural images in $\mathbb{R}^d$. The main focus of this chapter is to analyze the robustness of classifiers to *universal* (image-agnostic) noise. Specifically, we define $\boldsymbol{v}$ to be a *universal* noise vector if $\hat{k}(\boldsymbol{x} + \boldsymbol{v}) \neq \hat{k}(\boldsymbol{x})$ for "most" $\boldsymbol{x} \sim \mu$. Formally, a perturbation $\boldsymbol{v}$ is (ξ, δ)-universal, if the

[1] Throughout the chapter, the sign of the curvature is chosen according to the normal vector, and the data point x, as illustrated in Fig. 6.5

following two constraints are satisfied:

$$\|\boldsymbol{v}\|_2 \leq \xi,$$
$$\mathbb{P}\left(\hat{k}(\boldsymbol{x}+\boldsymbol{v}) \neq \hat{k}(\boldsymbol{x})\right) \geq 1-\delta.$$

This perturbation image $\boldsymbol{v}$ is coined "universal", as it represents a fixed image-agnostic perturbation that causes label change for a large fraction of images sampled from the data distribution μ. In Chapter 4, state-of-the-art classifiers have been shown to be surprisingly vulnerable to this simple perturbation regime.

It should be noted that universal perturbations are different from adversarial perturbations [86, 7], which are datapoint-specific perturbations that are sought to fool a *specific* image. An adversarial perturbation is a solution to the following optimization problem

$$\boldsymbol{r}(\boldsymbol{x}) = \arg\min_{\boldsymbol{r}\in\mathbb{R}^d} \|\boldsymbol{r}\|_2 \text{ subject to } \hat{k}(\boldsymbol{x}+\boldsymbol{r}) \neq \hat{k}(\boldsymbol{x}), \tag{6.1}$$

which corresponds to the smallest additive perturbation that is necessary to change the label of the classifier $\hat{k}$ for $\boldsymbol{x}$. From a geometric perspective, $\boldsymbol{r}(\boldsymbol{x})$ quantifies the distance from $\boldsymbol{x}$ to the decision boundary (see Fig. 6.1a). In addition, due to the optimality conditions of Eq. (6.1), $\boldsymbol{r}(\boldsymbol{x})$ is orthogonal to the decision boundary at $\boldsymbol{x}+\boldsymbol{r}(\boldsymbol{x})$, as illustrated in Fig. 6.1a.

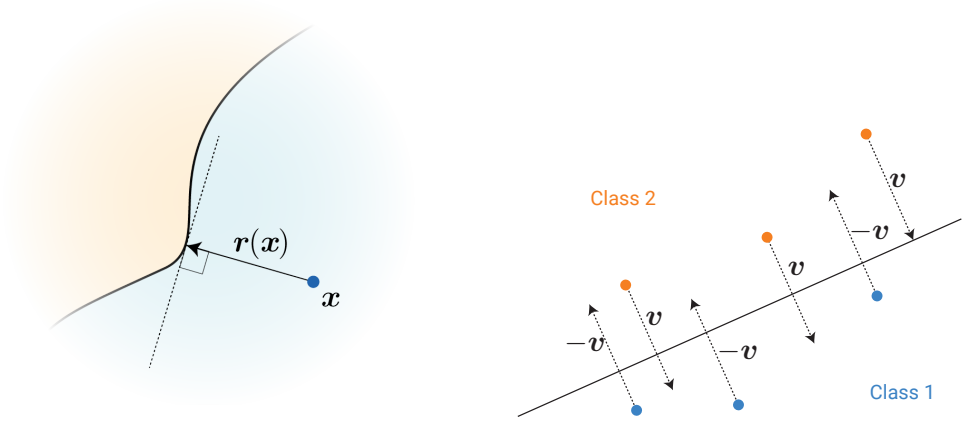
In the remainder of the chapter, we analyze the robustness of classifiers to universal noise, with respect to the geometry of the *decision boundary* of the classifier f. Formally, the pairwise decision boundary, when restricting the classifier to class i and j is defined by $\mathscr{B} = \{\boldsymbol{z} \in \mathbb{R}^d : f_i(\boldsymbol{z}) - f_j(\boldsymbol{z}) = 0\}$ (we omit the dependence of $\mathscr{B}$ on i, j for simplicity). The decision boundary of the classifier hence corresponds to points in the input space that are equally likely to be classified as i or j.

In the following sections, we introduce two models on the decision boundary, and quantify in each case the robustness of such classifiers to universal perturbations. We then show that the *locally curved* model better explains the vulnerability of deep networks to such perturbations.

6.3 Robustness of classifiers with flat decision boundaries

We start here our analysis by assuming a locally flat decision boundary model, and analyze the robustness of classifiers to universal perturbations under this decision boundary model. We specifically study the existence of a universal direction $\boldsymbol{v}$, such that

$$\hat{k}(\boldsymbol{x}+\boldsymbol{v}) \neq \hat{k}(\boldsymbol{x}) \text{ or } \hat{k}(\boldsymbol{x}-\boldsymbol{v}) \neq \hat{k}(\boldsymbol{x}), \tag{6.2}$$



(a) Local geometry of the decision boundary.

(b) Universal direction $\boldsymbol{v}$ of a linear binary classifier.

Figure 6.1

where $\boldsymbol{v}$ is a vector of sufficiently small norm. It should be noted that a universal *direction* (as opposed to a universal vector) is sought in Eq. (6.2), as this definition is more adapted to the analysis of classifiers with locally flat decision boundaries. For example, while a binary linear classifier has a universal direction that fools all the data points, only half of the data points can be fooled with a universal vector (provided the classes are balanced) (see Fig. 6.1b). We therefore consider this slightly modified definition in the remainder of this section.

We start our analysis by introducing our local decision boundary model. For $\boldsymbol{x} \in \mathbb{R}^d$, note that $\boldsymbol{x} + \boldsymbol{r}(\boldsymbol{x})$ belongs to the decision boundary and $\boldsymbol{r}(\boldsymbol{x})$ is normal to the decision boundary at $\boldsymbol{x} + \boldsymbol{r}(\boldsymbol{x})$ (see Fig. 6.1a). A linear approximation of the decision boundary of the classifier at $\boldsymbol{x} + \boldsymbol{r}(\boldsymbol{x})$ is therefore given by $\boldsymbol{x} + \{\boldsymbol{v} : \boldsymbol{r}(\boldsymbol{x})^T \boldsymbol{v} = \|\boldsymbol{r}(\boldsymbol{x})\|_2^2\}$. Under this approximation, the vector $\boldsymbol{r}(\boldsymbol{x})$ hence captures the local geometry of the decision boundary in the vicinity of datapoint $\boldsymbol{x}$. We assume a local decision boundary model in the vicinity of datapoints $\boldsymbol{x} \sim \mu$, where the local classification region of $\boldsymbol{x}$ occurs in the halfspace $\boldsymbol{r}(\boldsymbol{x})^T \boldsymbol{v} \leq \|\boldsymbol{r}(\boldsymbol{x})\|_2^2$. Equivalently, we assume that outside of this half-space, the classifier outputs a different label than $\hat{k}(\boldsymbol{x})$. However, since we are analyzing the robustness to universal *directions* (and not vectors), we consider the following condition, given by

$$\begin{aligned} \mathscr{L}_s(\boldsymbol{x}, \rho) : \forall \boldsymbol{v} \in B(\rho), |\boldsymbol{r}(\boldsymbol{x})^T \boldsymbol{v}| \geq \|\boldsymbol{r}(\boldsymbol{x})\|_2^2 \\ \implies \hat{k}(\boldsymbol{x} + \boldsymbol{v}) \neq \hat{k}(\boldsymbol{x}) \text{ or } \hat{k}(\boldsymbol{x} - \boldsymbol{v}) \neq \hat{k}(\boldsymbol{x}). \end{aligned} \tag{6.3}$$

where $B(\rho)$ is a ball of radius ρ centered at $\mathbf{0}$. An illustration of this decision boundary model is provided in Fig. 6.2a. It should be noted that linear classifiers satisfy this decision boundary model, as their decision boundaries are globally flat. This *local* decision boundary model is however more general, as we do *not* assume that the decision boundary

is linear, but rather that the classification region in the vicinity of $\boldsymbol{x}$ is included in $\boldsymbol{x} + \{\boldsymbol{v} : |\boldsymbol{r}(\boldsymbol{x})^T \boldsymbol{v}| \leq \|\boldsymbol{r}(\boldsymbol{x})\|_2^2\}$. Moreover, it should be noted that the model being assumed here is on the decision boundary of the classifier, and not an assumption on the classification function f.[2] Fig. 6.2a provides an example of nonlinear decision boundary that satisfies this model.

In all the theoretical results of this chapter, we assume that $\|\boldsymbol{r}(\boldsymbol{x})\|_2 = 1$, for all $\boldsymbol{x} \sim \mu$, for simplicity of the exposition. The results can be extended in a straightforward way to the case where $\|\boldsymbol{r}(\boldsymbol{x})\|_2$ takes different values for points sampled from μ. The following result shows that classifiers following the locally flat decision boundary model are *not* robust to small universal perturbations, provided the normals to the decision boundary (in the vicinity of datapoints) approximately belong to a low dimensional subspace of dimension $m \ll d$.

Theorem 3. *Let $\xi \geq 0, \delta \geq 0$. Let $\mathcal{S}$ be an m dimensional subspace such that $\|P_{\mathcal{S}} \boldsymbol{r}(\boldsymbol{x})\|_2 \geq 1 - \xi$ for almost all $\boldsymbol{x} \sim \mu$,, where $P_{\mathcal{S}}$ is the projection operator on the subspace. Assume moreover that $\mathscr{L}_s(\boldsymbol{x}, \rho)$ holds for almost all $\boldsymbol{x} \sim \mu$, with $\rho = \sqrt{em}/\delta(1-\xi)$. Then, there exists a universal noise vector $\boldsymbol{v}$, such that $\|\boldsymbol{v}\|_2 \leq \rho$ and*

$$\mathop{\mathbb{P}}_{\boldsymbol{x} \sim \mu} \left(\hat{k}(\boldsymbol{x} + \boldsymbol{v}) \neq \hat{k}(\boldsymbol{x}) \text{ or } \hat{k}(\boldsymbol{x} - \boldsymbol{v}) \neq \hat{k}(\boldsymbol{x}) \right) \geq 1 - \delta.$$

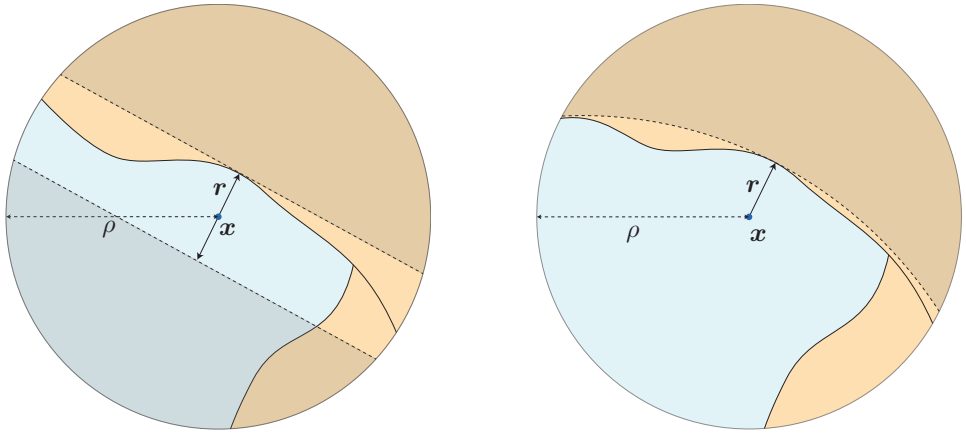
The proof can be found in supplementary material, and relies on the construction of a universal perturbation through randomly sampling from $\mathcal{S}$. The vulnerability of classifiers to universal perturbations can be attributed to the *shared* geometric properties of the classifier's decision boundary in the vicinity of different data points. In the above theorem, this shared geometric property across different data points is expressed in terms of the normal vectors $\boldsymbol{r}(\boldsymbol{x})$. The necessary condition of the above theorem is specifically that normal vectors $\boldsymbol{r}(\boldsymbol{x})$ to the decision boundary in the neighborhood of data points approximately live in a subspace $\mathcal{S}$ of low dimension $m < d$. Under this assumption, the above result shows the existence of universal perturbations of ℓ_2 norm of order $\sqrt{m}$. When $m \ll d$, Theorem 3 hence shows that very small (compared to random noise, which scales as $\sqrt{d}$ [29]) universal perturbations misclassifying most data points can be found.

Remark 1. Theorem 3 can be readily applied to assess the robustness of multiclass linear classifiers to universal perturbations. In fact, when $f(\boldsymbol{x}) = W^T \boldsymbol{x}$, with $W = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_L]$, the normal vectors are equal to $\boldsymbol{w}_i - \boldsymbol{w}_j$, for $1 \leq i, j \leq L, i \neq j$. These normal vectors exactly span a subspace of dimension $L - 1$. Hence, by applying the result with $\xi = 0$, and $m = L - 1$, we obtain that linear classifiers are vulnerable to universal noise, with magnitude proportional to $\sqrt{L-1}$. In typical problems, we have $L \ll d$, which leads to very small universal directions.

[2] The decision boundary $\mathscr{B}$ is the zero level set of the functions $f_i - f_j$. f can be a highly nonlinear function of the inputs, even when the zero-level set $\mathscr{B}$ is locally flat in the vicinity of datapoints.

Remark 2. Theorem 3 provides a partial explanation to the vulnerability of deep networks, provided a locally flat decision boundary is assumed. Evidence in favor of this assumption was given through visualization of randomly chosen cross-sections in [95, 29]. In addition, we show in Section 6.3.1 that normal vectors to the decision boundary of deep networks (near data points) approximately span a subspace $\mathcal{S}$ of sufficiently small dimension. However, unlike linear classifiers, the dimensionality of this subspace m is typically larger than the the number of classes L, leading to large upper bounds on the norm of the universal noise, under the flat decision boundary model.

We show in Section 6.4 that the second order information of the decision boundary contains crucial information (*curvature*) that captures the high vulnerability to universal perturbations. We verify this claim for state-of-the-art classifiers in Section 6.4.1.



(a) Flat decision boundary model $\mathscr{L}_s(\boldsymbol{x}, \rho)$.

(b) Curved decision boundary model $\mathscr{Q}(\boldsymbol{x}, \rho)$.

Figure 6.2 – Illustration of the decision boundary models considered in this chapter. (a): For the flat decision boundary model, the set $\{\boldsymbol{v} : |\boldsymbol{r}(\boldsymbol{x})^T \boldsymbol{v}| \leq \|\boldsymbol{r}(\boldsymbol{x})\|_2^2\}$ is illustrated (stripe). Note that for $\boldsymbol{v}$ taken outside the stripe (i.e., in the grayed area), we have $\hat{k}(\boldsymbol{x}+\boldsymbol{v}) \neq \hat{k}(\boldsymbol{x})$ or $\hat{k}(\boldsymbol{x}-\boldsymbol{v}) \neq \hat{k}(\boldsymbol{x})$ in the ρ neighborhood. (b): For the curved decision boundary model, the any vector $\boldsymbol{v}$ chosen in the grayed area is classified differently from $\hat{k}(\boldsymbol{x})$.

6.3.1 Experimental results

We first show that the normals to the decision boundary of state-of-the-art deep classifiers span a low-dimensional subspace. To do so, for each image $\boldsymbol{x}$ in the validation set, we compute the adversarial perturbation vector $\boldsymbol{r}(\boldsymbol{x}) = \arg\min_{\boldsymbol{r}} \|\boldsymbol{r}\|_2$ s.t. $\hat{k}(\boldsymbol{x}+\boldsymbol{r}) \neq \hat{k}(\boldsymbol{x})$. It is easy to see that $\boldsymbol{r}(\boldsymbol{x})$ is *normal* to the decision boundary of the classifier (at $\boldsymbol{x}+\boldsymbol{r}(\boldsymbol{x})$). The vector $\boldsymbol{r}(\boldsymbol{x})$ hence captures the local geometry of the decision boundary in the region surrounding the data point $\boldsymbol{x}$. To quantify the correlation between different regions of

the decision boundary of the classifier, we define the matrix

$$N = \left[\frac{\boldsymbol{r}(\boldsymbol{x}_1)}{\|\boldsymbol{r}(\boldsymbol{x}_1)\|_2} \cdots \frac{\boldsymbol{r}(\boldsymbol{x}_n)}{\|\boldsymbol{r}(\boldsymbol{x}_n)\|_2} \right]$$

of normal vectors to the decision boundary in the vicinity of n data points in the validation set. For binary linear classifiers, the decision boundary is a hyperplane, and N is of rank 1, as all normal vectors are collinear. To capture more generally the correlations in the decision boundary of complex classifiers, we compute the singular values of the matrix N. The singular values of the matrix N, computed for the CaffeNet architecture are shown in Fig. 6.3. We further show in the same figure the singular values obtained when the columns of N are sampled uniformly at random from the unit sphere. Observe that, while the latter singular values have a slow decay, the singular values of N decay quickly, which confirms the existence of large correlations and redundancies in the decision boundary of deep networks. More precisely, this suggests the existence of a subspace $\mathcal{S}_f$ of low dimension d' (with $d' \ll d$), that contains most normal vectors to the decision boundary in regions surrounding natural images. We hypothesize that the existence of universal perturbations fooling most natural images is partly due to the existence of such a low-dimensional subspace that captures the correlations among different regions of the decision boundary. In fact, this subspace "collects" normals to the decision boundary in different regions, and perturbations belonging to this subspace are therefore likely to fool datapoints. To verify this hypothesis, we choose a *random* vector of norm $\xi = 2000$ belonging to the subspace $\mathcal{S}_f$ spanned by the first 100 singular vectors, and compute its fooling ratio on a different set of images (i.e., a set of images that have not been used to compute the SVD). Such a perturbation can fool nearly 38% of these images, thereby showing that a *random* direction in this well-sought subspace $\mathcal{S}_f$ significantly outperforms random perturbations (we recall that such perturbations can only fool 10% of the data). Fig. 6.4 illustrates the subspace $\mathcal{S}_f$ that captures the correlations in the decision boundary. It should further be noted that the existence of this low dimensional subspace explains the surprising generalization properties of universal perturbations observed in Chapter 4, where one can build relatively generalizable universal perturbations with very few images.

The gap between the fooling rates obtained with the random vector strategy in $\mathcal{S}_f$ and Algorithm 4 of Chapter 4, demonstrates that this simplified flat model of the decision boundary fails to fully explain the large vulnerability of state-of-the-art deep neural networks to universal perturbations.

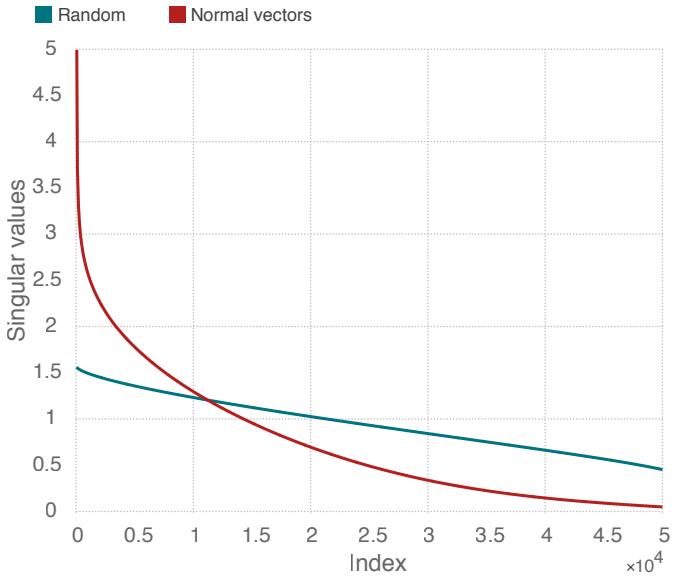


Figure 6.3 – Singular values of matrix N containing normal vectors to the decision decision boundary.

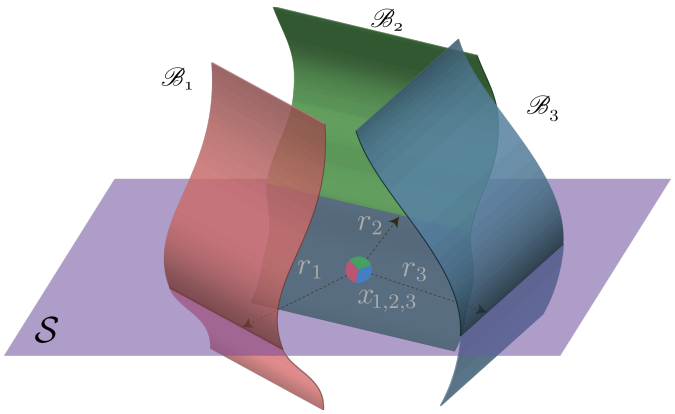


Figure 6.4 – Illustration of the low dimensional subspace $\mathcal{S}$ containing normal vectors to the decision boundary in regions surrounding natural images. For the purpose of this illustration, we super-impose three data-points $\{\boldsymbol{x}_i\}_{i=1}^3$, and the adversarial perturbations $\{\boldsymbol{r}_i\}_{i=1}^3$ that send the respective datapoints to the decision boundary $\{\mathscr{B}_i\}_{i=1}^3$ are shown. Note that $\{\boldsymbol{r}_i\}_{i=1}^3$ all live in the subspace $\mathcal{S}_f$.

6.4 Robustness of classifiers with curved decision boundaries

We now consider a model of the decision boundary in the vicinity of the data points that allows to leverage the *curvature* of nonlinear classifiers. Under this decision boundary model, we study the existence of universal perturbations satisfying $\hat{k}(\boldsymbol{x}+\boldsymbol{v}) \neq \hat{k}(\boldsymbol{x})$ for most $\boldsymbol{x} \sim \mu$.[3]

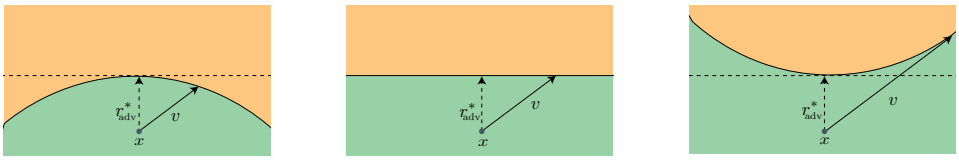


Figure 6.5 – Link between robustness and curvature of the decision boundary. When the decision boundary is *positively* curved (left), small universal perturbations are more likely to fool the classifier.

We start by establishing an informal link between curvature of the decision boundary and robustness to universal perturbations, that will be made clear later in this section. As illustrated in Fig. 6.5, the norm of the required perturbation to change the label of the classifier along a specific direction $\boldsymbol{v}$ is smaller if the decision boundary is positively curved, than if the decision boundary is flat (or with negative curvature). It therefore appears from Fig. 6.5 that the existence of universal perturbations (when the decision boundary is curved) can be attributed to the existence of *common* directions where the decision boundary is positively curved for many data points. In the remaining of this section, we formally prove the existence of universal perturbations, when there exists *common* positively curved directions of the decision boundary.

Recalling the definitions of Sec. 6.2, a quadratic approximation of the decision boundary at $\boldsymbol{z} = \boldsymbol{x}+\boldsymbol{r}(\boldsymbol{x})$ gives $\boldsymbol{x}+\{\boldsymbol{v} : (\boldsymbol{v}-\boldsymbol{r}(\boldsymbol{x}))^T H_{\boldsymbol{z}}(\boldsymbol{v}-\boldsymbol{r}(\boldsymbol{x}))+\alpha_x \boldsymbol{r}(\boldsymbol{x})^T(\boldsymbol{v}-\boldsymbol{r}(\boldsymbol{x}))=0\}$, where $H_{\boldsymbol{z}}$ denotes the Hessian of F at $\boldsymbol{z}$, and $\alpha_x = \frac{\|\nabla F(\boldsymbol{z})\|_2}{\|\boldsymbol{r}(\boldsymbol{x})\|_2}$, with $F = f_i - f_j$. In this model, the second order information (encoded in the Hessian matrix $H_{\boldsymbol{z}}$) captures the curvature of the decision boundary. We assume a *local* decision boundary model in the vicinity of datapoints $\boldsymbol{x} \sim \mu$, where the local classification region of $\boldsymbol{x}$ is bounded by a quadratic form. Formally, we assume that there exists $\rho > 0$ where the following condition holds for almost all $\boldsymbol{x} \sim \mu$:

$$\mathscr{Q}(\boldsymbol{x}, \rho) : \forall \boldsymbol{v} \in B(\rho),$$
$$(\boldsymbol{v}-\boldsymbol{r}(\boldsymbol{x}))^T H_{\boldsymbol{z}}(\boldsymbol{v}-\boldsymbol{r}(\boldsymbol{x})) + \alpha_x \boldsymbol{r}(\boldsymbol{x})^T(\boldsymbol{v}-\boldsymbol{r}(\boldsymbol{x})) \leq 0 \implies \hat{k}(\boldsymbol{x}+\boldsymbol{v}) \neq \hat{k}(\boldsymbol{x}).$$

[3] Unlike for classifiers with locally flat decision boundaries, we now consider the problem of finding a universal *vector* (as opposed to universal *direction*) that fools most of the data points. This corresponds to the notion of universal perturbations first highlighted in [68].

An illustration of this quadratic decision boundary model is shown in Fig. 6.2b. The following result shows the existence of universal perturbations, provided a subspace $\mathcal{S}$ exists where the decision boundary has positive curvature along most directions of $\mathcal{S}$:

Theorem 4. *Let $\kappa, \delta, \beta > 0$ and $m \in \mathbb{N}$. Assume that the quadratic decision boundary model $\mathcal{Q}(\boldsymbol{x}, \rho)$ holds for almost all $\boldsymbol{x} \sim \mu$, with $\rho = \sqrt{\frac{2\log(2/\delta)}{m}}\kappa^{-1} + \kappa^{-1/2}$. Let $\mathcal{S}$ be a m dimensional subspace such that*

$$\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\mathbb{S}}\left(\forall \boldsymbol{u} \in \mathbb{R}^2, \alpha_x^{-1}\boldsymbol{u}^T H_{\boldsymbol{z}}^{\boldsymbol{r}(\boldsymbol{x}),\boldsymbol{v}}\boldsymbol{u} \geq \kappa\|\boldsymbol{u}\|_2^2\right) \geq 1-\beta \text{ for almost all } \boldsymbol{x} \sim \mu,$$

where $H_{\boldsymbol{z}}^{\boldsymbol{r}(\boldsymbol{x}),\boldsymbol{v}} = \Pi^T H_{\boldsymbol{z}} \Pi$ with Π an orthonormal basis of span$(\boldsymbol{r}(\boldsymbol{x}), \boldsymbol{v})$, and $\mathbb{S}$ denotes the unit sphere in $\mathcal{S}$. Then, there is a universal perturbation vector $\boldsymbol{v}$ such that $\|\boldsymbol{v}\|_2 \leq \rho$ and $\mathop{\mathbb{P}}_{\boldsymbol{x}\sim\mu}\left(\hat{k}(\boldsymbol{x}+\boldsymbol{v}) \neq \hat{k}(\boldsymbol{x})\right) \geq 1-\delta-\beta$.

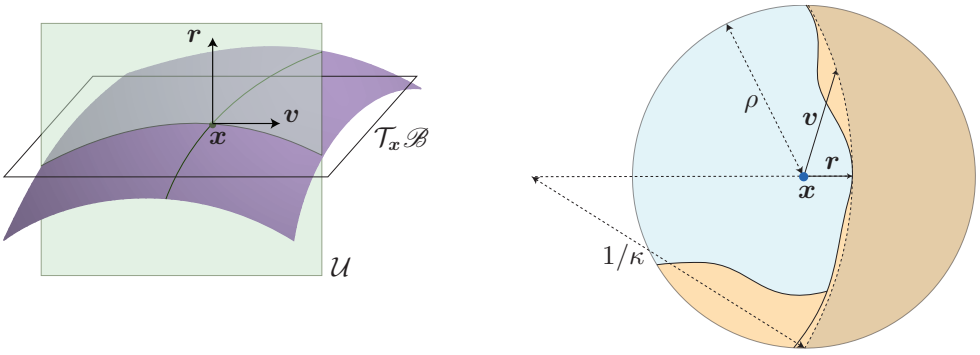


Figure 6.6 – **Left:** Normal section $\mathcal{U}$ of the decision boundary, along the plane spanned by the normal vector $\boldsymbol{r}(\boldsymbol{x})$ and $\boldsymbol{v}$. **Right:** Geometric interpretation of the assumption in Theorem 4. Theorem 4 assumes that the decision boundary along normal sections $(\boldsymbol{r}(\boldsymbol{x}), \boldsymbol{v})$ is locally (in a ρ neighborhood) located *inside* a disk of radius $1/\kappa$. Note the difference with respect to traditional notions of curvature, which express the curvature in terms of the osculating circle at $\boldsymbol{x} + \boldsymbol{r}(\boldsymbol{x})$. The assumption we use here is more "global".

The above theorem quantifies the robustness of classifiers to universal perturbations in terms of the curvature κ of the decision boundary, along normal sections spanned by $\boldsymbol{r}(\boldsymbol{x})$, and vectors $\boldsymbol{v} \in \mathcal{S}$ (see Fig. 6.6 (*left*) for an illustration of a normal section). Fig. 6.6 (*right*) provides a geometric illustration of the condition under which Theorem 4 holds. Provided a subspace $\mathcal{S}$ exists where the curvature of the decision boundary in the vicinity of datapoints $\boldsymbol{x}$ is positive (along directions in $\mathcal{S}$), Theorem 4 shows that universal perturbations can be found with a norm of approximately $\frac{\kappa^{-1}}{\sqrt{m}} + \kappa^{-1/2}$. Hence, when the curvature κ is sufficiently large, the existence of small universal perturbations is guaranteed with Theorem 4.[4]

[4]Theorem 4 should not be seen as a generalization of Theorem 3, as the models are distinct. In fact, while the latter shows the existence of universal *directions*, the former bounds the existence of universal *perturbations*.

Remark 1. We stress that Theorem 4 does *not* assume that the decision boundary is curved in the direction of all vectors in $\mathbb{R}^d$, but we rather assume the existence of a subspace $\mathcal{S}$ where the decision boundary is positively curved (in the vicinity of natural images $\boldsymbol{x}$) along most directions in $\mathcal{S}$. Moreover, it should be noted that, unlike Theorem 3, where the normals to the decision boundary are assumed to belong to a low dimensional subspace, no assumption is imposed on the normal vectors. Instead, we assume the existence of a subspace $\mathcal{S}$ leading to positive curvature, for points on the decision boundary in the vicinity of natural images.

Remark 2. Theorem 4 does not only predict the vulnerability of classifiers, but it also provides a constructive way to find such universal perturbations. In fact, *random vectors* sampled from the subspace $\mathcal{S}$ are predicted to be universal perturbations (see supp. material for more details). Now, we show that this new construction works remarkably well for deep networks, as predicted by our analysis.

6.4.1 Experimental results

We first evaluate the validity of the assumption of Theorem 4 for deep neural networks, that is the existence of a low dimensional subspace where the decision boundary is positively curved along most directions sampled from the subspace. To construct the subspace, we find the directions that lead to large positive curvature in the vicinity of a given set of training points $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$. We recall that principal directions $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{d-1}$ at a point $\boldsymbol{z}$ on the decision boundary correspond to the eigenvectors (with nonzero eigenvalue) of the matrix $H^t_{\boldsymbol{z}}$, given by $H^t_{\boldsymbol{z}} = PH_{\boldsymbol{z}}P$, where P denotes the projection operator on the tangent to the decision boundary at $\boldsymbol{z}$, and $H_{\boldsymbol{z}}$ denotes the Hessian of the decision boundary function evaluated at $\boldsymbol{z}$ [54]. Common directions with large average curvature at $\boldsymbol{z}_i = \boldsymbol{x}_i + \boldsymbol{r}(\boldsymbol{x}_i)$ (where $\boldsymbol{r}(\boldsymbol{x}_i)$ is the minimal perturbation defined in Eq. (6.1)) hence correspond to the eigenvectors of the average Hessian matrix $\overline{H} = n^{-1}\sum_{i=1}^{n} H^t_{\boldsymbol{z}_i}$. We therefore set our subspace, $\mathcal{S}_c$, to be the span of the first m eigenvectors of $\overline{H}$, and show that the subspace constructed in this way satisfies the assumption of Theorem 4. To determine whether the decision boundary is positively curved in most directions of $\mathcal{S}_c$ (for unseen datapoints from the validation set), we compute the average curvature across random directions in $\mathcal{S}_c$ for points on the decision boundary, i.e. $\boldsymbol{z} = \boldsymbol{x} + \boldsymbol{r}(\boldsymbol{x})$; the average curvature is formally given by

$$\overline{\kappa}_{\mathcal{S}}(\boldsymbol{x}) = \mathop{\mathbb{E}}_{\boldsymbol{v}\sim\mathbb{S}} \left(\frac{(P\boldsymbol{v})^T H_{\boldsymbol{z}}(P\boldsymbol{v})}{\|P\boldsymbol{v}\|_2^2} \right), \tag{6.4}$$

where $\mathbb{S}$ denotes the unit sphere in $\mathcal{S}_c$. In Fig. 6.9 (a), the average of $\overline{\kappa}_{\mathcal{S}}(\boldsymbol{x})$ across points sampled from the *validation set* is shown (as well as the standard deviation) in function of the subspace dimension m, for a LeNet architecture [52] trained on the CIFAR-10

dataset.[5] Observe that when the dimension of the subspace is sufficiently small, the average curvature is strongly oriented towards positive curvature, which empirically shows the existence of this subspace $\mathcal{S}_c$ where the decision boundary is positively curved for most data points in the validation set. This empirical evidence hence suggests that the assumption of Theorem 4 is satisfied, and that universal perturbations hence represent random vectors sampled from this subspace $\mathcal{S}_c$.

To show this strong link between the vulnerability of universal perturbations and the *positive curvature* of the decision boundary, we now visualize normal sections of the decision boundary of deep networks trained on ImageNet (CaffeNet [44] and ResNet-152 [38]) and CIFAR-10 (LeNet [52] and ResNet-18 [38]) in the direction of their respective universal perturbations.[6] Specifically, we visualize normal sections of the decision boundary in the plane $(\boldsymbol{r}(\boldsymbol{x}), \boldsymbol{v})$, where $\boldsymbol{v}$ is a universal perturbation computed using Algorithm 4 of Chapter 4. The visualizations are shown in Fig. 6.7 and 6.8. Interestingly, the universal perturbations belong to highly positively curved directions of the decision boundary, despite the absence of any geometric constraint in the algorithm to compute universal perturbations. To fool most data points, universal perturbations hence naturally seek *common directions* of the embedding space, where the decision boundary is positively curved. These directions lead to very small universal perturbations, as highlighted by our analysis in Theorem 4. It should be noted that such *highly curved* directions of the decision boundary are rare, as random normal sections are comparatively flat (see Fig. 6.7 and 6.8, second row). This is due to the fact that most principal curvatures are approximately zero, for points sampled on the decision boundary in the vicinity of data points.

Recall that Theorem 4 suggests a novel procedure to generate universal perturbations; in fact, random perturbations from $\mathcal{S}_c$ are predicted to be universal perturbations. To assess the validity of this result, Fig. 6.9 (b) illustrates the fooling rate of the universal perturbations (for the LeNet network on CIFAR-10) sampled uniformly at random from the unit sphere in subspace $\mathcal{S}_c$, and scaled to have a fixed norm (1/5th of the norm of the random noise required to fool most data points). We assess the quality of such perturbation by further indicating in Fig. 6.9 (b) the fooling rate of the universal perturbation computed using the algorithm in Chapter 4. Observe that random perturbations sampled from $\mathcal{S}_c$ (with m small) provide very powerful universal perturbations, fooling nearly 85% of data points from the validation set. This rate is comparable to that of the algorithm in Chapter 4, while using much less training points (only $n = 100$, while at least $2,000$ training points are required by the algorithm in Chapter 4). The very large fooling

[5] The LeNet architecture we used has two convolutional layers (filters of size 5) followed by three fully connected layers. We used SGD for training, with a step size 0.01 and a momentum term of 0.9 and weight decay of 10^{-4}. The accuracy of the network on the test set is 78.4%.

[6] For the networks on ImageNet, we used the Caffe pre-trained models https://github.com/BVLC/caffe/wiki/Model-Zoo. The ResNet-18 architecture was trained on the CIFAR-10 task with stochastic gradient descent with momentum and weight decay regularization. It achieves an accuracy on the test of 94.18%.

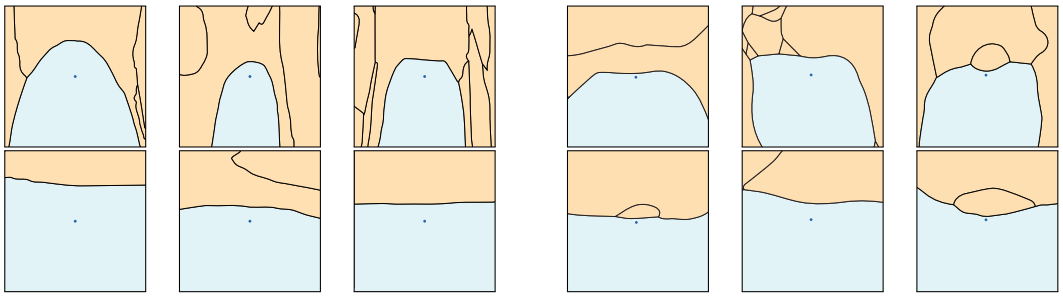


Figure 6.7 – Visualization of normal cross-sections of the decision boundary, for CIFAR-10 (Left: LeNet, Right: ResNet-18). **Top:** Normal cross-sections along $(\boldsymbol{r}(\boldsymbol{x}), \boldsymbol{v})$, where $\boldsymbol{v}$ is the universal perturbation computed using Algorithm 4 in Chapter 4. **Bottom:** Normal cross-sections along $(\boldsymbol{r}(\boldsymbol{x}), \boldsymbol{v})$, where $\boldsymbol{v}$ is a *random* vector uniformly sampled from the unit sphere in $\mathbb{R}^d$.

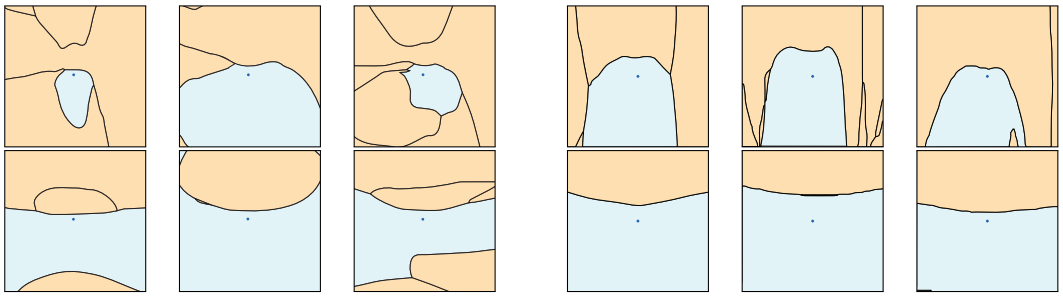


Figure 6.8 – Visualization of normal cross-sections of the decision boundary, for ImageNet (Left: ResNet-152, and Right: CaffeNet) **Top:** Normal cross-sections along $(\boldsymbol{r}(\boldsymbol{x}), \boldsymbol{v})$, where $\boldsymbol{v}$ is the universal perturbation computed using the algorithm in Chapter 4. **Bottom:** Normal cross-sections along $(\boldsymbol{r}(\boldsymbol{x}), \boldsymbol{v})$, where $\boldsymbol{v}$ is a *random* vector uniformly sampled from the unit sphere in $\mathbb{R}^d$.

rates achieved with such a simple procedure (random generation in $\mathcal{S}_c$) confirms that the curvature is the governing factor that controls the robustness of classifiers to universal perturbations, as analyzed in Section 6.4. In fact, such high fooling rates cannot be achieved by only using the model of Section 6.3 (neglecting the curvature information), as illustrated in Fig. 6.9 (b). Specifically, by generating random perturbations from the subspace $\mathcal{S}_f$ collecting normal vectors $\boldsymbol{r}(\boldsymbol{x})$ (which is the procedure that is suggested by Theorem 3 to compute universal perturbations, without taking into account second order information), the best universal perturbation achieves a fooling rate of 65%, which is significantly worse than if the curvature is used to craft the perturbation. It can be seen that, similarly to Fig. 6.9 (b), the proposed approach of generating universal perturbations through random sampling from the subspace $\mathcal{S}_c$ achieves high fooling rates (comparable to the algorithm in Chapter 4, and significantly higher than by using $\mathcal{S}_f$).

Fig 6.10 illustrates a universal perturbation for ImageNet, corresponding to the maximally curved shared direction (or in other words, the maximum eigenvalue of $\overline{H}$ computed

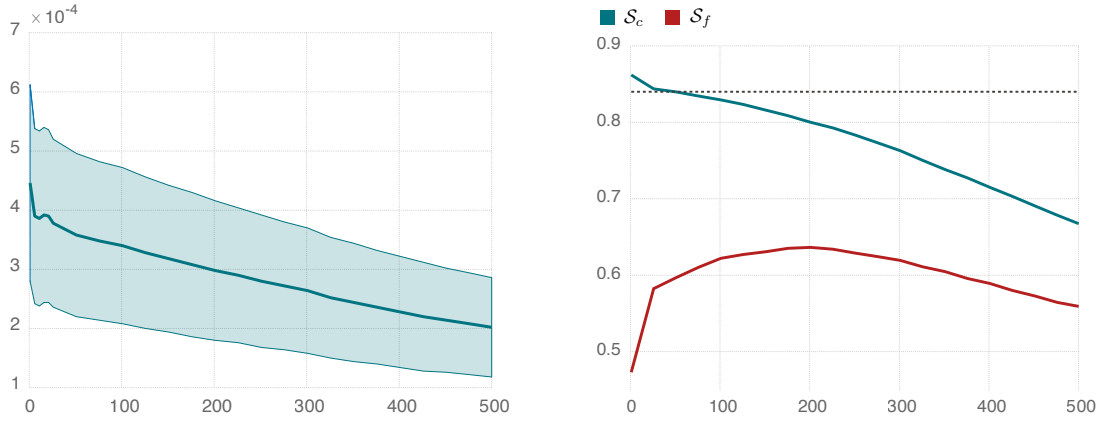


Figure 6.9 – **(a)** Average curvature $\overline{\kappa}_{\mathcal{S}}$, averaged over 1000 *validation* datapoints, as a function of the subspace dimension. **(b)** Fooling rate of universal perturbations (on an unseen *validation* set) computed using random perturbations in 1) $\mathcal{S}_c$: the subspace of positively curved directions, and 2) $\mathcal{S}_f$: the subspace collecting normal vectors $\boldsymbol{r}(\boldsymbol{x})$. The dotted line corresponds to the fooling rate using the algorithm in Chapter 4. $\mathcal{S}_f$ corresponds to the largest singular vectors corresponding to the matrix gathering the *normal vectors* $\boldsymbol{r}(\boldsymbol{x})$ in the training set (similar to the approach in [68]).

using $n = 200$ random samples).[7] The CaffeNet architecture is used, and Fig. 6.10 also represents sample perturbed images that fool the classifier. Just like the universal perturbation computed using Algorithm 4 of Chapter 4, the perturbations are not very perceptible, and lead to misclassification of most unseen images in the validation set. For this example on ImageNet, the fooling rate of this perturbation is 67.2% on the validation set. This is significantly larger than the fooling rate of the perturbation computed using $\mathcal{S}_f$ only (38%), but lower than that of Algorithm 4 (85.4%). We hypothesize that this gap for ImageNet is partially due to the small number of samples, which was made due to computational restrictions.

Figure 6.10 – Left column: Universal perturbation computed through random sampling from $\mathcal{S}_c$. Second column to end: All images are (incorrectly) classified as "bubble". The CaffeNet architecture is used. Similarly to Chapter 4, the perturbation is constrained to have ℓ_2 norm of $2,000$.

The existence of this subspace $\mathcal{S}_c$ (and that universal perturbations are random vectors in $\mathcal{S}_c$) further explains the high diversity of universal perturbations. Fig. 6.11 illustrates

[7] We used $m = 1$ in this experiment as the matrix $\overline{H}$ is prohibitively large for ImageNet.

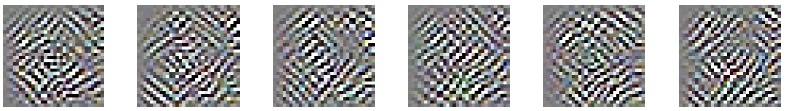


Figure 6.11 – Diversity of universal perturbations randomly sampled from the subspace $\mathcal{S}_c$. The normalized inner product between two perturbations is less than 0.1.

different universal perturbations for CIFAR-10 computed by sampling random directions from $\mathcal{S}_c$. The diversity of such perturbations justifies why re-training with perturbed images (as done in Section 4.5) does *not* significantly improve the robustness of such networks, as other directions in $\mathcal{S}_c$ can still lead to universal perturbations, even if the network becomes robust to some directions.

6.5 Transferability of universal perturbations

It is interesting to note that the subspace $\mathcal{S}_c$ is likely to be shared not only across datapoints, but also different networks (to some extent). To support this claim, Fig. 6.12 shows the cosine of the principal angles between subspaces $\mathcal{S}_c^{\text{LeNet}}$ and $\mathcal{S}_c^{\text{NiN}}$, computed for LeNet and NiN [56] models. Note that the first principal angles between the two subspaces are very small, leading to shared directions between the two subspaces.

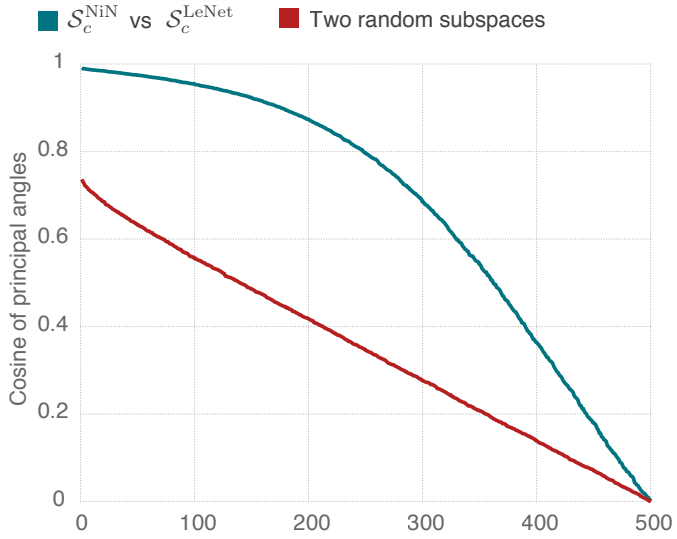


Figure 6.12 – Cosine of principal angles between $\mathcal{S}_c^{\text{LeNet}}$ and $\mathcal{S}_c^{\text{NiN}}$. For comparison, cosine of angles between two random subspaces is also shown.

For networks trained on ImageNet, Fig. 6.13 shows examples of normal cross-sections of the decision boundary across a *fixed* direction in $\mathcal{S}_c$, for the VGG-16 architecture (but where $\mathcal{S}_c$ is computed for *CaffeNet*). Note that the decision boundary across this *fixed* direction is positively curved for both networks, albeit computing this subspace for a

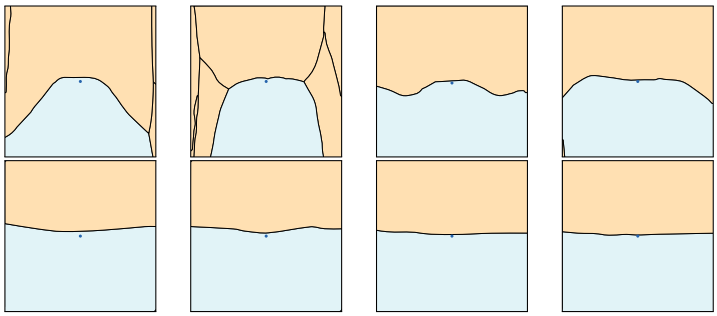


Figure 6.13 – Transferability of the subspace $\mathcal{S}_c$ across different *networks*. The first row shows normal cross sections along a fixed direction in $\mathcal{S}_c$ for VGG-16, with a subspace $\mathcal{S}_c$ computed with CaffeNet. Note the positive curvature in most cases. To provide a baseline for comparison, the second row illustrates normal sections along random directions.

distinct network. The sharing of $\mathcal{S}_c$ across different nets explains the transferability of universal perturbations observed in Chapter 4.

6.6 Conclusion

In this chapter, we analyzed the robustness of classifiers to universal perturbations, under two decision boundary models: Locally flat and curved. We showed that the first are not robust to universal directions, provided the normal vectors in the vicinity of natural images are correlated. While this model explains the vulnerability for e.g., linear classifiers, this model discards the curvature information, which is essential to fully analyze the robustness of deep nets to universal perturbations. The second, classifiers with *curved* decision boundaries, are instead not robust to universal perturbations, provided the existence of a shared subspace along which the decision boundary is positively curved (for most directions). We empirically verify these assumptions for deep networks. Our analysis hence explains the existence of universal perturbations, and further provides a purely geometric approach for computing such perturbations, in addition to explaining properties of perturbations, such as their diversity.

Our analysis hence shows that to construct classifiers that are robust to universal perturbations, it is key to *suppress* this subspace of shared curved directions, which can possibly be done through regularization of the objective function.

In Part III, we will show how geometric analysis can be deliberately used to construct robust classifiers. The study of the curvature of classifiers can particularly lead to effective geometric regularization techniques that significantly improve adversarial robustness of state-of-the-art classifiers. Furthermore, one can find geometric characterizations of adversarial examples based on the curvature of the decision boundary in order to devise effective methods to detect adversarial examples.

Part III Applications

7 Topology and geometry of decision regions

"Geometry is not true, it is advantageous."
— Henri Poincaré

7.1 Introduction

In previous chapters, we provided geometric analysis of classifiers' robustness to different perturbation regimes. In particular, we showed in Chapter 5 that the vulnerability of deep networks to semi-random perturbations suggests that their decision boundary possess a small curvature. On the other hand, in Chapter 6, the existence of universal perturbations is attributed to the high curvature of the decision boundary of deep networks. We resolve this seeming paradox by providing a thorough empirical analysis of the local geometry of the decision boundary of state-of-the-art deep networks. To perform such analysis, we employ the algorithms developed in Part I in order to sample points on the decision boundary of deep networks, in the vicinity of data samples. Besides using adversarial perturbations to study the geometric properties of the decision boundary, we show an application of adversarial perturbations in studying the connectedness of the decision regions of deep networks.

In this chapter, we specifically view classification regions as topological spaces and decision boundaries as hypersurfaces, and we examine their geometric properties. We first study the classification regions induced by state-of-the-art deep networks, and provide empirical evidence suggesting that these classification regions are *connected*; that is, there exists a continuous path that remains in the region between any two points of the same label. Up to our knowledge, this represents the first instance where the connectivity of classification regions is empirically shown. Then, to study the complexity of the functions learned by

Part of this chapter has been published in
Alhussein Fawzi*, Seyed-Mohsen Moosavi-Dezfooli*, Pascal Frossard, and Stefano Soatto. Empirical study of the topology and geometry of deep networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. (*: Equal contribution)

the deep network, we analyze the curvature of their decision boundary. We empirically show that

- The decision boundary in the vicinity of natural images is flat in most directions, with only a very few directions that are significantly curved.
- We reveal the existence of a fundamental asymmetry in the decision boundary of deep networks, whereby the decision boundary (near natural images x) is biased towards negative curvatures.[1]
- Directions with significantly curved decision boundaries are shared between different datapoints.
- We demonstrate the existence of a relation between the sensitivity of a classifier to perturbations of the inputs, and these shared directions: a deep net is vulnerable to perturbations along these directions, and is insensitive to perturbations along the remaining directions.

We finally leverage the fundamental asymmetry of deep networks revealed in our analysis, and propose an algorithm to detect natural images from imperceptibly similar images with very small adversarial perturbations [86], as well as estimate the correct label of these perturbed samples. We show that our purely geometric characterization of (small) adversarial examples, which does not involve any re-training, is very effective to recognize perturbed samples.

7.2 Definitions and notations

Let $f : \mathbb{R}^d \to \mathbb{R}^L$ denote a L class classifier. Given a datapoint $\boldsymbol{x}_0 \in \mathbb{R}^d$, the estimated label is obtained by $\hat{k}(\boldsymbol{x}_0) = \arg\max_k f_k(\boldsymbol{x}_0)$, where $f_k(\boldsymbol{x})$ is the k^{th} component of $f(\boldsymbol{x})$ that corresponds to the k^{th} class. The classifier f partitions the space $\mathbb{R}^d$ into *classification regions* $\mathcal{R}_1, \ldots, \mathcal{R}_L$ of constant label. That is, for any $\boldsymbol{x} \in \mathcal{R}_i$, $\hat{k}(\boldsymbol{x}) = i$. For a neighboring class j, the pairwise decision boundary of the classifier (between these two classes i and j) is defined as the set $\mathscr{B} = \{ \boldsymbol{z} : F(\boldsymbol{z}) = 0\}$, where $F(\boldsymbol{z}) = f_i(\boldsymbol{z}) - f_j(\boldsymbol{z})$ (we omit dependence on i,j for simplicity). The decision boundary defines a hypersurface (of dimension $d-1$) in $\mathbb{R}^d$. Note that for any point on the decision boundary $\boldsymbol{z} \in \mathscr{B}$, the gradient $\nabla F(\boldsymbol{z})$ is orthogonal to the tangent space $\mathcal{T}_{\boldsymbol{z}}(\mathscr{B})$ of $\mathscr{B}$ at $\boldsymbol{z}$ (see Fig. 7.5 (a) for an illustration).

In this chapter, we are interested in studying the decision boundary of a deep neural network in the vicinity of natural images. To do so, for a given point $\boldsymbol{x}$, we define the

[1] Throughout the chapter, the sign of the curvature is chosen according to the normal vector, and the data point x, as illustrated in Fig. 7.8 (top).

mapping $\boldsymbol{r}(\boldsymbol{x})$, given by

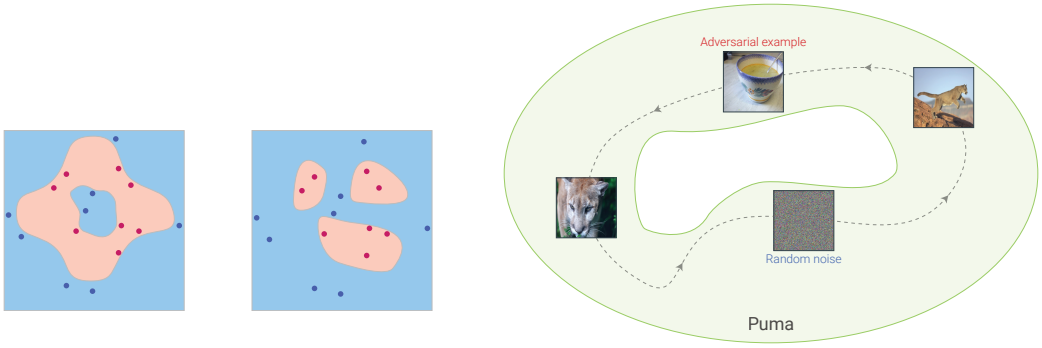
$$\boldsymbol{r}(\boldsymbol{x}) = \underset{\boldsymbol{r}\in\mathbb{R}^d}{\arg\min} \|\boldsymbol{r}\|_2 \text{ subject to } \hat{k}(\boldsymbol{x}+\boldsymbol{r}) \neq \hat{k}(\boldsymbol{x}), \tag{7.1}$$

which corresponds to the smallest perturbation required to misclassify image $\boldsymbol{x}$. Note that $\boldsymbol{r}(\boldsymbol{x})$ corresponds geometrically to the vector of minimal norm required to reach the decision boundary of the classifier, and is often dubbed an *adversarial perturbation* [86]. It should further be noted that, due to simple optimality conditions, $\boldsymbol{r}(\boldsymbol{x})$ is orthogonal to the decision boundary at $\boldsymbol{x} + \boldsymbol{r}(\boldsymbol{x})$.

In the remainder of this chapter, our goal is to analyze the geometric properties of classification regions $\{\mathcal{R}_i\}$ and decision boundaries $\mathscr{B}$ of deep networks. In particular, we study the connectedness of classification regions in Sec. 7.3, and the curvature of decision boundaries in Sec. 7.4, and draw a connection with the robustness of classifiers. We then use the developed geometric insights, and propose a method in Sec. 7.5 to detect artificially perturbed data points, and improve the robustness of classifiers.

7.3 Topology of classification regions

Do deep networks create shattered and disconnected classification regions, or on the contrary, one large connected region per label (see Fig. 7.1a)? While ReLU networks have an exponential number of linear regions (with respect to the number of layers) in the input space [67], it remains unclear whether deep nets create one connected region per class, or shatters a classification region around a large number of small connected sets. In the following, we treat the regions $\mathcal{R}_i$ as topological spaces, and study their path connectness. We formally cast the problem of connectivity of classification regions as follows: given any two data points $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{R}_i$, does a continuous curve $\gamma : [0,1] \to \mathcal{R}_i$ exist, such that $\gamma(0) = \boldsymbol{x}_1, \gamma(1) = \boldsymbol{x}_2$? The problem is complex to address theoretically; we therefore propose a heuristic method to study this question. To assess the connectivity of regions, we propose a path finding algorithm between two points belonging to the same classification region. That is, given two points $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d$, our proposed approach attempts to construct a piecewise linear path $\mathcal{P}$ that remains in the classification region. The path $\mathcal{P}$ is represented as a finite set of anchor points $(\boldsymbol{p}_0 = \boldsymbol{x}_1, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_n, \boldsymbol{p}_{n+1} = \boldsymbol{x}_2)$, where a convex path is taken between two consecutive points. To find the path (i.e., the anchor points), the algorithm first attempts to take a convex path between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$; when the path is not entirely included in the classification region, the path is modified by projecting the midpoint $\boldsymbol{p} = (\boldsymbol{x}_1 + \boldsymbol{x}_2)/2$ onto the target classification region. The same procedure is applied recursively on the two segments of the path $(\boldsymbol{x}_1, \boldsymbol{p})$ and $(\boldsymbol{x}_2, \boldsymbol{p})$ till the whole path is entirely in the region. The algorithm is summarized in Algorithm 5. In practice, the validity of a path $\mathcal{P}$ is checked empirically through a fine sampling of the convex combinations of the consecutive anchor points. Specifically, we set in practice the distance between sampled points to four orders of magnitude smaller than the distance



(a) (b)

Figure 7.1 – **(a)** Disconnected versus connected yet complex classification regions. **(b)** All four images are classified as puma. There exists a path between two images classified with the same label.

between the original two images.

Algorithm 5 Finding a path between two data points.

function FINDPATH($\boldsymbol{x}_1$,$\boldsymbol{x}_2$)
  // input: Datapoints $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d$.
  // output: Path $\mathcal{P}$ represented by anchor points.
  $\boldsymbol{x}_\mathrm{m} \leftarrow (\boldsymbol{x}_1 + \boldsymbol{x}_2)/2$
  if $\hat{k}(\boldsymbol{x}_\mathrm{m}) \neq \hat{k}(\boldsymbol{x}_1)$ **then**
    $\boldsymbol{r} \leftarrow \arg\min_{\boldsymbol{r}} \|\boldsymbol{r}\|_2$ s.t. $\hat{k}(\boldsymbol{x}_\mathrm{m} + \boldsymbol{r}) = \hat{k}(\boldsymbol{x}_1)$
    $\boldsymbol{x}_\mathrm{m} \leftarrow \boldsymbol{x}_\mathrm{m} + \boldsymbol{r}$
  end if
  $\mathcal{P} \leftarrow (\boldsymbol{x}_1, \boldsymbol{x}_\mathrm{m}, \boldsymbol{x}_2)$
  // Check the validity of the path by sampling in the convex combinations of consecutive anchor points
  if $\mathcal{P}$ is a valid path **then**
    return $\mathcal{P}$
  end if
  $\mathcal{P}_1 \leftarrow$FINDPATH($\boldsymbol{x}_1$,$\boldsymbol{x}_\mathrm{m}$)
  $\mathcal{P}_2 \leftarrow$FINDPATH($\boldsymbol{x}_\mathrm{m}$,$\boldsymbol{x}_2$)
  $\mathcal{P} \leftarrow$ **concat**$(\mathcal{P}_1, \mathcal{P}_2)$
  return $\mathcal{P}$
end function

The proposed approach is used to assess the connectivity of the CaffeNet architecture[2] [44] on the ImageNet classification. To do so, we examine the existence of paths between

1. Two randomly sampled points from the validation set with the same estimated label,

2. A randomly sampled point from the validation set, and an adversarially perturbed

[2]We tested other architectures (GoogLeNet, VGG-19, ResNet-152), and the results were similar to CaffeNet. We therefore report only results on CaffeNet in this section.

image [86]. That is, we consider $\boldsymbol{x}_1$ to be an image from the validation set, and $\boldsymbol{x}_2 = \tilde{\boldsymbol{x}}_2 + \boldsymbol{r}$, where $\tilde{\boldsymbol{x}}_2$ corresponds to an image classified differently than $\boldsymbol{x}_1$. $\boldsymbol{x}_2$ is however classified similarly as $\boldsymbol{x}_1$, due to the targeted perturbation $\boldsymbol{r}$.

3. A randomly sampled point from the validation set, and a perturbed random point. This is similar to scenario 2, but $\tilde{\boldsymbol{x}}_2$ is set to be a random image (i.e., an image sampled uniformly at random from the sphere $\rho\mathbb{S}^{d-1}$, where ρ denotes the typical norm of images).

Note that in all scenarios, we check the connectivity between two images that have the same *estimated* label by the classifier (but not necessarily the same true label). In particular, in scenario 2 and 3, $\boldsymbol{x}_2$ does not even visually correspond to an image of the same class as $\boldsymbol{x}_1$ (but has the same estimated label as $\boldsymbol{x}_1$ by the classifier). With this setting, the geometric properties of the classification regions are analyzed independently of the visual properties of the images. These scenarios are illustrated in Fig. 7.1b. For each scenario, 1,000 pairs of points are considered, and the approach described above is used to find the connecting path. Our results can be stated as follows:

1. In all three scenarios, evidence hints that **a continuous path included in the region always exists** between points sampled from the same classification region.[3]

2. Moreover, the continuous path connecting the points is **approximately a straight path**.

The first result suggests that the classification regions created by deep neural networks are *connected* in $\mathbb{R}^d$: deep nets create single large regions containing all points of the same label. This goes against common belief, whereby classification regions are thought to be disconnected, and to concentrate around data points. To further understand the paths found by Algorithm 5, we show in Fig. 7.2 (right) an illustrative example of a path connecting two data points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ from the validation set (i.e., scenario 1). While this nonstraight path connecting $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ is entirely included in the classification region, observe that the convex path illustrated in Fig. 7.2 (left) is *not* a valid path. In practice, the paths found by Algorithm 5 have, in average, 10 anchor points for the three scenarios.

Our second result provides an answer to the next natural question: how do the paths connecting data points (and staying inside a classification region) "look like"? Specifically, we show that two points in a classification region can be connected by an *approximately straight path*. To quantify how the paths of Algorithm 5 deviate from the straight path,

[3] While not providing a formal certificate that the *continuous* path is entirely included in the classification region (as boundary regions can meander between neighbouring points in the continuum), we believe the sampling procedure used to verify the connectedness of a region is conservative, especially in the presence of regularizers that bound the curvature of the decision boundary (e.g., weight decay).

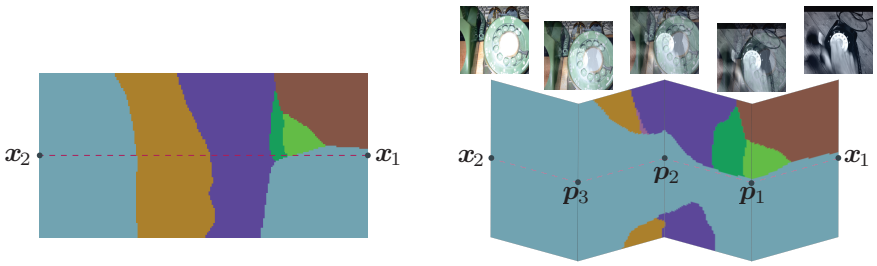


Figure 7.2 – Classification regions (shown with different colors), and illustration of different paths between images $\boldsymbol{x}_1$, $\boldsymbol{x}_2$. **Left:** Example where the convex path between two datapoints is not entirely included in the classification region (note that the linear path traverses 4 other regions, each depicted with a different color). The image is the cross-section spanned by $\boldsymbol{r}(\boldsymbol{x}_1)$ (adversarial perturbation of $\boldsymbol{x}_1$) and $\boldsymbol{x}_1 - \boldsymbol{x}_2$. Images $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are natural images from the ILSVRC 12 validation set, and the CaffeNet deep network is used. **Right:** Illustration of the classification regions along a nonconvex path; observe that the path entirely remains in the same classification region. The illustration is obtained by stitching cross-sections spanned by $\boldsymbol{r}(\boldsymbol{x}_1)$ (vertical axis) and $\boldsymbol{p}_i - \boldsymbol{p}_{i+1}$ (two consecutive anchor points in the path $\mathcal{P}$) (horizontal axis). It is shown broken to emphasize that the horizontal axes are different. Angles between stitched cross-sections are purely illustrative. On top of each anchor point (as well as $\boldsymbol{x}_1, \boldsymbol{x}_2$), image on the path is visualized.

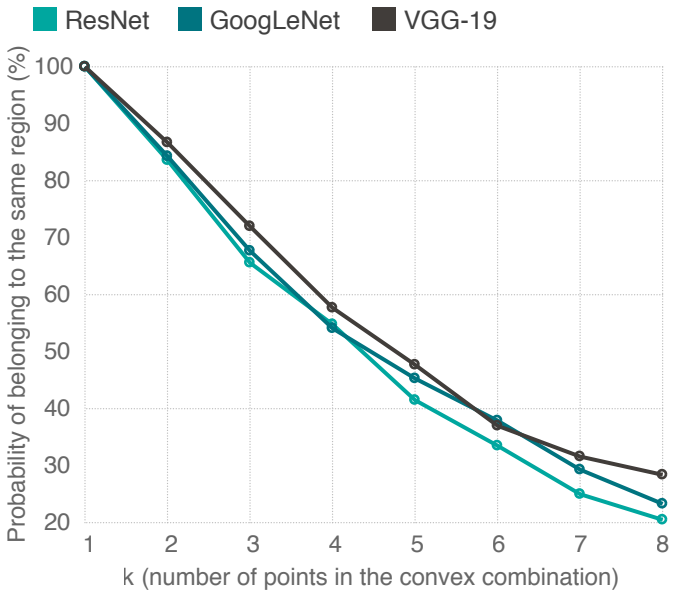


Figure 7.3 – Empirical probability (y axis) that a convex combination of k samples (x axis) from the same classification region stays in the region, for networks trained on ImageNet. Samples are randomly chosen from the validation set.

we report the quantity

$$D(\boldsymbol{p}) = \frac{\sum_{i=0}^{n} \|\boldsymbol{p}_i - \boldsymbol{p}_{i+1}\|_2}{\|\boldsymbol{p}_0 - \boldsymbol{p}_{n+1}\|_2}.$$

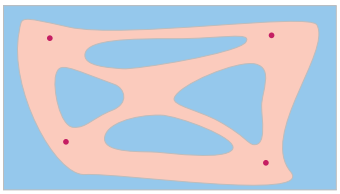


Figure 7.4 – Schematic illustration in 2d of the properties of a classification region of a deep net. A classification region is connected by an almost convex path, despite classification regions being *non-convex* sets.

Values of $D(\boldsymbol{p}) \approx 1$ indicate that the path $\boldsymbol{p}$ is close to a straight line. For the three scenarios, we have an average deviation $D(\boldsymbol{p}) = 1 + 10^{-4}$, which indicates that the paths found in Algorithm 5 are slight deviations from the straight path. With this very small deviation from the straight path, it is possible to connect arbitrary points in classification regions.[4] This observation is intriguingly similar to that of [34], where it is shown that solutions (in the *weight space*) achieving small error can be connected with an approximately straight path. This suggests that the data space and weight space have common properties; the specifics of this duality between these spaces is outside the scope of this thesis and will be subject of future work.

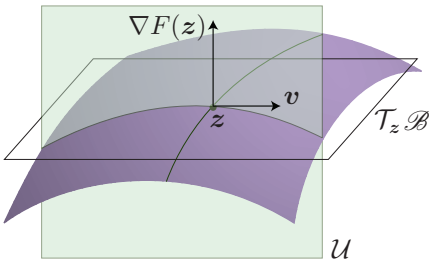
Despite the existence of approximately straight paths connecting any pairs of points in the classification regions, it is important to note that classification regions are *not* convex bodies. In fact, Fig. 7.3 illustrates the estimated probability that random convex combinations of k images $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k \in \mathcal{R}_i$ belong to $\mathcal{R}_i$. Observe that for the different tested networks, random convex combinations of two images (i.e., case where $k = 2$) belong with probability ≈ 0.8 to the classification region. However, for larger k, this probability gets much smaller, which implies that classification regions are *not* convex bodies in $\mathbb{R}^d$. These results suggest that the classification regions of deep networks extrapolate their classification regions in an approximately flat way between different images (i.e., there exist near-convex paths between pairs of images of the same class), but that the classification region is *not* a convex body. In a simplistic two-dimensional world, a classification region satisfying these two constraints would look like Fig. 7.4.

In the next section, we explore the *complexity* of the boundaries of these classification regions learned by deep networks, through their curvature property.

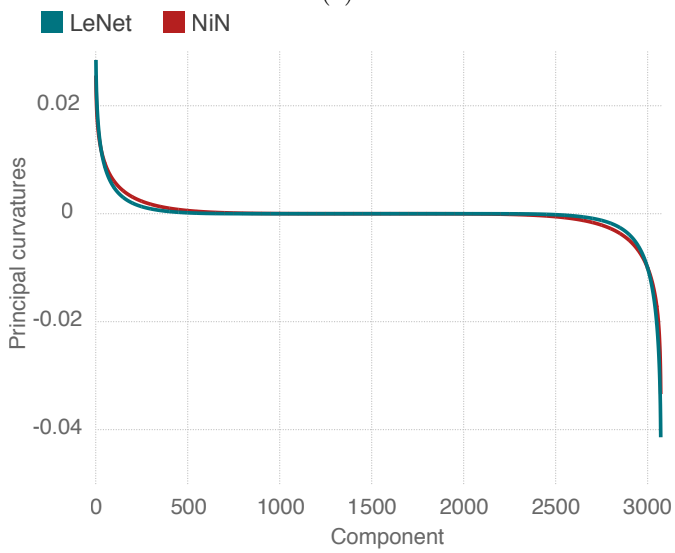
7.4 Curvature of the decision boundaries

We start with basic definitions of curvature. The *normal* curvature $\kappa(\boldsymbol{z}, \boldsymbol{v})$ along a tangent direction $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{z}}(\mathscr{B})$ is defined as the curvature of the planar curve resulting from the

[4] Straight paths might not be entirely inside the classification region; tiny deviations are crucial to guarantee that complete paths are inside the region.



(a)



(b)

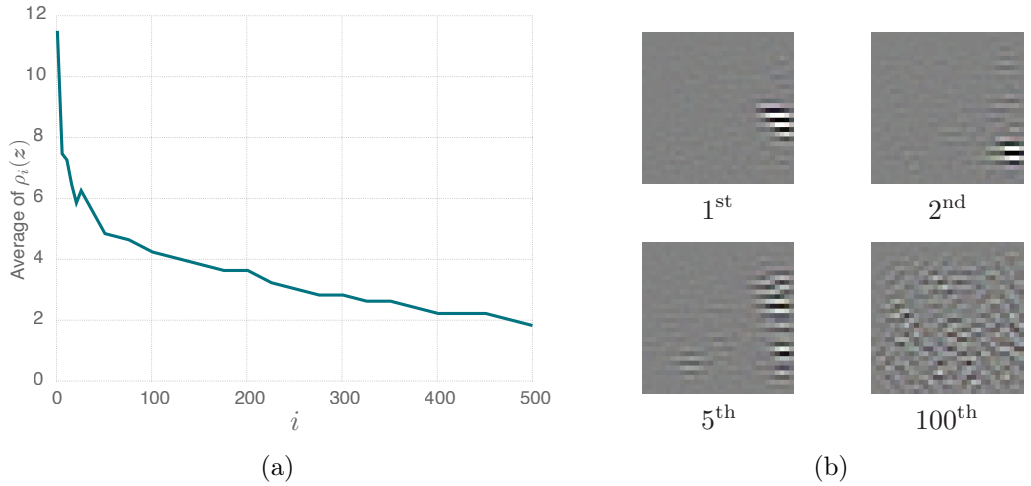
Figure 7.5 – **(a)** Normal section $\mathcal{U}$ of the decision boundary, along the plane spanned by the normal vector $\nabla F(\boldsymbol{z})$ and $\boldsymbol{v}$. **(b)** Principal curvatures for NiN and LeNet, computed at a point $\boldsymbol{z}$ on the decision boundary in the vicinity of a natural image.

cross-section of $\mathscr{B}$ along the two-dimensional normal plane spanning $(\nabla F(\boldsymbol{z}), \boldsymbol{v})$ (see Fig. 7.5a for details). The curvature along a tangent vector $\boldsymbol{v}$ can be expressed in terms of the Hessian matrix H_F of F [54]:

$$\kappa(\boldsymbol{z}, \boldsymbol{v}) = \frac{\boldsymbol{v}^T H_F \boldsymbol{v}}{\|\boldsymbol{v}\|_2^2 \|\nabla F(\boldsymbol{z})\|_2}. \tag{7.2}$$

Principal directions correspond to the orthogonal directions in the tangent space maximizing the curvature $\kappa(\boldsymbol{z}, \boldsymbol{v})$. Specifically, the l-th principal direction $\boldsymbol{v}_l$ (and the corresponding principal curvature κ_l) is obtained by maximizing $\kappa(\boldsymbol{z}, \boldsymbol{v})$ with the constraint $\boldsymbol{v}_l \perp \boldsymbol{v}_1 \dots \boldsymbol{v}_{l-1}$. Alternatively, the principal curvatures correspond to the nonzero eigenvalues of the matrix $\frac{1}{\|\nabla F(\boldsymbol{z})\|_2} P H_F P$, where P is the projection operator on the tangent space; i.e., $P = I - \nabla F(\boldsymbol{z}) \nabla F(\boldsymbol{z})^T$.

We now analyze the curvature of the decision boundary of deep neural networks in the



(a) (b)

Figure 7.6 – **(a)** Average of $\rho_i(\boldsymbol{z})$ as a function of i for different points $\boldsymbol{z}$ in the vicinity of natural images. **(b)** Basis of $\mathcal{S}$.

vicinity of natural images. We consider the LeNet and NiN [56] architectures trained on the CIFAR-10 task, and show the principal curvatures of the decision boundary, in the vicinity of 1,000 randomly chosen images from the validation set. Specifically, for a given image $\boldsymbol{x}$, the perturbed sample $\boldsymbol{z} = \boldsymbol{x} + \boldsymbol{r}(\boldsymbol{x})$ corresponds to the closest point to $\boldsymbol{x}$ on the decision boundary. We then compute the principal curvatures at point $\boldsymbol{z}$ with Eq. 7.2. The average profile of the principal curvatures (over $1,000$ data points) is illustrated in Fig. 7.5b. Observe that, for both networks, the large majority of principal curvatures are approximately zero: along these principal directions, the decision boundary is almost flat. Along the remaining principal directions, the decision boundary has (non-negligible) positive or negative curvature. Interestingly, the principal curvature profile is asymmetric towards *negatively curved* directions. We have consistently observed this asymmetry in different settings: different datapoints, different networks (e.g., LeNet and NiN), and even different datasets (CIFAR-10 and ImageNet, see Section 7.5 for more details), which suggests that this property (negatively curved decision boundary) is not an artifact of the experimental setting. In the next section, we leverage this characteristic asymmetry of the decision boundaries of deep neural networks (in the vicinity of natural images) to detect adversarial examples from clean examples.

While the above local analysis shows the existence of few directions along which the decision boundary is curved, we now examine whether these directions are *shared* across different datapoints, and relate these directions with the robustness of deep nets. To estimate the *shared* common curved directions, we compute the largest *principal directions* for a randomly chosen batch of 100 training samples and merge these directions into a matrix M. We then estimate the common curved directions as the m largest singular vectors of M that we denote by $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m$. To assess whether the decision boundary is curved in such directions, we then evaluate the curvature of the decision boundary

in such directions for points $\boldsymbol{z}$ in the vicinity of *unseen* samples from the *validation* set. That is, for $\boldsymbol{x}$ in the validation set, and $\boldsymbol{z} = \boldsymbol{x} + \boldsymbol{r}(\boldsymbol{x})$, we compute

$$\rho_i(\boldsymbol{z}) = \frac{|\boldsymbol{u}_i^T P H_F P \boldsymbol{u}_i|}{\mathop{\mathbb{E}}_{\boldsymbol{v} \sim \mathbb{S}^{d-1}} (|\boldsymbol{v}^T P H_F P \boldsymbol{v}|)}, \tag{7.3}$$

which measures how relatively curved is the decision boundary in direction $\boldsymbol{u}_i$, compared to random directions sampled from the unit sphere in $\mathbb{R}^d$. When $\rho_i(\boldsymbol{z}) \gg 1$, this indicates that $\boldsymbol{u}_i$ constitutes a direction that significantly curves the decision boundary at $\boldsymbol{z}$. Fig. 7.6a shows the average of $\rho_i(\boldsymbol{z})$ over 1,000 points $\boldsymbol{z}$ on the decision boundary in the vicinity of *unseen* natural images, for the LeNet architecture on CIFAR-10. Note that the directions $\boldsymbol{u}_i$ (with i sufficiently small) lead to universally curved directions across *unseen* points. That is, the decision boundary is highly curved along such data-independent directions. Note that, despite using a relatively small number of samples (i.e., 100 samples) to compute the shared directions, these generalize well to unseen points. We illustrate in Fig. 7.6b these directions $\boldsymbol{u}_i$, along which decision boundary is universally curved in the vicinity of natural images; interestingly, the first principal directions (i.e., directions along which the decision boundary is highly curved) are very localized Gabor-like filters. Through discriminative training, the deep neural network has implicitly learned to curve the decision boundary along such directions, and preserve a flat decision boundary along the orthogonal subspace.

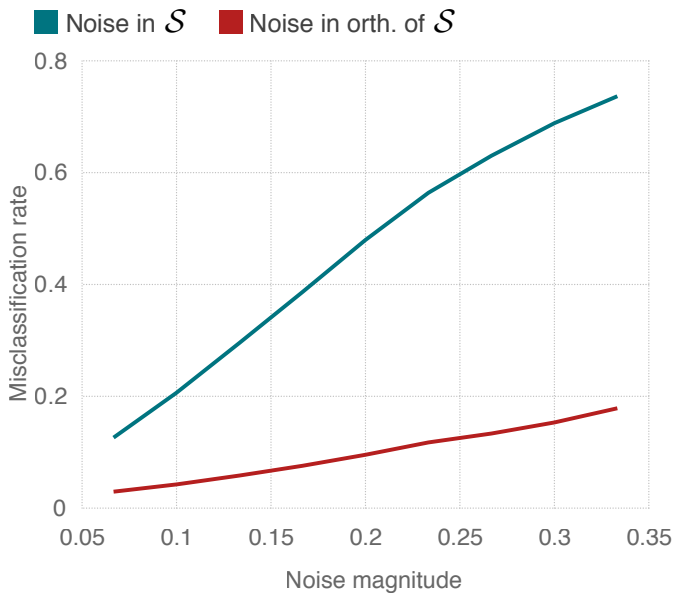


Figure 7.7 – Misclassification rate (% of images that change labels) on the noisy validation set, w.r.t. the noise magnitude (ℓ_2 norm of noise divided by the typical norm of images).

Interestingly, the data-independent directions $\boldsymbol{u}_i$ (where the decision boundary is highly curved) are also tightly connected with the *sensitivity* of the classifier to perturbations.

Table 7.1 – Norm of projected perturbation on $\mathcal{S}$, normalized by norm of perturbation: $\frac{\|P_{\mathcal{S}}\boldsymbol{v}\|_2}{\|\boldsymbol{v}\|_2}$, with $\boldsymbol{v}$ the perturbation. Larger values (i.e., closer to 1) indicate that the perturbation has a larger component on subspace $\mathcal{S}$.

Type of perturbation $\boldsymbol{v}$	LeNet [53]	NiN [56]
Random	0.25	0.25
Adversarial	0.64	0.60
$\boldsymbol{x}_2 - \boldsymbol{x}_1$	0.10	0.09
$\nabla \boldsymbol{x}$	0.22	0.24

To elucidate this relation, we construct a subspace $\mathcal{S} = \text{span}(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{200})$ containing the first 200 shared curved directions. Then, we show in Fig. 7.7 the accuracy of the CIFAR-10 LeNet model on a noisy validation set, where the noise either belongs to $\mathcal{S}$, or to $\mathcal{S}^\perp$ (i.e., orthogonal of $\mathcal{S}$). It can be observed that the deep network is much more robust to noise orthogonal to $\mathcal{S}$, than to noise in $\mathcal{S}$. Hence, $\mathcal{S}$ also represents the subspace of perturbations to which the classifier is highly vulnerable, while the classifier has learned to be invariant to perturbations in $\mathcal{S}^\perp$. To support this claim, we report in Table 7.1, the norm of the projection of adversarial perturbations (computed using the method in [70]) on the subspace $\mathcal{S}$, and compare it to that of the projection of random noise onto $\mathcal{S}$. Note that for both networks under study, adversarial perturbations project well onto the subspace $\mathcal{S}$ comparatively to random perturbations, which have a significant component in $\mathcal{S}^\perp$. In contrast, the perturbations obtained by taking the difference of two random images belong overwhelmingly to $\mathcal{S}^\perp$, which agrees with the observation drawn in Sec. 7.3 whereby straight paths are likely to belong to the classification region. Finally, note that the image gradient $\nabla \boldsymbol{x}$ (directional change in the intensity in image $\boldsymbol{x}$) also does not have an important component in $\mathcal{S}$, as the robustness to such directions is fundamental to achieve invariance to small geometric deformations.[5]

The importance of the shared directions $\{\boldsymbol{u}_i\}$, where the decision boundary is curved, hence goes beyond our curvature analysis, and capture the modes of sensitivity learned by the deep network.

7.5 Detection of perturbed samples

State-of-the-art image classifiers are highly vulnerable to imperceptible adversarial perturbations [86, 7]. That is, adding a well-sought small perturbation to an image causes state-of-the-art classifiers to misclassify. In this section, we leverage the asymmetry of the

[5] In fact, a first order Taylor approximation of a translated image $\boldsymbol{x}(\cdot + \tau_1, \cdot + \tau_2) \approx \boldsymbol{x} + \tau_1 \nabla_x \boldsymbol{x} + \tau_2 \nabla_y \boldsymbol{x}$. To achieve robustness to translations, a deep neural network hence needs to be locally invariant to perturbations along the gradient directions.

principal curvatures (illustrated in Fig. 7.5b), and propose a method to distinguish between original images, and images perturbed with small adversarial perturbations, as well as improve the robustness of classifiers. For an element $\boldsymbol{z}$ on the decision boundary, denote by $\overline{\kappa}(\boldsymbol{z}) = \frac{1}{d-1}\sum_{i=1}^{d-1}\kappa_i(\boldsymbol{z})$ the average of the principal curvatures. For points $\boldsymbol{z}$ sampled in the vicinity of natural images, the profile of the principal curvature is asymmetric (see Fig. 7.5b), leading to a negative average curvature; i.e., $\overline{\kappa}(\boldsymbol{z}) < 0$. In contrast, if $\boldsymbol{x}$ is now perturbed with an adversarial example (that is, we observe $\boldsymbol{x}_{\text{pert}} = \boldsymbol{x} + \boldsymbol{r}(\boldsymbol{x})$ instead of $\boldsymbol{x}$), the average curvature at the vicinity of $\boldsymbol{x}_{\text{pert}}$ is instead *positive*, as schematically illustrated in Fig. 7.8. Table 7.2 supports this observation empirically with adversarial examples computed with the method in [70]. Note that for both networks, the asymmetry of the principal curvatures allows to distinguish very accurately original samples from perturbed samples using the *sign* of the curvature.[6] Based on this simple idea, we now derive an algorithm for detecting adversarial perturbations.

Since the computation of all the principal curvatures is intractable for large-scale datasets, we derive a tractable estimate of the average curvature. Note that the average curvature $\overline{\kappa}$ can be equivalently written as $\mathop{\mathbb{E}}_{v\sim\mathbb{S}^{d-1}}\left(\boldsymbol{v}^T G(\boldsymbol{z})\boldsymbol{v}\right)$, where $G(\boldsymbol{z}) = \|\nabla F(\boldsymbol{z})\|_2^{-1}(I - \nabla F(\boldsymbol{z})\nabla F(\boldsymbol{z})^T)H_F(\boldsymbol{z})(I - \nabla F(\boldsymbol{z})\nabla F(\boldsymbol{z})^T)$. In fact, we have

$$\mathop{\mathbb{E}}_{\boldsymbol{v}\sim\mathbb{S}^{d-1}}\left(\boldsymbol{v}^T G(\boldsymbol{z})\boldsymbol{v}\right) = \mathop{\mathbb{E}}_{\boldsymbol{v}\sim\mathbb{S}^{d-1}}\left(\boldsymbol{v}^T\left(\sum_{i=1}^{d-1}\kappa_i\boldsymbol{v}_i\boldsymbol{v}_i^T\right)\boldsymbol{v}\right) = \frac{1}{d-1}\sum_{i=1}^{d-1}\kappa_i, \tag{7.4}$$

where $\boldsymbol{v}_i$ denote the principal directions. It therefore follows that the average curvature $\overline{\kappa}$ can be efficiently estimated using a sample estimate of $\mathop{\mathbb{E}}_{v\sim\mathbb{S}^{d-1}}\left(\boldsymbol{v}^T G(\boldsymbol{z})\boldsymbol{v}\right)$ (and without requiring the full eigen-decomposition of G). To further make the approach of detecting perturbed samples more practical, we approximate $G(\boldsymbol{z})$ (for $\boldsymbol{z}$ on the decision boundary) with $G(\boldsymbol{x})$, assuming that $\boldsymbol{x}$ is sufficiently close to the decision boundary.[7] This approximation avoids the computation of the closest point on the decision boundary $\boldsymbol{z}$, for each $\boldsymbol{x}$.

We provide the details in Algorithm 6. Note that, in order to extend this approach to multiclass classification, an empirical average is taken over the decision boundaries with respect to all other classes. Moreover, while we have used a threshold of 0 to detect adversarial examples from original data in the above explanation, a threshold parameter

[6] This idea might first appear counter-intuitive: if curvature is negative at the vicinity of data points, then the curvature has to be positive for data points lying on the other side of the boundary! However, natural data points are very "sparse" in $\mathbb{R}^d$; hence, two natural images never lie exactly opposite to each other (from the two sides of the boundary). Instead, different data points lie at the vicinity of very distinct parts of the decision boundary, which makes it possible to have negatively curved decision boundary at the vicinity of all data points. See Fig. 7.1a (left) for an illustration of such a decision boundary, with negative curvature at the vicinity of all points.

[7] The matrix G is never computed in practice, since only matrix vector multiplications of G are needed.

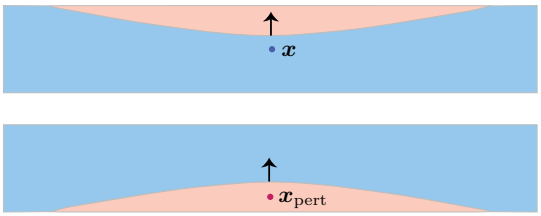


Figure 7.8 – Schematic representation of normal sections in the vicinity of a natural image (top), and perturbed image (bottom). The normal vector to the decision boundary is indicated with an arrow.

Table 7.2 – Percentage of points on the boundary with positive (resp. negative) average curvature, when sampled in the vicinity of natural images (resp. perturbed images). CIFAR-10 dataset is used; results are computed on the test set.

	LeNet	NiN
% $\overline{\kappa} > 0$ for original samples	97%	94%
% $\overline{\kappa} < 0$ for perturbed samples	96%	93%

t is used in practice (which controls the true positive vs. false positive tradeoff). Finally, it should be noted that in addition to detecting whether an image is perturbed, the algorithm also provides an estimate of the original label when a perturbed sample is detected (the class leading to the highest positive curvature is returned).

We now test the proposed approach on different networks trained on the ImageNet dataset [79], with adversarial examples computed using the approach in [70]. The latter approach is used as it provides small and difficult to detect adversarial examples, as mentioned in [65, 59]. Fig. 7.9 (left) shows the accuracy of the detection of Algorithm 6 on *original* images with respect to the detection error on *perturbed* images, for varying values of

Algorithm 6 Detecting and denoising perturbed samples.

input: classifier f, sample $\boldsymbol{x}$, threshold t.
output: boolean *perturbed*, recovered label *label*.
Set $F_i \leftarrow f_i - f_{\hat{k}}$ for $i \in [L]$.
Draw iid samples $\boldsymbol{v}_1, \dots, \boldsymbol{v}_T$ from the uniform distribution on $\mathbb{S}^{d-1}$.
Compute $\rho \leftarrow \frac{1}{LT} \sum_{\substack{i=1 \\ i \neq \hat{k}(\boldsymbol{x})}}^{L} \sum_{j=1}^{T} \boldsymbol{v}_j^T G_{F_i} \boldsymbol{v}_j$, where G_{F_i} denotes the Hessian of F_i projected on the tangent space; i.e., $G_{F_i}(\boldsymbol{x}) = \|\nabla F(\boldsymbol{x})\|_2^{-1} (I - \nabla F(\boldsymbol{x}) \nabla F(\boldsymbol{x})^T) H_{F_i}(\boldsymbol{x}) (I - \nabla F(\boldsymbol{x}) \nabla F(\boldsymbol{x})^T)$.
if $\rho < t$ **then** *perturbed* $\leftarrow$ *false*.
else *perturbed* $\leftarrow$ *false* and *label* $\leftarrow \underset{\substack{i \in \{1,\dots,L\} \\ i \neq \hat{k}(\boldsymbol{x})}}{\arg\max} \sum_{j=1}^{T} \boldsymbol{v}_j^T G_{F_i} \boldsymbol{v}_j$.
end if

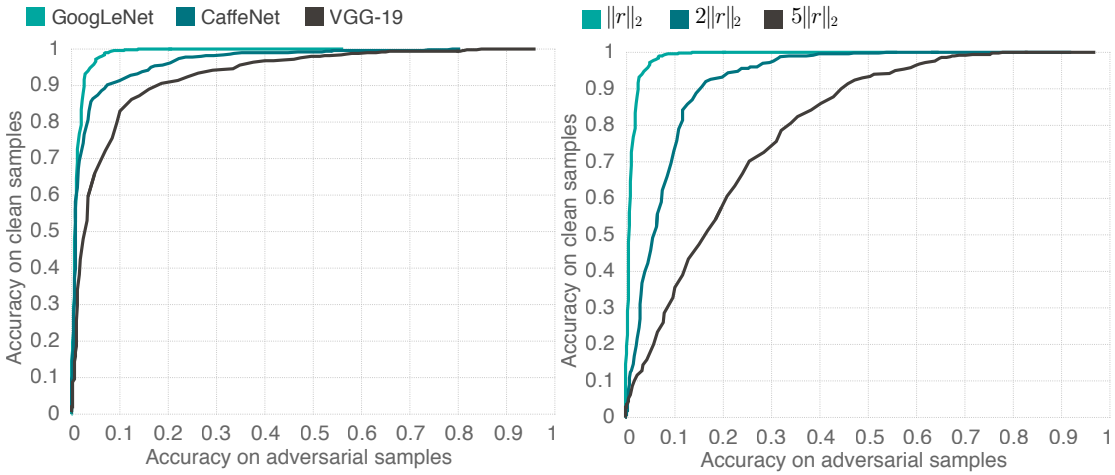


Figure 7.9 – True positives (i.e., detection accuracy on *clean samples*) vs. False positives (i.e., detection error on *perturbed samples*) on the ImageNet classification task. **Left:** Results reported for GoogLeNet, CaffeNet and VGG-19 architectures, with perturbations computed using the approach in [70]. **Right:** Results reported for GoogLeNet, where perturbations are scaled by a constant factor $\alpha = 1, 2, 5$.

the threshold t. For the three networks under test, the approach achieves very accurate detection of adversarial examples (e.g., more than 95% accuracy on GoogLeNet with an optimal threshold). Note first that the success of this strategy *confirms the asymmetry* of the curvature of the decision boundary on the more complex setting of large-scale networks trained on ImageNet. Moreover, this simple curvature-based detection strategy outperforms the detection approach recently proposed in [59]. In addition, unlike other approaches of detecting perturbed samples (or improving the robustness), our approach only uses the characteristic geometry of the decision boundary of deep neural networks (i.e., the curvature *asymmetry*), and does not involve any training/fine-tuning with perturbed samples, as commonly done.

The proposed approach not only distinguishes original from perturbed samples, but it also provides an estimate of the correct label, in the case a perturbed sample is detected. Algorithm 6 correctly recovers the labels of perturbed samples with an accuracy of 92%, 88% and 74% respectively for GoogLeNet, CaffeNet and VGG-19, with $t = 0$. This shows that the proposed approach can be effectively used to denoise the perturbed samples, in addition to their detection.

Finally, Fig. 7.9 (right) reports a similar graph to that of Fig. 7.9 (left) for the GoogLeNet architecture, where the perturbations are now multiplied by a factor $\alpha \geq 1$. Note that, as α increases, the detection accuracy of our method decreases, as it heavily relies on *local* geometric properties of the classifier (i.e., the curvature). Interestingly enough, [59, 65] report that the regime where perturbations are very small (like those produced by [70]) are the hardest to detect; we therefore foresee that this geometric approach will be used

along with other detection approaches, as it provides very accurate detection in a distinct regime where traditional detectors do not work well (i.e., when the perturbations are very small).

7.6 Conclusion

We analyzed in this chapter the geometry induced by deep neural network classifiers in the input space. Specifically, we provided empirical evidence showing that classification regions are connected. Next, to analyze the complexity of the functions learned by deep networks, we provided an empirical analysis of the curvature of the decision boundaries. We showed in particular that, in the vicinity of natural images, the decision boundaries learned by deep networks are flat along most (but not all) directions, and that some curved directions are *shared* across datapoints. We finally leveraged a fundamental observation on the *asymmetry* in the curvature of deep nets, and proposed an algorithm for detecting adversarially perturbed samples from original samples. This geometric approach was shown to be very effective, when the perturbations are sufficiently small, and that recovering the label was further possible using this algorithm. This shows that the study of the geometry of state-of-the-art deep networks is not only key from an analysis perspective, but it can also lead to classifiers with better properties.

Based on the analysis provided in this chapter, in [43], the curved directions of the decision boundary of state-of-the-art classifiers are identified as "the directions which they use to achieve their classification performance in the first place." Therefore, they conclude that the adversarial robustness and the generalization performance of deep networks are two sides of the same coin.

In the next chapter, we introduce a different approach to exploit the curvature of state-of-the-art deep classifiers in order to improve their robustness properties. We specifically introduce a curvature-based regularizer that can resemble the effect of adversarial training used in [35, 70, 62].

8 Design of robust networks

"Nature always tends to act in the simplest way."
— Daniel Bernoulli

8.1 Introduction

In addition to detect adversarial perturbations based on the curvature of the decision boundary as studied in the previous chapter, the curvature can be used as a leverage to improve the adversarial robustness of state-of-the-art deep networks. In this chapter, we specifically establish a link between adversarial training, as one of the most successful methods to improve robustness, and the curvature of the loss function and decision boundaries of the classifier. Our analysis results in a regularization technique that can significantly improve the robustness of state-of-the-art classifiers.

Adversarial training has recently been shown to be one of the most successful methods for increasing the robustness to adversarial perturbations of deep neural networks [35, 70, 62]. This approach consists in training the classifier on *perturbed* samples, with the aim of achieving higher robustness than a network trained on the original training set. Despite the importance and popularity of this training mechanism, the effect of adversarial training on the geometric properties of the classifier – its loss landscape with respect to the input and decision boundaries – is not well understood. In particular, how do the decision boundaries and loss landscapes of adversarially trained models compare to the ones trained on the original dataset?

In this chapter, we analyze such properties and show that one of the main effects of adversarial training is to induce a significant *decrease* in the curvature of the loss function and decision boundaries of the classifier. More than that, we show that such a geometric

Part of this chapter has been published in
Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. Robustness via curvature regularization, and vice versa. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

implication of adversarial training allows us to explain the high robustness of adversarially trained models. To support this claim, we follow a *synthesis* approach, where a new regularization strategy, Curvature Regularization (CURE), encouraging small curvature is proposed and shown to achieve robustness levels that are comparable to that of adversarial training. This highlights the importance of small curvature for improved robustness. In more detail, our contributions are summarized as follows:

- We empirically show that adversarial training induces a significant *decrease in the curvature* of the decision boundary and loss landscape *in the input space*.
- Using a quadratic approximation of the loss function, we establish upper and lower bounds on the robustness to adversarial perturbations with respect to the curvature of the loss. These bounds confirm the existence of a relation between low curvature and high robustness.
- Inspired by the implications of adversarially trained networks on the curvature of the loss function and our theoretical bounds, we propose an efficient regularizer that encourages small curvatures. On standard datasets (CIFAR-10 and SVHN), we show that the proposed regularizer leads to a significant boost of the robustness of neural networks, comparable to that of adversarial training.

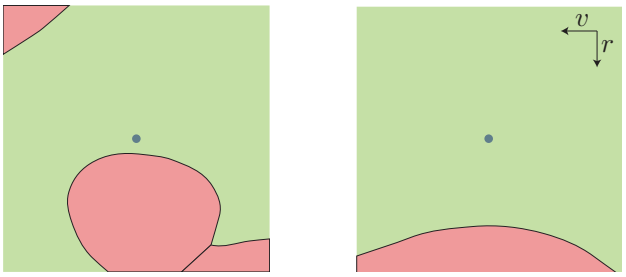
The latter step shows that the proposed regularizer can be seen as a more efficient alternative to adversarial training. More importantly, it shows that the effect of adversarial training on the curvature reduction is not a mere by-product, but rather a driving effect that causes the robustness to increase. We stress here that the main focus of this chapter is mainly on the latter – analyzing the geometry of adversarial training – rather than outperforming adversarial training.

8.2 Geometric analysis of adversarial training

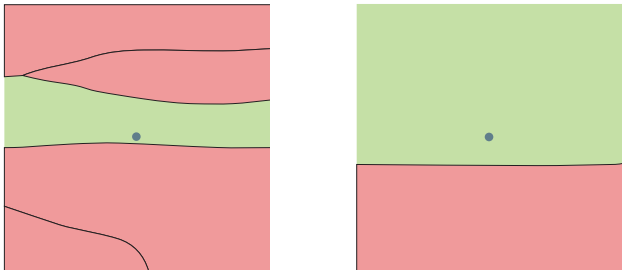
We start our analysis by inspecting the effect of adversarial training on the geometric properties of the decision boundaries of classifiers. To do so, we first compare qualitatively the decision boundaries of classifiers *with* and *without* adversarial training. Specifically, we examine the effect of *adversarial fine-tuning*, which consists in fine-tuning a trained network with a few extra epochs on adversarial examples.[1] We consider the CIFAR-10 [48] and SVHN [74] datasets, and use a ResNet-18 [38] architecture. For fine-tuning on adversarial examples, we use DeepFool [70].

Fig. 8.1 illustrates normal cross-sections of the decision boundaries before and after adversarial fine-tuning for classifiers trained on CIFAR-10 and SVHN datasets. Specifically,

[1] While adversarial fine-tuning is distinct from vanilla adversarial training, which consists in training on adversarial images *from scratch*, we use an adversarially fine-tuned network in this chapter as it allows to single out the effect of training on adversarial examples, as opposed to other uncontrolled phenomenon happening in the course of vanilla adversarial training.



(a) Original (CIFAR-10) (b) Finetuned (CIFAR-10)



(c) Original (SVHN) (d) Fine-tuned (SVHN)

Figure 8.1 – Random normal cross-sections of the decision boundary for ResNet-18 classifiers trained on CIFAR-10 (first row) and SVHN (second row). The first column is for classifiers trained on the original dataset, and the second column shows the boundaries after adversarial fine-tuning on 20 epochs for CIFAR-10 and 10 epochs for SVHN. The green and red regions represent the correct class and incorrect classes, respectively. The point at the center shows the datapoint, while the lines represent the different decision boundaries (note that the red regions can include different incorrect classes).

the classification regions are shown in the plane spanned by (r, v), where r is the normal to the decision boundary and v corresponds to a random direction.

In addition to inducing a larger distance between the data point and the decision boundary (hence resulting in a higher robustness), observe that the decision regions of fine-tuned networks are flatter and more regular. In particular, note that the curvature of the decision boundaries decreased after fine-tuning.

To quantify this phenomenon, we now compute the *curvature profile* of the loss function (with respect to the inputs) before and after adversarial fine-tuning. Formally, let ℓ denote the function that represents the loss of the network with respect to the inputs; e.g., in the case of cross-entropy, $\ell(x) = \mathrm{XEnt}(f_\theta(x), y)$, where y is the true label of image $x \in \mathbb{R}^d$, and $f_\theta(x)$ denotes the logits.[2] The curvature profile corresponds to the set of eigenvalues of the Hessian matrix

$$H = \left(\frac{\partial^2 \ell}{\partial x_i \partial x_j} \right) \in \mathbb{R}^{d \times d}$$

where $x_i, i = 1, \ldots, d$ denote the input pixels. We stress on the fact that the above

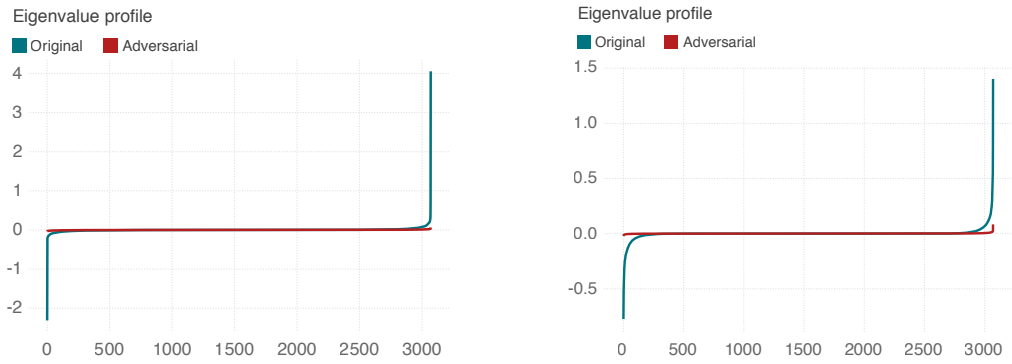
[2]We omit the label y from ℓ for simplicity, as the label can be understood from the context.

Hessian is with respect to the inputs, and not the weights of the network. To compute these eigenvalues in practice, we note that Hessian vector products are given by the following for any z;

$$Hz = \frac{\nabla \ell(x + hz) - \nabla \ell(x)}{h} \text{ for } h \to 0. \tag{8.1}$$

We then proceed to a finite difference approximation by choosing a finite h in Eq. (8.1). Besides being more efficient than generating the full Hessian matrix (which would be prohibitive for high-dimensional datasets), the finite difference approach has the benefit of measuring *larger-scale* variations of the gradient (where the scale is set using the parameter h) in the neighborhood of the datapoint, rather than an infinitesimal point-wise curvature. This is crucial in the setting of adversarial classification, where we analyze the loss function in a small neighbourhood of data points, rather than the asymptotic regime $h \to 0$ which might capture very local (and not relevant) variations of the function.[3]

Intuitively, small eigenvalues (in absolute value) of H indicate a small curvature of the graph of ℓ around x, hence implying that the classifier has a "locally linear" behaviour in the vicinity of x. In contrast, large eigenvalues (in absolute value) imply a high curvature of the loss function in the neighbourhood of image x. For example, in the case where the eigenvalues are exactly zero, the function becomes locally linear, hence leading to a flat decision surface.

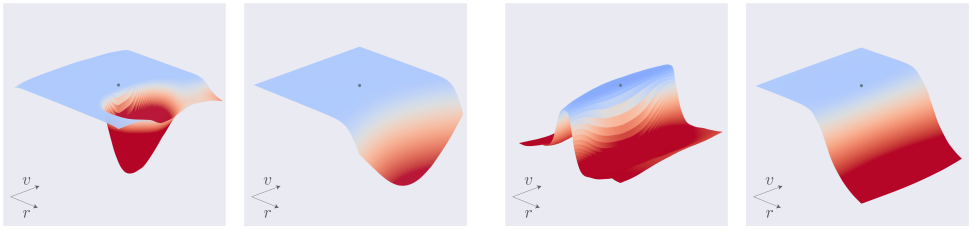


(a) CIFAR-10 (b) SVHN

Figure 8.2 – Curvature profiles, which correspond to sorted eigenvalues of the Hessian, of the original and the adversarially fine-tuned networks. Note that the number of eigenvalues is equal to $32 \times 32 \times 3 = 3072$, which corresponds to the number of input dimensions. The ResNet-18 architecture is used.

We compute the curvature profile at 100 random test samples, and show the average curvature in Fig. 8.2 for CIFAR-10 and SVHN datasets. Note that adversarial fine-tuning

[3] For example, using ReLU non-linearities result in a piecewise linear neural network as a function of the inputs. This implies that the Hessian computed at the logits is exactly 0. This result is however very local; using the finite-difference approximation, we focus on larger-scale neighbourhoods.



(a) Original (CIFAR-10) (b) Fine-tuned (CIFAR-10) (c) Original (SVHN) (d) Fine-tuned (SVHN)

Figure 8.3 – Illustration of the negative of the loss function; i.e., $-\ell(s)$ for points s belonging to a plane spanned by a normal direction r to the decision boundary, and random direction v. The original sample is illustrated with a blue dot. The light blue part of the surface corresponds to low loss (i.e., corresponding to the classification region of the sample), and the red part corresponds to the high loss (i.e., adversarial region).

	FGSM	ℓ_∞-DF	PGD(7)	PGD(20)
Original	38.0%	11.0%	0.5%	0.2%
Fine-tuned	61.0%	57.5%	57.2%	56.9%

Table 8.1 – Adversarial accuracies for original and fine-tuned network on CIFAR-10, where adversarial examples are computed with different attacks; FGSM [35], DF [70] and PGD [62]. Perturbations are constrained to have ℓ_∞ norm smaller than $\epsilon = 4$ (images have pixel values in $[0, 255]$).

has led to a strong decrease in the curvature of the loss in the neighborhood of data points. To further illustrate qualitatively this significant decrease in curvature due to adversarial training, Fig. 8.3 shows the loss surface before and after adversarial training along normal and random directions r and v. Observe that while the original network has large curvature in certain directions, the effect of adversarial training is to "regularize" the surface, resulting in a smoother, lower curvature (i.e., linear-like) loss.

We finally note that this effect of adversarial training on the loss surface has the following somewhat paradoxical implication: while adversarially trained models are *more robust* to adversarial perturbations (compared to original networks), they are also *easier to fool*, in the sense that simple attacks are as effective as complex ones. This is in stark contrast with original networks, where complex networks involving many gradient steps (e.g., PGD(20)) are much more effective than simple methods (e.g., FGSM). See Table 8.1. The comparatively small gap between the adversarial accuracies for different attacks on adversarially trained models is a direct consequence of the significant decrease of the curvature of the loss, thereby requiring a small number of gradient steps to find adversarial perturbations.

8.3 Analysis of the influence of curvature on robustness

While our results show that adversarial training leads to a decrease in the curvature of the loss, the relation between adversarial robustness and curvature of the loss remains unclear. To elucidate this relation, we consider a simple binary classification setting between class 1 and -1. Recall that $\ell(\cdot, 1)$ denotes the function that represents the loss of the network with respect to an input from class 1. For example, in the setting where the log-loss is considered, we have $\ell(x, 1) = -\log(p(x))$, where $p(x)$ denotes the output of softmax corresponding to class 1. In that setting, x is classified as class 1 iff $\ell(x, 1) \leq \log(2)$. For simplicity, we assume in our analysis that x belongs to class 1 without loss of generality, and hence omit the second argument in ℓ in the rest of this section. We assume that the function ℓ can be locally well approximated using a quadratic function; that is, for "sufficiently small" r, we can write:

$$\ell(x + r) \approx \ell(x) + \nabla\ell(x)^T r + \frac{1}{2} r^T H r,$$

where $\nabla\ell(x)$ and H denote respectively the gradient and Hessian of ℓ at x. Let x be a point classified as class 1; i.e., $\ell(x) \leq t$, where t denotes the loss threshold (e.g., $t = \log(2)$ for the log loss). For this datapoint x, we then define r^* to be the minimal perturbation in the ℓ_2 sense[4], which fools the classifier assuming the quadratic approximation holds; that is,

$$r^* := \arg\min_r \|r\| \text{ s.t. } \ell(x) + \nabla\ell(x)^T r + \frac{1}{2} r^T H r \geq t.$$

In the following result, we provide upper and lower bounds on the magnitude of r^* with respect to properties of the loss function at x.

Theorem 5. *Let x be such that $c := t - \ell(x) \geq 0$, and let $g = \nabla\ell(x)$. Assume that $\nu := \lambda_{\max}(H) \geq 0$, and let u be the eigenvector corresponding to ν. Then, we have*

$$\frac{\|g\|}{\nu}\left(\sqrt{1 + \frac{2\nu c}{\|g\|^2}} - 1\right) \leq \|r^*\| \tag{8.2}$$

$$\leq \frac{|g^T u|}{\nu}\left(\sqrt{1 + \frac{2\nu c}{(g^T u)^2}} - 1\right) \tag{8.3}$$

The above bounds can further be simplified to:

$$\frac{c}{\|g\|} - 2\nu\frac{c^2}{\|g\|^3} \leq \|r^*\| \leq \frac{c}{|g^T u|} \tag{8.4}$$

[4]We use the ℓ_2 norm for simplicity. Using the equivalence of norms in finite dimensional spaces, our result allows us to also bound the magnitude of ℓ_∞ adversarial perturbations.

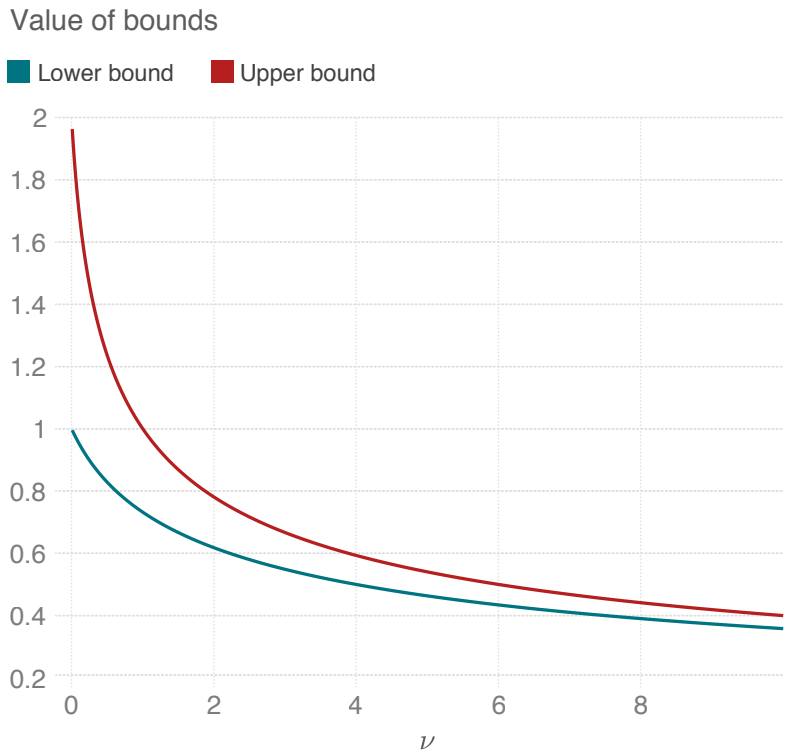


Figure 8.4 – Illustration of upper and lower bounds in Eq. (8.2) and (8.3) on the robustness with respect to curvature ν. We have set $\|\nabla\ell(x)\| = 1, c = 1, \nabla\ell(x)^T v = 0.5$ in this example.

Proof. **Lower bound.** Let $\alpha := \|r^*\|$. We note that α satisfies

$$-c + \|g\|\alpha + \frac{\nu}{2}\alpha^2 \geq -c + g^T r^* + \frac{1}{2}(r^*)^T H r^* \geq 0.$$

Solving the above second-order inequality, we get $\alpha \geq \frac{\|g\|}{\nu}\left(\sqrt{1 + \frac{2\nu c}{\|g\|^2}} - 1\right)$ or $\alpha \leq -\frac{\|g\|}{\nu}\left(\sqrt{1 + \frac{2\nu c}{\|g\|^2}} + 1\right)$. However, since $\alpha \geq 0$, the first inequality holds, which precisely corresponds to the lower bound.

Upper bound. Let $\alpha \geq 0$. Define $r := \alpha u$, and let us find the minimal $|\alpha|$ such that

$$-c + g^T r + \frac{1}{2} r^T H r = -c + \alpha g^T u + \frac{\alpha^2 \nu}{2} \geq 0.$$

We note that the above inequality holds for any $|\alpha| \geq |\alpha_{\min}|$, with

$$|\alpha_{\min}| = \frac{|g^T u|}{\nu}\left(\sqrt{1 + \frac{2\nu c}{(g^T u)^2}} - 1\right).$$

Hence, we have that $\|r^*\| \leq |\alpha_{\min}|$, which concludes the proof of the upper bound. The

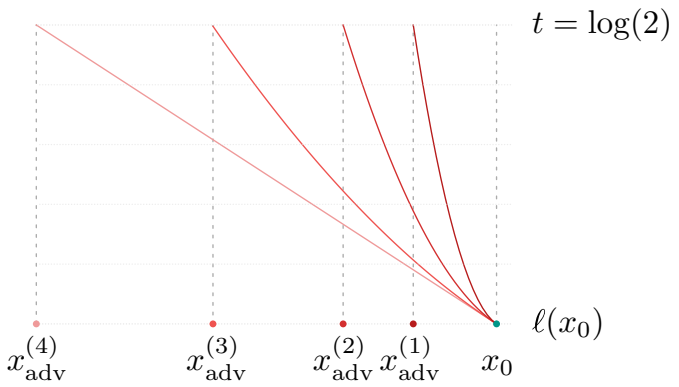


Figure 8.5 – Geometric illustration in 1d of the effect of curvature on the adversarial robustness. Different loss functions (with varying curvatures) are illustrated at the vicinity of data point x_0, and $x_{\text{adv}}^{(i)}$ indicate the points at which such losses exceed t (where t is the misclassification threshold). All curves have the same loss and gradient at x_0. Note that increasing curvature leads to smaller adversarial examples (i.e., smaller $|x_0 - x_{\text{adv}}^{(i)}|$).

simplified bounds are proven using the inequality $1 + \frac{x}{2} - \frac{x^2}{2} \leq \sqrt{1+x} \leq 1 + \frac{x}{2}$. □

Remark 1. ***Increasing*** **robustness with** ***decreasing*** **curvature.** Note that upper and lower bounds on the robustness in Eq. (8.2), (8.3) *decrease* with increasing curvature ν. To see this, Fig. 8.4 illustrates the dependence of the bounds on the curvature ν. In other words, under the second order approximation, this shows that *small curvature* (i.e., small eigenvalues of the Hessian) is beneficial to obtain classifiers with higher robustness (when the other parameters are kept fixed). This is in line with our observations from Section 8.2, where robust models are observed to have a smaller curvature than networks trained on original data. Fig. 8.5 provides intuition to the decreasing robustness with increasing curvature in a one-dimensional example.

Remark 2. Dependence on the gradient. In addition to the dependence on the curvature ν, note that the upper and lower bounds depend on the gradient $\nabla\ell(x)$. In particular, these bounds *decrease* with the norm $\|\nabla\ell(x)\|$ (for a fixed direction). Hence, under the second order approximation, this suggests that the robustness decreases with larger gradients. However, as previously noted in [92, 2], imposing small gradients might provide a false sense of robustness. That is, while having small gradients can make it hard for gradient-based methods to attack the network, the network can still be intrinsically vulnerable to small perturbations.

Remark 3. Bound tightness. Note that the upper and lower bounds match (and hence bounds are exact) when the gradient $\nabla\ell(x)$ is collinear to the largest eigenvector u. Interestingly, this condition seems to be approximately satisfied in practice, as the average normalized inner product $\frac{|\nabla\ell(x)^T u|}{\|\nabla\ell(x)\|_2}$ for CIFAR-10 is equal to 0.43 before adversarial fine-tuning, and 0.90 after fine-tuning (average over 1000 test points). This inner product is

significantly larger than the inner product between two typical vectors uniformly sampled from the sphere, which is approximately $\frac{1}{\sqrt{d}} \approx 0.02$. Hence, the gradient aligns well with the direction of largest curvature of the loss function in practice, which leads to approximately tight bounds.

8.4 Improving robustness through curvature regularization

While adversarial training leads to a regularity of the loss in the vicinity of data points, it remains unclear whether this regularity is the *main* effect of adversarial training, which confers robustness to the network, or it is rather a *byproduct* of a more sophisticated phenomenon. To answer this question, we follow here a *synthesis* approach, where we derive a regularizer which mimics the effect of adversarial training on the loss function – encouraging small curvatures.

Recall that H denotes the Hessian of the loss ℓ at datapoint x. We denote by $\lambda_1, \dots, \lambda_d$ the eigenvalues of H. Our aim is to penalize large eigenvalues of H; we therefore consider a regularizer $L_r = \sum_i p(\lambda_i)$, where p is a non-negative function, which we set to be $p(t) = t^2$ to encourage all eigenvalues to be small. For this choice of p, L_r corresponds to the Frobenius norm of the matrix H. We further note that

$$L_r = \sum_i p(\lambda_i) = \operatorname{trace}(p(H)) = \mathbb{E}(z^T p(H) z) = \mathbb{E}\|Hz\|^2, \tag{8.5}$$

where the expectation is taken over $z \sim \mathcal{N}(0, I_d)$. By using a finite difference approximation of the Hessian, we have $Hz \approx \frac{\nabla \ell(x+hz) - \nabla \ell(x)}{h}$, where h denotes the discretization step, and controls the scale on which we require the variation of the gradients to be small. Hence, L_r becomes

$$L_r = \frac{1}{h^2} \mathbb{E}\left\|\nabla \ell(x + hz) - \nabla \ell(x)\right\|^2. \tag{8.6}$$

The above regularizer involves computing an expectation over $z \sim \mathcal{N}(0, I_d)$, and penalizes large curvatures along all directions equally. Rather than approximating the above with an empirical expectation of $\|Hz\|^2$ over isotropic directions drawn from $\mathcal{N}(0, I_d)$, we instead *select* directions which are known to lead to high curvature (e.g., due to observation made in Chapter 7 and [43]), and minimize the curvature along such chosen directions. The latter approach is more efficient, as the computation of each matrix-vector product Hz involves one backward pass; focusing on high-curvature directions is therefore essential to minimize the overall curvature without having to go through each single direction in the input space. This selective approach is all the more adapted to the very sparse nature of curvature profiles we see in practice (see Fig. 8.2), where only a few eigenvalues are large. This provides further motivation for identifying large curvature directions and penalizing the curvature along such directions.

In Chapter 7, gradient directions have been identified as high curvature directions. In addition, empirical evidence reported in Section 8.3 (Remark 3) shows a large inner product between the eigenvector corresponding to maximum eigenvalue and the gradient direction; this provides further indication that the gradient is pointing in high curvature directions, and is therefore a suitable candidate for z. We set in practice $z = \frac{\text{sign}(\nabla \ell(x))}{\|\text{sign}(\nabla \ell(x))\|}$, and finally consider the regualizer [5]

$$L_r = \|\nabla \ell(x + hz) - \nabla \ell(x)\|^2, \tag{8.7}$$

where the $\frac{1}{h^2}$ is absorbed by the regularization parameter. Our fine-tuning procedure then corresponds to minimizing the regularized loss function $\ell + \gamma L_r$ with respect to the weight parameters, where γ controls the weight of the regularization relative to the loss term.

We stress that the proposed regularization approach significantly departs from adversarial training. In particular, while adversarial training consists in minimizing *the loss on perturbed points* (which involves solving an optimization problem), our approach here consists in imposing regularity *of the gradients* on a sufficiently small scale (i.e., determined by h). Previous works [62] have shown that adversarial training using a weak attack (such as FGSM [35], which involves a single gradient step) does *not* improve the robustness. We show that our approach, which rather imposes gradient regularity (i.e., small curvature) along such directions, does lead to a significant improvement in the robustness of the network.

We use two pre-trained networks, ResNet-18 [38] and WResNet-28x10 [99], on the CIFAR-10 and SVHN datasets, where the pixel values are in $[0, 255]$. For the optimization of the regularized objective, we use the Adam optimizer with a decreasing learning rate between $[10^{-4}, 10^{-6}]$ for a duration of 20 epochs starting from a pre-trained network. We linearly increase the value of h from 0 to 1.5 during the first 5 epochs, and from there on, we use a fixed value of $h = 1.5$. For γ, we set it to 4 and 8 for ResNet-18 and WResNet-28 respectively.

8.5 Experimental evaluations of CURE

We evaluate the regularized networks with a strong PGD attack of 20 iterations, as it has been shown to outperform other adversarial attack algorithms [62]. The adversarial accuracies of the regularized networks are reported in Table 8.2 for CIFAR-10, and in Table 8.3 for SVHN. Moreover, the adversarial accuracy as a function of the perturbation magnitude ϵ is reported in Fig. 8.6.

[5] The choice of $z \propto \nabla \ell(x)$ leads to almost identical results. We have chosen to set $z \propto \text{sign}(\nabla \ell(x))$, as we are testing the robustness of the classifier to ℓ_∞ perturbations. Hence, setting z be the sign of the gradient is more relevant, as it constrains the z direction to belong to the hypercube of interest.

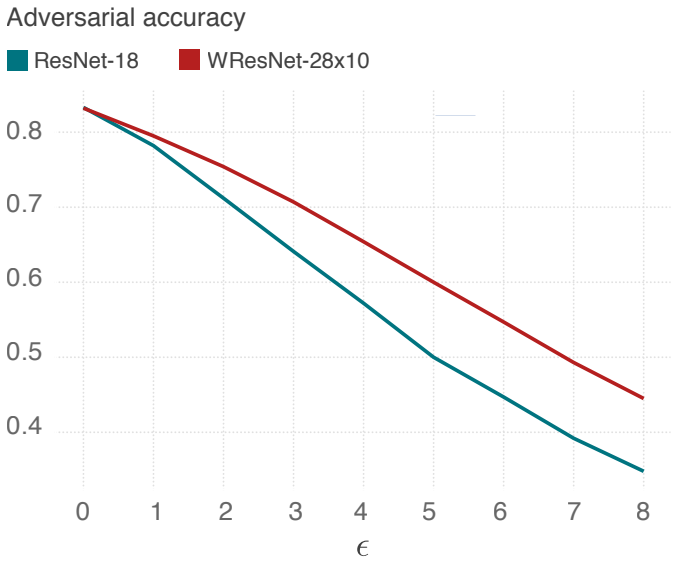


Figure 8.6 – Adversarial accuracy versus perturbation magnitude ϵ computed using PGD(20), for ResNet-18 and WResNet-28x10 trained with CURE on CIFAR-10. See [62] for the curve corresponding to adversarial training. Curve generated for 2000 random test points.

Table 8.2 – Adversarial and clean accuracy for CIFAR-10 for original, regularized and adversarially trained models. Performance is reported for ResNet and WideResNet models, and the perturbations are computed using PGD(20). Perturbations are constrained to have ℓ_∞ norm less than $\epsilon = 8$ (where pixel values are in $[0, 255]$).

	ResNet-18		WideResNet-28×10	
	Clean	Adversarial	Clean	Adversarial
Normal training	94.9%	0.0%	94.6%	0.0%
CURE	81.2%	36.3%	83.1%	41.4%
Adversarial training [62]	79.4%	43.7%	87.3%	45.8%

Table 8.3 – Adversarial and clean accuracy for SVHN for original, regularized and adversarially trained models. Performance is reported for a ResNet-18 model, and the perturbations are computed using PGD(10) with $\epsilon = 12$.

	ResNet-18	
	Clean	Adversarial
Normal training	96.3%	0.9%
CURE	91.1%	28.4%
Adversarial training (reported in [9])	93%	33%

Observe that, while networks trained on the original dataset are not robust to perturbations as expected, performing 20 epochs of fine-tuning with the proposed regularizer leads to a significant boost in adversarial performance. In particular, the performance with the proposed regularizer is comparable to that of adversarial training reported in [62]. This result hence shows the importance of the curvature decrease phenomenon described in this chapter in explaining the success of adversarial training.

In addition to verifying our claim that small curvature confers robustness to the network (and that it is the underlying effect in adversarial training), we note that the proposed regularizer has practical value, as it is efficient to compute and can therefore be used as an alternative to adversarial training. In fact, the proposed regularizer requires 2 backward passes to compute, and is used in fine-tuning for 20 epochs. In contrast, one needs to run adversarial training against a *strong* adversary in order to reach good robustness [62], and start the adversarial training procedure from scratch. We note that strong adversaries generally require around 10 backward passes, making the proposed regularization scheme a more efficient alternative. We note however that the obtained results are slightly worse than adversarial training; we hypothesize that this might be either due to higher order effects in adversarial training not captured with our second order analysis or potentially due to a sub-optimal choice of hyper-parameters γ and h.

8.5.1 Stronger attacks and verifying the absence of gradient masking

To provide further evidence on the robustness of the network fine-tuned with CURE, we attempt to find perturbations for the network with more complex attack algorithms. For the WideResNet-28x10, we obtain an addversarial accuracy of 41.1% on the test set when using PGD(40) and PGD(100). This is only slightly worse than the result reported in Table 8.2 with PGD(20). This shows that increasing the complexity of the attack does not lead to a significant decrease in the adversarial accuracy. Moreover, we evaluate the model against a gradient-free optimization method (SPSA), similar to the methodology used in [92], and obtained an adversarial accuracy of 44.5%. We compare moreover in Fig. 8.7 the *adversarial loss* (which represents the difference between the logit scores of the true and adversarial class) computed using SPSA and PGD for a batch of test data points. Observe that both methods lead to comparable adversarial loss (except on a few data points), hence further justifying that CURE truly improves the robustness, as opposed to masking or obfuscating gradients. Hence, just like adversarial training which was shown empirically to lead to networks that are robust to all tested attacks in [92, 2], our experiments show that the regularized network has similar robustness properties.

8.5.2 Curvature and robustness

We now analyze the network obtained using CURE fine-tuning, and show that the obtained network has similar geometric properties to the adversarially trained one. Fig. 8.8 shows

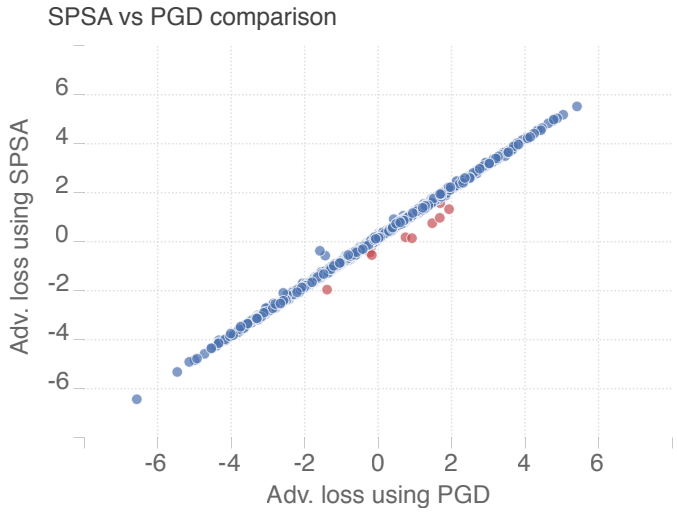
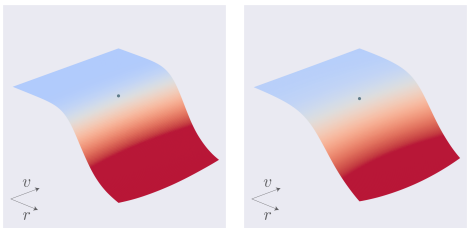


Figure 8.7 – Analysis of gradient masking in a network trained with CURE. Adversarial loss computed with SPSA (y-axis) vs. adversarial loss with PGD(100) (x-axis) on a batch of 1000 datapoints. Adversarial loss corresponds to the difference of logits on true and adversarial class. Each point in the scatter plot corresponds to a single test sample. Negative loss indicates that the data point is misclassified. Points close to the line $y = x$ indicate that both attacks identified similar adversarial perturbations. Points below the line, shown in red, indicate points for which SPSA identified stronger adversarial perturbation than PGD. Note that overall, SPSA and PGD identified similarly perturbations.



(a) ResNet-18 (b) WideResNet-28

Figure 8.8 – Similar plot to Fig. 8.3, but where the loss surfaces of the network obtained with CURE are shown.

the loss surface in a plane spanned by (r, v), where r and v denote respectively a normal to the decision boundary and a random direction. Note that the loss surface obtained with CURE is qualitatively very similar to the one obtained with adversarial training (Fig. 8.3), whereby the loss has a more linear behavior in the vicinity of the data point. Quantitatively, Fig. 8.9 compares the curvature profiles for the networks trained with CURE and adversarial fine-tuning. Observe that both profiles are very similar.

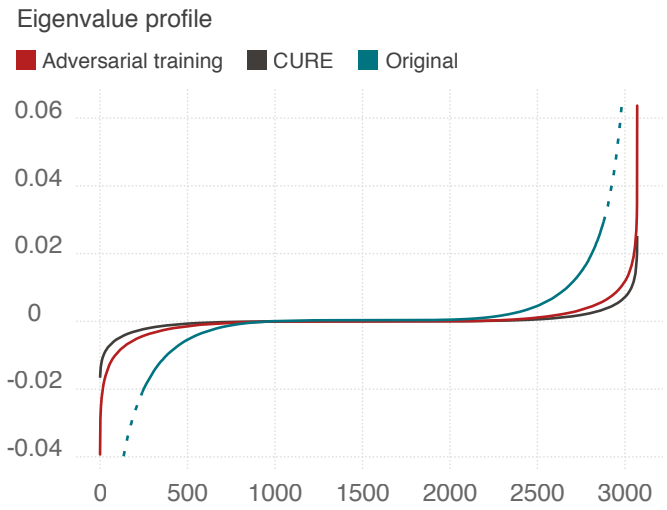


Figure 8.9 – Curvature profile for a network fine-tuned using adversarial training and CURE. The ResNet-18 architecture on CIFAR-10 is used. For comparison, we also report the profile for the original network (same as Fig. 8.2), where we clipped the values to fit in the y range.

We also report the evolution of the adversarial accuracy and curvature quantities in Fig. 8.10 during fine-tuning with CURE. Note that throughout the fine-tuning process, the curvature decreases while the adversarial accuracy increases, which further shows the link between robustness and curvature. Note also that, while we explicitly regularized for $\|Hz\|$ (where z is a fixed direction for each data point) as a proxy for $\|H\|_F$, the network does show that the intended target $\|H\|_F$ decreases in the course of training, hence further suggesting that $\|Hz\|$ acts as an efficient proxy of the global curvature.

8.5.3 Qualitative evaluation of adversarial perturbations

We finally illustrate some adversarial examples in Fig. 8.11 for networks trained on SVHN. Observe that the network trained with CURE exhibits visually meaningful adversarial examples, as perturbed images do resemble images from the adversary class. A similar observation for adversarially trained models has been made in [91].

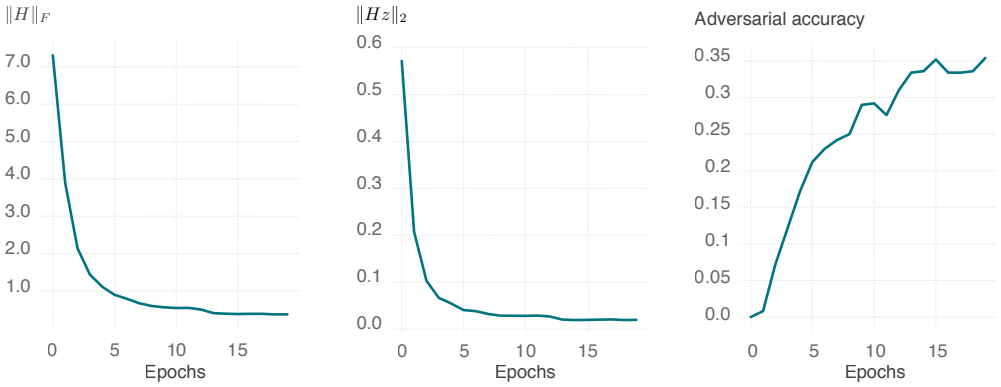


Figure 8.10 – Evolution throughout the course of CURE fine-tuning for a ResNet-18 on CIFAR-10. The curves are averaged over 1000 datapoints. **Left:** estimate of Frobenius norm , **Middle:** $\|Hz\|$, where $z = \text{sign}(\nabla \ell(x))/\|\text{sign}(\nabla \ell(x))\|_2$ and **Right:** adversarial accuracy computed using PGD(20). The Frobenius norm is estimated with $\|H\|_F^2 = \mathbb{E}_{z\sim\mathcal{N}(0,I)}\|Hz\|^2$, where the expectation is approximated with an empirical expectation over 100 samples $z_i \sim \mathcal{N}(0, I)$.

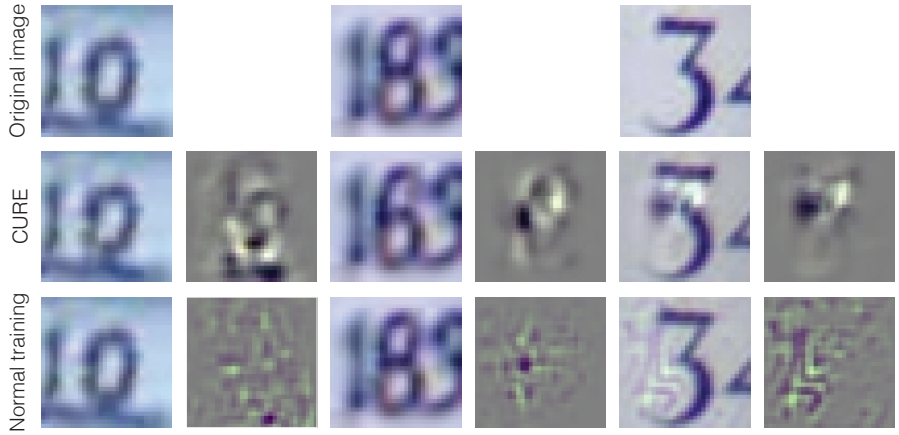


Figure 8.11 – Visualizations of perturbed images and perturbations on SVHN for the ResNet-18 classifier. Note that the perturbed images corresponding to CURE are more visually meaningful.

8.6 Conclusion

Guided by the analysis of the local geometry of deep networks introduced in the previous chapters, we have provided empirical and theoretical evidence showing the existence of a strong correlation between small curvature and robustness. To validate our analysis, we proposed a new regularizer (CURE), which directly encourages small curvatures (in other words, promotes local linearity). This regularizer is shown to significantly improve the robustness of deep networks and even achieve performance that is comparable to adversarial training. In light of prior works attributing the vulnerability of classifiers to the "linearity of deep networks", this result is somewhat surprising, as it shows that

one needs to decrease the curvature (and not increase it) to improve the robustness. In addition to validating the importance of controlling the curvature for improving the robustness, the proposed regularizer also provides an efficient alternative to adversarial training.

9 Conclusion

9.1 Summary

In this thesis, we provided new powerful algorithmic tools to measure the robustness properties of deep image classifiers. We have further used these tools to identify important geometric properties of the decision boundary of deep classifiers and to explain their vulnerability to adversarial perturbations. Our unique geometric analysis was also employed to highlight some of the key elements of the robustness of deep networks. Our analysis was further shown to be effective in building more robust image classifiers.

We first studied methods to quantify the robustness of classifiers to additive adversarial perturbations. Although the computation of such perturbations for state-of-the-art deep image classifiers requires solving a high-dimensional and non-convex optimization problem, we showed through extensive experimental evaluations that this problem can be addressed in a fast yet efficient way using our DeepFool algorithm. It efficiently computes the minimal ℓ_2-norm adversarial perturbations for deep neural networks, and can thus reliably quantify the robustness of such image classifiers. Furthermore, DeepFool provides a general framework to design scalable algorithms to study the robustness properties of classifiers to structured adversarial perturbations. Next, we showed that the vulnerability of deep networks to adversarial manipulations is not limited to image-dependent perturbations. Indeed, there exist universal perturbations, which are very small image-agnostic perturbation vectors that cause natural images to be misclassified with high probability. We proposed a simple algorithm for computing these perturbations, and showed that state-of-the-art deep neural networks are highly vulnerable to such perturbations, even if those are quasi-imperceptible to the human eye. The existence of universal perturbations implies serious concerns regarding deploying image classifiers in hostile environments.

Then, we moved to explain the vulnerability of deep image classifiers to adversarial and universal perturbations through novel empirical and theoretical analyses of the geometry

of these classifiers. Despite the fact that natural image classifiers are acting on high-dimensional data, our analysis, facilitated by our powerful algorithmic tools, could quantify some of the key geometric aspects contributing to the high vulnerability of these classifiers to adversarial and universal perturbations. Specifically, we established theoretical bounds on the robustness of classifiers to adversarial perturbations which depend on the curvature of the classifier's decision boundary. We particularly showed that having "almost flat" decision boundaries can partially explain the adversarial instability of classifiers. We further showed that the high curvature of the decision boundary of classifiers along a few shared directions well explains the existence of universal perturbations for deep image classifiers. Such understanding can potentially help us design geometric regularizers to improve the robustness of classifiers to universal perturbations.

We then presented an application of our algorithmic tools to study the topological properties of the decision regions of image classifiers. In particular, we proposed a simple yet effective method to study the path-connectedness of classification regions created by deep neural networks. Through a systematic empirical study, we showed that state-of-the-art deep networks surprisingly learn connected classification regions. Such observation questions our understanding of the expressive power of deep networks.

We finally demonstrated how to exploit curvature information to mitigate the adversarial vulnerability of deep neural networks. We specifically showed how to leverage a fundamental asymmetry property in the curvature of the decision boundary of deep networks in order to detect images perturbed with minimal adversarial perturbations. We showed the effectiveness of this purely geometric approach for detecting small adversarial perturbations in images, and for recovering the labels of perturbed images.

Moreover, we provided a novel geometric characterization of the effect of adversarial training on the robustness of state-of-the-art classifiers. We showed in particular that adversarial training leads to a significant decrease in the curvature of the loss surface with respect to inputs, leading to a drastically more "linear" behaviour of the network. To further show the importance of reduced curvature for improving the robustness, we proposed an efficient regularization, CURE, that directly minimises curvature of the loss surface, and leads to adversarial robustness that is on par with adversarial training. Our method seems to be a promising approach for building inherently more robust classifiers.

9.2 Future directions

To better evaluate the risk associated to the deployment of state-of-the-art classifiers in real-world environments, it is crucial to assess their robustness properties beyond mere additive perturbations. While our proposed algorithms efficiently compute additive adversarial perturbations for state-of-the-art deep neural networks, designing a general framework to efficiently assess the robustness to more realistic adversarial manipulations

of data remains challenging. However, it would be possible to use the geometric properties of deep networks as prior information to design efficient evaluation methods for non-additive and black-box adversarial manipulations. The robustness to such adversarial manipulations are highly desired in safety-critical applications of image classifiers in real-world settings such as autonomous driving.

While we have identified some of the important geometric properties of the decision boundary of deep networks, it is equally important to understand how different factors involved in the process of training these networks – such as their architecture, optimizer, loss function, etc. – affect the geometry of the decision boundary. Some of the observation related to the geometric properties of deep neural networks, such as path-connectedness and flatness of the decision boundary, are intriguingly similar to the observations made for the optimization landscape of these networks (see e.g., [34, 15]). Though some recent works (e.g., [98]) have established links between the robustness properties and the Hessian (curvature information) of the optimization landscape of deep networks, the full extent of the relation between the geometric properties of the input space and the optimization landscape of these classifiers is to be explored. Furthermore, the surprising resemblance of the geometric properties of deep neural networks to linear classifiers, such as flatness of the decision boundary and connectedness of the decision regions, begs for a deeper understanding of the expressive power of such non-linear functions in terms of their geometry, and the fundamental aspects in which they are different from linear classifiers.

One of the ultimate goals of the analysis of the adversarial robustness is to build more robust classifiers and eventually more robust autonomous agents. While *adversarial training* has been shown to be one of the most successful methods to improve the adversarial robustness of deep networks, we lack a thorough understanding of the key factors behind its surprising effectiveness. Though our geometric analysis of adversarial training has led to an effective regularizer, more computationally efficient methods are needed in order to develop practical algorithms that improve the adversarial robustness in very high-dimensional classification tasks (e.g., ImageNet). Furthermore, on the theoretical side, despite the recent efforts to explain the potential trade-off between the adversarial robustness and the generalization performance of deep networks (e.g., [43, 91]), the question whether adversarial training achieves the best trade-off has remained unsolved.

A Appendix of Chapter 5

A.1 Proof of Theorem 1 (affine classifiers)

Lemma 1 ([20]). *Let Y be a point chosen uniformly at random from the surface of the d-dimensional sphere $\mathbb{S}^{d-1}$. Let the vector Z be the projection of Y onto its first m coordinates, with $m < d$. Then,*

1. *If $\beta < 1$, then*

$$\mathbb{P}\left(\|Z\|_2^2 \leq \frac{\beta m}{d}\right) \leq \beta^{m/2}\left(1 + \frac{(1-\beta)m}{(d-m)}\right)^{(d-m)/2} \leq \exp\left(\frac{m}{2}(1-\beta+\ln\beta)\right). \tag{A.1}$$

2. *If $\beta > 1$, then*

$$\mathbb{P}\left(\|Z\|_2^2 \geq \frac{\beta m}{d}\right) \leq \beta^{m/2}\left(1 + \frac{(1-\beta)m}{(d-m)}\right)^{(d-m)/2} \leq \exp\left(\frac{m}{2}(1-\beta+\ln\beta)\right). \tag{A.2}$$

Lemma 2. *Let $\boldsymbol{v}$ be a random vector uniformly drawn from the unit sphere $\mathbb{S}^{d-1}$, and $\mathbf{P}_m$ be the projection matrix onto the first m coordinates. Then,*

$$\mathbb{P}\left(\beta_1(\delta,m)\frac{m}{d} \leq \|\mathbf{P}_m\boldsymbol{v}\|_2^2 \leq \beta_2(\delta,m)\frac{m}{d}\right) \geq 1-2\delta, \tag{A.3}$$

with $\beta_1(\delta,m) = \max((1/e)\delta^{2/m}, 1-\sqrt{2(1-\delta^{2/m})}$, and $\beta_2(\delta,m) = 1+2\sqrt{\frac{\ln(1/\delta)}{m}}+\frac{2\ln(1/\delta)}{m}$.

Proof. Note first that the upper bound of Lemma 1 can be bounded as follows:

$$\beta^{m/2}\left(1 + \frac{(1-\beta)m}{d-m}\right)^{(d-m)/2} \leq \beta^{m/2}\exp\left(\frac{(1-\beta)m}{2}\right), \tag{A.4}$$

using $1 + x \leq \exp(x)$. We find β such that $\beta^{m/2} \exp\left(\frac{(1-\beta)m}{2}\right) \leq \delta$, or equivalently, $\beta \exp(1-\beta) \leq \delta^{2/m}$. It is easy to see that when $\beta = \frac{1}{e}\delta^{2/m}$, the inequality holds. Note however that $\frac{1}{e}\delta^{2/m}$ does not converge to 1 as $m \to \infty$. We therefore need to derive a tighter bound for this regime. Using the inequality $\beta \exp(1-\beta) \leq 1 - \frac{1}{2}(1-\beta)^2$ for $0 \leq \beta \leq 1$, it follows that the inequality $\beta \exp(1-\beta) \leq \delta^{2/m}$ holds for $\beta = 1 - \sqrt{2(1-\delta^{2/m})}$. In this case, we have $1 - \sqrt{2(1-\delta^{2/m})} \to 1$, as $m \to \infty$. We take our lower bound to be the max of both derived bounds (the latter is more appropriate for large m, whereas the former is tighter for small m).

For β_2, note that the requirement $\beta \exp(1-\beta) \leq \delta^{2/m}$ is equivalent to $-\ln(\beta) + (\beta - 1) \geq \frac{2}{m}\ln(1/\delta)$. By setting $\beta = \beta_2(\delta, m)$, this condition is equivalent to $2\sqrt{\frac{\ln(1/\delta)}{m}} - \ln(\beta_2(\delta, m)) \geq 0$, or equivalently, $2z - \ln(1 + 2z + 2z^2) \geq 0$, with $z = \sqrt{\frac{\ln(1/\delta)}{m}}$. The function $z \mapsto 2z - \ln(1 + 2z + 2z^2) \geq 0$ is positive on $\mathbb{R}^+$. Hence, $\beta_2(\delta, m)$ satisfies $\beta \exp(1-\beta) \leq \delta^{2/m}$, which concludes the proof. □

We now prove our main theorem that we recall as follows:

Theorem 1. *Let $\mathcal{S}$ be a random m-dimensional subspace of $\mathbb{R}^d$. The following inequalities hold between the norms of semi-random perturbation $\boldsymbol{r}^*_{\mathcal{S}}$ and the worst-case perturbation $\boldsymbol{r}^*$. Let $\zeta_1(m, \delta) = \frac{1}{\beta_2(m,\delta)}$, and $\zeta_2(m, \delta) = \frac{1}{\beta_1(m,\delta)}$.*

$$\zeta_1(m, \delta)\frac{d}{m}\|\boldsymbol{r}^*\|_2^2 \leq \|\boldsymbol{r}^*_{\mathcal{S}}\|_2^2 \leq \zeta_2(m, \delta)\frac{d}{m}\|\boldsymbol{r}^*\|_2^2, \tag{A.5}$$

with probability exceeding $1 - 2(L+1)\delta$.

Proof. For the linear case, $\boldsymbol{r}^*$ and $\boldsymbol{r}^*_{\mathcal{S}}$ can be computed in closed form. We recall that, for any subspace $\mathcal{S}$, we have

$$\boldsymbol{r}^k_{\mathcal{S}} = \frac{\left|f_k(\boldsymbol{x}_0) - f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_0)\right|}{\|\mathbf{P}_{\mathcal{S}}\boldsymbol{w}_k - \mathbf{P}_{\mathcal{S}}\boldsymbol{w}_{\hat{k}(\boldsymbol{x}_0)}\|_2^2}(\mathbf{P}_{\mathcal{S}}\boldsymbol{w}_k - \mathbf{P}_{\mathcal{S}}\boldsymbol{w}_{\hat{k}(\boldsymbol{x}_0)}), \tag{A.6}$$

where $\boldsymbol{r}^k_{\mathcal{S}}$ was defined in Eq. (5.8) in the main paper. In particular, when $\mathcal{S} = \mathbb{R}^d$, we have

$$\boldsymbol{r}^k = \frac{\left|f_k(\boldsymbol{x}_0) - f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_0)\right|}{\|\boldsymbol{w}_k - \boldsymbol{w}_{\hat{k}(\boldsymbol{x}_0)}\|_2^2}(\boldsymbol{w}_k - \boldsymbol{w}_{\hat{k}(\boldsymbol{x}_0)}). \tag{A.7}$$

Let $k \neq \hat{k}(\boldsymbol{x}_0)$. Define, for the sake of readability

$$f^k = \left|f_k(\boldsymbol{x}_0) - f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_0)\right|,$$
$$\boldsymbol{z}^k = \boldsymbol{w}_k - \boldsymbol{w}_{\hat{k}(\boldsymbol{x}_0)}.$$

Note that

$$\frac{\|\boldsymbol{r}^k\|_2^2}{\|\boldsymbol{r}_{\mathcal{S}}^k\|_2^2} = \frac{\|\mathbf{P}_{\mathcal{S}}\boldsymbol{z}^k\|_2^2}{\|\boldsymbol{z}^k\|_2^2}. \tag{A.8}$$

The projection of a fixed vector in $\mathbb{S}^{d-1}$ onto a random m dimensional subspace is equivalent (up to a unitary transformation $\mathbf{U}$) to the projection of a random vector uniformly sampled from $\mathbb{S}^{d-1}$ into a fixed subspace. Let $\mathbf{P}_m$ be the projection onto the first m coordinates. We have

$$\|\mathbf{P}_{\mathcal{S}}\boldsymbol{z}^k\|_2^2 = \|\mathbf{U}^T\mathbf{P}_m\mathbf{U}\boldsymbol{z}^k\|_2^2 = \|\mathbf{P}_m\mathbf{U}\boldsymbol{z}^k\|_2, \tag{A.9}$$

Hence, we have

$$\frac{\|\mathbf{P}_{\mathcal{S}}\boldsymbol{z}^k\|_2^2}{\|\boldsymbol{z}^k\|_2^2} = \|\mathbf{P}_m\boldsymbol{y}\|_2^2, \tag{A.10}$$

where $\boldsymbol{y}$ is a random vector distributed uniformly in the unit sphere $\mathbb{S}^{d-1}$. We apply Lemma 2, and obtain

$$\mathbb{P}\left(\beta_1(m,\delta)\frac{m}{d} \leq \|\mathbf{P}_m\boldsymbol{y}\|_2^2 \leq \beta_2(m,\delta)\frac{m}{d}\right) \geq 1-2\delta. \tag{A.11}$$

Hence,

$$\mathbb{P}\left\{\frac{1}{\beta_2(m,\delta)}\frac{d}{m} \leq \frac{\|\boldsymbol{r}_{\mathcal{S}}^k\|_2^2}{\|\boldsymbol{r}^k\|_2^2} \leq \frac{1}{\beta_1(m,\delta)}\frac{d}{m}\right\} \geq 1-2\delta. \tag{A.12}$$

Using the multi-class extension in Lemma 3, we conclude that

$$\mathbb{P}\left\{\zeta_1(m,\delta)\frac{d}{m} \leq \frac{\|\boldsymbol{r}_{\mathcal{S}}^*\|_2^2}{\|\boldsymbol{r}^*\|_2^2} \leq \zeta_2(m,\delta)\frac{d}{m}\right\} \geq 1-2(L+1)\delta. \tag{A.13}$$

□

Lemma 3 (Binary case to multiclass)**.** *Assume that, for all* $k \in \{1,\dots,L]\}\backslash\{\hat{k}(\boldsymbol{x}_0)\}$

$$\mathbb{P}\left(l \leq \frac{\|\boldsymbol{r}_{\mathcal{S}}^k\|_2}{\|\boldsymbol{r}^k\|_2} \leq u\right) \geq 1-\delta. \tag{A.14}$$

Then, we have

$$\mathbb{P}\left(l \leq \frac{\|\boldsymbol{r}_{\mathcal{S}}^*\|_2}{\|\boldsymbol{r}^*\|_2} \leq u\right) \geq 1-(L+1)\delta. \tag{A.15}$$

Proof. Let $p := \arg\min_i \|\boldsymbol{r}^i\|_2$. Note that we have $\mathbb{P}\left(\frac{\|\boldsymbol{r}_{\mathcal{S}}^*\|_2}{\|\boldsymbol{r}^*\|_2} \geq u\right) \leq \mathbb{P}\left(\frac{\|\boldsymbol{r}_{\mathcal{S}}^p\|_2}{\|\boldsymbol{r}^p\|_2} \geq u\right) \leq \delta$.

Moreover, we use a union bound to bound the the other bad event probability:

$$\mathbb{P}\left(\frac{\|\boldsymbol{r}^*_{\mathcal{S}}\|_2}{\|\boldsymbol{r}^*\|_2} \leq l\right) \leq \mathbb{P}\left(\bigcup_k \left\{\frac{\|\boldsymbol{r}^k_{\mathcal{S}}\|_2}{\|\boldsymbol{r}^k\|_2} \leq l\right\}\right) \leq L\delta, \tag{A.16}$$

$$\tag{A.17}$$

We conclude by using the fact that

$$\mathbb{P}\left(l \leq \frac{\|\boldsymbol{r}^*_{\mathcal{S}}\|_2}{\|\boldsymbol{r}^*\|_2} \leq u\right) = 1 - \mathbb{P}\left(\frac{\|\boldsymbol{r}^*_{\mathcal{S}}\|_2}{\|\boldsymbol{r}^*\|_2} \leq l\right) - \mathbb{P}\left(\frac{\|\boldsymbol{r}^*_{\mathcal{S}}\|_2}{\|\boldsymbol{r}^*\|_2} \geq u\right). \tag{A.18}$$

□

A.2 Proof of Theorem 2 and Corollary 1 (nonlinear classifiers)

First, we present an important geometric lemma and then use it to bound $\|\boldsymbol{r}^*_{\mathcal{S}}\|_2$. For the sake of the general readability of the section, some auxiliary results are given in Section A.2.

In the following result, we show that, when the curvature of a planar curve is constant and sufficiently small, the distance between a point $\boldsymbol{x}$ and the curve at a specific direction θ is well approximated by the distance between $\boldsymbol{x}$ and a straight line (see Fig. A.1 for an illustration).

Lemma 4. *Let γ be a planar curve of constant curvature κ. We denote by r the distance between a point $\boldsymbol{x}$ and the curve γ. Denote moreover by $\mathcal{T}$ the tangent to γ at the closest point to $\boldsymbol{x}$ (see Fig. A.1). Let θ be the angle between $\boldsymbol{u}$ and $\boldsymbol{v}$ as depicted in Fig. A.1. We assume that $r\kappa < 1$. We have*

$$-C_1 r\kappa \tan^2(\theta) \leq \frac{\|\boldsymbol{x}_\gamma - \boldsymbol{x}\|_2}{\|\boldsymbol{u}\|_2} - 1 \tag{A.19}$$

Moreover, if

$$\tan^2(\theta) \leq \frac{0.2}{r\kappa},$$

then, the following upper bound holds

$$\frac{\|\boldsymbol{x}_\gamma - \boldsymbol{x}\|_2}{\|\boldsymbol{u}\|_2} - 1 \leq C_2 r\kappa \tan^2(\theta). \tag{A.20}$$

We can set $C_1 = 0.625$ and $C_2 = 2.25$.

Proof of upper bound. We consider two distinct cases for the curve γ. In the case where

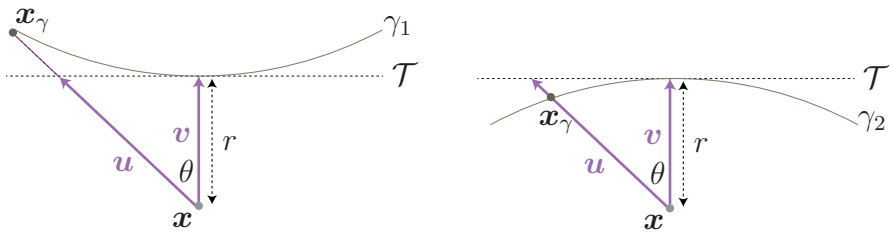


Figure A.1 – Bounding $\|\boldsymbol{x}_\gamma - \boldsymbol{x}\|_2$ in terms of κ.

γ is concave-shaped (Fig. A.1, right figure), we have

$$\frac{\|\boldsymbol{x}_\gamma - \boldsymbol{x}\|_2}{\|\boldsymbol{u}\|_2} \leq 1,$$

and the upper bound in Eq. (A.20) directly holds. We therefore focus on the case where γ is convex-shaped as illustrated in the left figure of Fig. A.1. Define $R := 1/\kappa$, one can write using simple geometric inspection

$$R^2 = \sin(\theta) r'^2 + (R + r - r'\cos(\theta))^2, \tag{A.21}$$

where $r' = \|\boldsymbol{x}_\gamma - \boldsymbol{x}\|_2$. The discriminant of the second order equation (with variable r') is equal to

$$\Delta = 4\left((R+r)^2\cos^2(\theta) - (2rR + r^2)\right).$$

We have $\Delta \geq 0$ as θ satisfies the two assumptions $\tan^2(\theta) \leq 0.2R/r$ and $r/R < 1$. The smallest solution of this second order equation is given as follows

$$r' = (R+r)\cos(\theta) - \sqrt{(R+r)^2\cos^2(\theta) - 2Rr - r^2}. \tag{A.22}$$

Using some simple algebraic manipulations, we obtain

$$r' = \frac{r}{\cos(\theta)}\left(\left(\frac{R}{r}+1\right)\cos^2(\theta) - \frac{R}{r}\cos^2(\theta)\sqrt{1 - \tan^2(\theta)\frac{2Rr + r^2}{R^2}}\right). \tag{A.23}$$

Using the inequality in Lemma 7 together with the two assumptions, we get

$$\begin{aligned} r' \leq \frac{r}{\cos(\theta)}\Bigg(&\cos^2(\theta) + \frac{R}{r}\cos^2(\theta)\tan^2(\theta)\left(\frac{2Rr + r^2}{2R^2}\right) \\ &+ \frac{R}{r}\cos^2(\theta)\tan^4(\theta)\left(\frac{2Rr + r^2}{2R^2}\right)^2\Bigg). \end{aligned} \tag{A.24}$$

With simple trigonometric identities, the above expression can be simplified to

$$r' \leq \frac{r}{\cos(\theta)} \left(1 + \frac{r}{R} \left(\frac{\sin^2(\theta)}{2} + \frac{\sin^4(\theta)}{\cos^2(\theta)} \left(1 + \frac{r}{2R}\right)^2 \right) \right). \tag{A.25}$$

We expand this quantity, and obtain

$$r' \leq \frac{r}{\cos(\theta)} \left(1 + \left(\frac{\sin^2(\theta)}{2} + \frac{\sin^4(\theta)}{\cos^2(\theta)} \right) \frac{r}{R} + \frac{\sin^4(\theta)}{\cos^2(\theta)} \frac{r^2}{R^2} + \frac{\sin^4(\theta)}{4\cos^2(\theta)} \frac{r^3}{R^3} \right). \tag{A.26}$$

Since $\sin^2(\theta)\tan^2(\theta) = \tan^2(\theta) - \sin^2(\theta)$, we have

$$r' \leq \frac{r}{\cos(\theta)} \left(1 + \tan^2(\theta) \left(\frac{r}{R} + \frac{r^2}{R^2} + \frac{r^3}{4R^3} \right) \right). \tag{A.27}$$

According to the assumptions $r/R < 1$, therefore

$$r' \leq \frac{r}{\cos(\theta)} \left(1 + 2.25 \tan^2(\theta) \frac{r}{R}\right). \tag{A.28}$$

Since $r/\cos(\theta) = \|\boldsymbol{u}\|_2$, one can finally conclude on the upper bound

$$\frac{\|\boldsymbol{x}_\gamma - \boldsymbol{x}\|_2}{\|\boldsymbol{u}\|_2} - 1 \leq 2.25 r\kappa \tan^2(\theta). \tag{A.29}$$

□

Proof of lower bound. When the curve is convex shaped (Fig. A.1 left), we have $\|\boldsymbol{x}_\gamma - \boldsymbol{x}\|_2 \geq \|\boldsymbol{u}\|_2$, and the desired lower bound holds. We focus therefore on the case where γ has a concave shape, and coincides with with γ_2 (see Fig. A.1 right). The following equation holds using simple geometric arguments

$$R^2 = \sin(\theta) r'^2 + (R - r + r'\cos(\theta))^2. \tag{A.30}$$

where $r' = \|\boldsymbol{x}_\gamma - \boldsymbol{x}\|_2$. Solving this second order equation gives

$$r' = -(R - r)\cos(\theta) + \sqrt{(R-r)^2\cos^2(\theta) - r^2 + 2Rr}. \tag{A.31}$$

After some algebraic manipulations, we get

$$r' = \frac{r}{\cos(\theta)} \left(-\left(\frac{R}{r} - 1\right)\cos^2(\theta) + \frac{R}{r}\cos^2(\theta)\sqrt{1 + \tan^2(\theta)\frac{2Rr - r^2}{R^2}} \right). \tag{A.32}$$

Using the inequality in Lemma 8, together with the fact that $r\kappa < 1$, we obtain

$$r' \geq \frac{r}{\cos(\theta)}\left(\cos^2(\theta) + \frac{R}{r}\cos^2(\theta)\tan^2(\theta)\left(\frac{2Rr - r^2}{2R^2}\right) - \frac{R}{r}\frac{\cos^2(\theta)\tan^4(\theta)}{2}\left(\frac{2Rr - r^2}{2R^2}\right)^2\right). \tag{A.33}$$

Using simple trigonometric identities, the above expression is simplified to

$$r' \geq \frac{r}{\cos(\theta)}\left(1 + \frac{r}{R}\left(-\frac{\sin^2(\theta)}{2} - \frac{\sin^4(\theta)}{2\cos^2(\theta)}\left(1 - \frac{r}{2R}\right)^2\right)\right). \tag{A.34}$$

When expanding it, we obtain

$$r' \geq \frac{r}{\cos(\theta)}\left(1 - \left(\frac{\sin^2(\theta)}{2} + \frac{\sin^4(\theta)}{2\cos^2(\theta)}\right)\frac{r}{R} + \frac{\sin^4(\theta)}{2\cos^2(\theta)}\frac{r^2}{R^2} - \frac{\sin^4(\theta)}{8\cos^2(\theta)}\frac{r^3}{R^3}\right). \tag{A.35}$$

Since $\sin^2(\theta)\tan^2(\theta) = \tan^2(\theta) - \sin^2(\theta)$, we have

$$r' \geq \frac{r}{\cos(\theta)}\left(1 - \tan^2(\theta)\left(\frac{r}{2R} + \frac{r^3}{8R^3}\right)\right). \tag{A.36}$$

Using again the assumption $r/R < 1$, we obtain

$$r' \geq \frac{r}{\cos(\theta)}\left(1 - 0.625\tan^2(\theta)\frac{r}{R}\right). \tag{A.37}$$

Since $r/\cos(\theta) = \|\boldsymbol{u}\|_2$, one can rewrite it as

$$\frac{\|\boldsymbol{x}_\gamma - \boldsymbol{x}\|_2}{\|\boldsymbol{u}\|_2} - 1 \geq -0.625r\kappa\tan^2(\theta), \tag{A.38}$$

which completes the proof. □

We now use the previous lemma to bound the semi-random robustness of the classifier, i.e. $\|\boldsymbol{r}^k_{\mathcal{S}}\|_2$, to the worst-case robustness $\|\boldsymbol{r}^k\|_2$ in the case where the curvature is sufficiently small.

Theorem 2. *Let $\mathcal{S}$ be a random m-dimensional subspace of $\mathbb{R}^d$. Define $\alpha := \sqrt{m/d}$, and let $\kappa := \kappa(\mathscr{B}_k)$. Assuming that $\kappa \leq \frac{C\alpha^2}{\zeta_2(m,\delta)\|\boldsymbol{r}^k\|_2}$, the following inequalities hold between $\|\boldsymbol{r}^k_{\mathcal{S}}\|_2$ and the worst-case perturbation $\|\boldsymbol{r}^k\|_2$*

$$\frac{\zeta_1(m,\delta)}{\alpha^2}\left(1 - \frac{C_1\|\boldsymbol{r}^k\|_2\kappa\zeta_2(m,\delta)}{\alpha^2}\right)^2 \leq \frac{\|\boldsymbol{r}^k_{\mathcal{S}}\|_2^2}{\|\boldsymbol{r}^k\|_2^2} \leq \frac{\zeta_2(m,\delta)}{\alpha^2}\left(1 + \frac{C_2\|\boldsymbol{r}^k\|_2\kappa\zeta_2(m,\delta)}{\alpha^2}\right)^2 \tag{A.39}$$

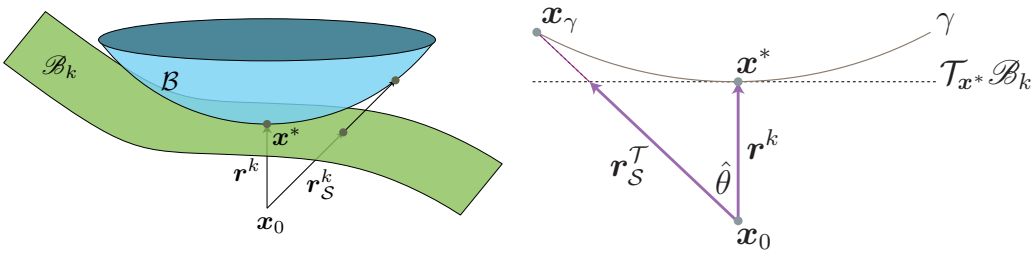


Figure A.2 – Left: To prove the upper bound, we consider a ball $\mathcal{B}$ included in $\mathcal{R}_k$ that intersects with the boundary at $\boldsymbol{x}^*$. Upper bounds on $\|\boldsymbol{r}^k_{\mathcal{S}}\|_2$ derived when the boundary is $\partial\mathcal{B}$ are also valid upper bounds for the real boundary $\mathscr{B}_k$. Right: Normal section to the decision boundary $\mathscr{B}_k = \partial\mathcal{B}$ along the normal plane $\mathcal{U} = \text{span}\left(\boldsymbol{r}^{\mathcal{T}}_{\mathcal{S}}, \boldsymbol{r}^k\right)$. We denote by γ the normal section of boundary $\mathscr{B}_k$, along the plane $\mathcal{U}$, and by $\mathcal{T}_{\boldsymbol{x}^*}\mathscr{B}_k$ the tangent space to the sphere $\partial\mathcal{B}$ at $\boldsymbol{x}^*$.

with probability larger than $1 - 4\delta$*. The constants can be taken* $C = 0.2, C_1 = 0.625, C_2 = 2.25$*.*

Proof of upper bound. Denote by $\boldsymbol{x}^*$ the point belonging to the boundary $\mathscr{B}_k$ that is closest to the original data point $\boldsymbol{x}_0$. By definition of the curvature κ, there exists a point $\boldsymbol{z}^*$ such that the ball $\mathcal{B}$ centered at $\boldsymbol{z}^*$ and of radius $1/\kappa = \|\boldsymbol{z}^* - \boldsymbol{x}^*\|_2$ is inscribed in the region $\mathcal{R}_k = \{x \in \mathbb{R}^d : f_k(\boldsymbol{x}) > f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x})\}$ (see Fig. A.2 (a)).[1]

Observe that the worst-case perturbation along any subspace $\mathcal{S}$ that reaches the ball $\mathcal{B}$ is larger than the perturbation along $\mathcal{S}$ that reaches the region $\mathcal{R}_k$, as $\mathcal{B} \subseteq \mathcal{R}_k$. Therefore, any upper bound derived when the boundary is the sphere of radius $1/\kappa$; i.e., $\mathscr{B}_k = \partial\mathcal{B}$ is also a valid upper bound for boundary $\mathscr{B}_k$ (see Fig. A.2 (a)). It is therefore sufficient to derive an upper bound in the worst case scenario where the boundary $\mathscr{B}_k = \partial\mathcal{B}$, and we consider this case for the remainder of the proof of the upper bound.

We now consider the linear classifier whose boundary is tangent to $\mathscr{B}_k$ at $\boldsymbol{x}^*$. For the random subspace $\mathcal{S}$, we denote by $\boldsymbol{r}^{\mathcal{T}}_{\mathcal{S}}$ the worst-case subspace perturbation for this linear classifier. We then focus on the intersection between the boundary $\mathscr{B}_k$ and the two-dimensional plane $\mathcal{U}$ spanned by the vectors $\boldsymbol{r}^k$ and $\boldsymbol{r}^{\mathcal{T}}_{\mathcal{S}}$. This *normal* section of the boundary cuts the ball $\mathcal{B}$ through its center as the tangent spaces of the decision boundary and the ball coincide. See Fig. A.2 for a clarifying figure of this two-dimensional cross-section. We define the angle $\hat{\theta}$ as denoted in Fig. A.2, such that $\cos(\hat{\theta}) = \frac{\|\boldsymbol{r}^k\|_2}{\|\boldsymbol{r}^{\mathcal{T}}_{\mathcal{S}}\|_2}$.

[1] For a fixed point $\boldsymbol{x}^*$ on the boundary, the maximal radius $1/\kappa$ might not be achieved. To prove the result in the general case where the supremum is not achieved, one can consider instead a sequence $(\kappa_n)_n$ converging to κ, such that the balls of radius $1/\kappa_n$ and intersecting the boundary at $\boldsymbol{x}^*$ are included in $\mathcal{R}_k$. The same proof and results follow by taking the limit on the bounds derived with ball of radius $1/\kappa_n$.

We apply our result on linear classifiers in Theorem 1 for the tangent classifier. We have

$$\frac{1}{\cos(\hat{\theta})^2} = \frac{\|\boldsymbol{r}_{\mathcal{S}}^{\mathcal{T}}\|_2^2}{\|\boldsymbol{r}^k\|_2^2} \leq \frac{1}{\alpha^2}\zeta_2(m,\delta), \tag{A.40}$$

with probability exceeding $1-2\delta$. Hence, using $\tan^2(\hat{\theta}) \leq (\cos^2(\hat{\theta}))^{-1}$ and the assumption of the theorem, we deduce that

$$\tan^2(\hat{\theta}) \leq \frac{1}{\alpha^2}\zeta_2(m,\delta) \leq \frac{0.2}{\kappa\|\boldsymbol{r}^k\|_2},$$

with probability exceeding $1-2\delta$. Note moreover that

$$\|\boldsymbol{r}^k\|_2\kappa \leq \frac{0.2\alpha^2}{\zeta_2(m,\delta)} < 1.$$

Hence, the assumptions of Lemma 4 hold with probability larger than $1-2\delta$. Using the notations of Fig. A.2, we therefore obtain from Lemma 4

$$\frac{\|\boldsymbol{x}_\gamma - \boldsymbol{x}_0\|_2}{\|\boldsymbol{r}_{\mathcal{S}}^{\mathcal{T}}\|_2} - 1 \leq C_2\kappa\|\boldsymbol{r}^k\|_2\tan^2(\hat{\theta}) \tag{A.41}$$

with probability larger than $1-2\delta$.

Observe that $\|\boldsymbol{x}_\gamma - \boldsymbol{x}_0\|_2 \geq \|\boldsymbol{r}_{\mathcal{S}}^k\|_2$, and that $\tan^2(\hat{\theta}) \leq \frac{\|\boldsymbol{r}_{\mathcal{S}}^T\|_2^2}{\|\boldsymbol{r}^k\|_2^2}$. Hence, we obtain by re-writing Eq. (A.41)

$$\mathbb{P}\left(\frac{\|\boldsymbol{r}_{\mathcal{S}}^k\|_2^2}{\|\boldsymbol{r}^k\|_2^2} \leq \left\{1 + C_2\kappa\|\boldsymbol{r}^k\|_2\frac{\|\boldsymbol{r}_{\mathcal{S}}^{\mathcal{T}}\|_2^2}{\|\boldsymbol{r}^k\|_2^2}\right\}^2 \frac{\|\boldsymbol{r}_{\mathcal{S}}^{\mathcal{T}}\|_2^2}{\|\boldsymbol{r}^k\|_2^2}\right) \geq 1-2\delta. \tag{A.42}$$

Using the inequality in Eq. (A.40), we obtain

$$\mathbb{P}\left(\frac{\|\boldsymbol{r}_{\mathcal{S}}^k\|_2^2}{\|\boldsymbol{r}^k\|_2^2} \leq \left\{1 + C_2\kappa\|\boldsymbol{r}^k\|_2\frac{\zeta_2(m,\delta)}{\alpha^2}\right\}^2 \frac{\zeta_2(m,\delta)}{\alpha^2}\right) \geq 1-2\delta,$$

which concludes the proof of the upper bound. □

Proof of the lower bound. We now consider the ball $\mathcal{B}'$ of center $\boldsymbol{z}^*$ and radius $1/\kappa = \|\boldsymbol{z}^* - \boldsymbol{x}^*\|_2$ that is included in the region $\mathcal{R}_{\hat{k}(\boldsymbol{x}_0)}$. Since the ball $\mathcal{B}'$ is, by definition, included in the region $\mathcal{R}_{\hat{k}(\boldsymbol{x}_0)}$, the worst-case scenario for the lower bound on $\|\boldsymbol{r}_{\mathcal{S}}^k\|_2$ occurs whenever the decision boundary $\mathscr{B}_k$ coincides with the ball $\mathcal{B}'$ (see Fig. A.3 (a)). We consider this case in the remainder of the proof.

To derive the lower bound, we consider the cross-section $\mathcal{U}'$ spanned by the vectors $\boldsymbol{r}_{\mathcal{S}}^k$ and

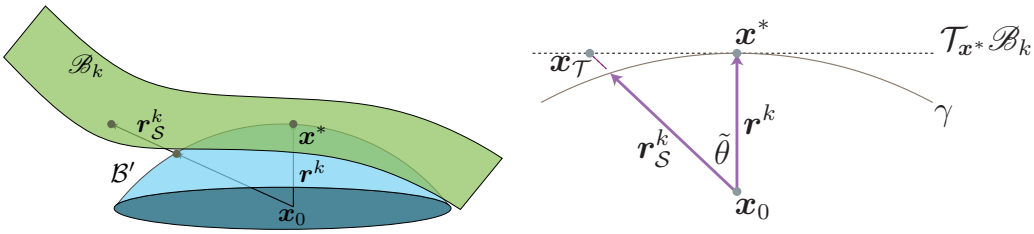


Figure A.3 – Left: To prove the lower bound, we consider a ball $\mathcal{B}'$ included in $\mathcal{R}_{\hat{k}(\boldsymbol{x}_0)}$ that intersects with the boundary at $\boldsymbol{x}^*$. Lower bounds on $\|\boldsymbol{r}_{\mathcal{S}}^k\|_2$ derived when the boundary is the sphere $\partial\mathcal{B}'$ are also valid lower bounds for the real boundary $\mathscr{B}_k$. Right: Cross section of the problem along the plane $\mathcal{U}' = \text{span}\left(\boldsymbol{r}_{\mathcal{S}}^k, \boldsymbol{r}^k\right)$. γ denotes the normal section of $\mathscr{B}_k = \mathcal{B}'$ along the plane $\mathcal{U}'$.

$\boldsymbol{r}^k$ (Fig. A.3 (b)). We have $\|\boldsymbol{r}^k\|_2\kappa < 1$; using the lower bound of Lemma 4, we obtain

$$-C_1\kappa\|\boldsymbol{r}^k\|_2 \tan^2(\tilde{\theta}) \leq \frac{\|\boldsymbol{r}_{\mathcal{S}}^k\|_2}{\|\boldsymbol{x}_{\mathcal{T}} - \boldsymbol{x}_0\|_2} - 1 \tag{A.43}$$

for any $\mathcal{S}$. Observe moreover that

$$\tan^2(\tilde{\theta}) \leq \frac{1}{\cos(\tilde{\theta})^2} = \frac{\|\boldsymbol{x}_{\mathcal{T}} - \boldsymbol{x}_0\|_2^2}{\|\boldsymbol{r}^k\|_2^2}.$$

Hence, the following bound holds:

$$\frac{\|\boldsymbol{x}_{\mathcal{T}} - \boldsymbol{x}_0\|_2^2}{\|\boldsymbol{r}^k\|_2^2}\left(1 - C_1\kappa\|\boldsymbol{r}^k\|_2\frac{\|\boldsymbol{x}_{\mathcal{T}} - \boldsymbol{x}_0\|_2^2}{\|\boldsymbol{r}^k\|_2^2}\right)^2 \leq \frac{\|\boldsymbol{r}_{\mathcal{S}}^k\|_2^2}{\|\boldsymbol{r}^k\|_2^2}.$$

Let $\boldsymbol{r}_{\mathcal{S}}^{\mathcal{T}}$ denote the worst-case perturbation belonging to subspace $\mathcal{S}$ for the *linear* classifier $\mathcal{T}_{x^*}\mathscr{B}_k$. It is not hard to see that $\boldsymbol{r}_{\mathcal{S}}^{\mathcal{T}}$ is *collinear* to $\boldsymbol{r}_{\mathcal{S}}^k$ (see Lemma 6 for a proof). Hence, we have $\boldsymbol{r}_{\mathcal{S}}^{\mathcal{T}} = \boldsymbol{x}_{\mathcal{T}} - \boldsymbol{x}_0$. By applying our result on linear classifiers in Theorem 1 for the tangent classifier $\mathcal{T}_{x^*}\mathscr{B}_k$, we have:

$$\mathbb{P}\left(\frac{\zeta_1(m,\delta)}{\alpha^2} \leq \frac{\|\boldsymbol{r}_{\mathcal{S}}^{\mathcal{T}}\|_2^2}{\|\boldsymbol{r}^k\|_2^2} \leq \frac{\zeta_2(m,\delta)}{\alpha^2}\right) \geq 1 - 2\delta.$$

We therefore conclude that

$$\mathbb{P}\left(\frac{\zeta_1(m,\delta)}{\alpha^2}\left\{1 - C_1\kappa\|\boldsymbol{r}^k\|_2\frac{\zeta_2(m,\delta)}{\alpha^2}\right\}^2 \leq \frac{\|\boldsymbol{r}_{\mathcal{S}}^k\|_2^2}{\|\boldsymbol{r}^k\|_2^2}\right) \geq 1 - 2\delta,$$

which concludes the proof of the lower bound.

□

The goal is now to extend the previous result, derived for binary classifiers, to the

multiclass classification case. To do so, we show the following lemma.

Lemma 5 (Binary case to multiclass)**.** *Let $p = \arg\min_i \|\boldsymbol{r}^i\|_2$. Define the deterministic set*

$$A = \left\{ k : \|\boldsymbol{r}^k\|_2 \geq 1.45\sqrt{\zeta_2(m,\delta)}\sqrt{\frac{d}{m}}\|\boldsymbol{r}^*\|_2 \right\}. \tag{A.44}$$

Assume that, for all $k \in A^c$, we have

$$\mathbb{P}\left(l \leq \frac{\|\boldsymbol{r}^k_{\mathcal{S}}\|_2}{\|\boldsymbol{r}^k\|_2} \leq u \right) \geq 1 - \delta. \tag{A.45}$$

and that

$$\mathbb{P}\left(\|\boldsymbol{r}^p_{\mathcal{S}}\|_2 \geq 1.45\sqrt{\zeta_2(m,\delta)}\sqrt{\frac{d}{m}}\|\boldsymbol{r}^*\|_2 \right) \leq t. \tag{A.46}$$

Then, we have

$$\mathbb{P}\left(l \leq \frac{\|\boldsymbol{r}^*_{\mathcal{S}}\|_2}{\|\boldsymbol{r}^*\|_2} \leq u \right) \geq 1 - (L+1)\delta - t. \tag{A.47}$$

Proof. Note first that

$$\mathbb{P}\left(\frac{\|\boldsymbol{r}^*_{\mathcal{S}}\|_2}{\|\boldsymbol{r}^*\|_2} \geq u \right) \leq \mathbb{P}\left(\left\{ \frac{\|\boldsymbol{r}^p_{\mathcal{S}}\|_2}{\|\boldsymbol{r}^p\|_2} \geq u \right\} \right) \leq \delta. \tag{A.48}$$

We now focus on bounding the other bad event probability $\mathbb{P}\left(\frac{\|\boldsymbol{r}^*_{\mathcal{S}}\|_2}{\|\boldsymbol{r}^*\|_2} \leq l \right)$. We have

$$\begin{aligned} \mathbb{P}\left(\frac{\|\boldsymbol{r}^*_{\mathcal{S}}\|_2}{\|\boldsymbol{r}^*\|_2} \leq l \right) &= \mathbb{P}\left(\min_{k \notin A} \|\boldsymbol{r}^k_{\mathcal{S}}\|_2 = \|\boldsymbol{r}^*_{\mathcal{S}}\|_2, \frac{\|\boldsymbol{r}^*_{\mathcal{S}}\|_2}{\|\boldsymbol{r}^*\|_2} \leq l \right) \\ &\quad + \mathbb{P}\left(\min_{k \in A} \|\boldsymbol{r}^k_{\mathcal{S}}\|_2 = \|\boldsymbol{r}^*_{\mathcal{S}}\|_2, \frac{\|\boldsymbol{r}^*_{\mathcal{S}}\|_2}{\|\boldsymbol{r}^*\|_2} \leq l \right) \end{aligned} \tag{A.49}$$

The first probability can be bounded as follows:

$$\mathbb{P}\left(\min_{k \notin A} \|\boldsymbol{r}^k_{\mathcal{S}}\|_2 = \|\boldsymbol{r}^*_{\mathcal{S}}\|_2, \frac{\|\boldsymbol{r}^*_{\mathcal{S}}\|_2}{\|\boldsymbol{r}^*\|_2} \leq l \right) \leq \mathbb{P}\left(\bigcup_{k \notin A} \frac{\|\boldsymbol{r}^*_{\mathcal{S}}\|_2}{\|\boldsymbol{r}^*\|_2} \leq l \right) \leq L\delta. \tag{A.50}$$

The second probability can also be bounded in the following way

$$\mathbb{P}\left(\min_{k\in A}\|\boldsymbol{r}_{\mathcal{S}}^{k}\|_2=\|\boldsymbol{r}_{\mathcal{S}}^{*}\|_2,\frac{\|\boldsymbol{r}_{\mathcal{S}}^{*}\|_2}{\|\boldsymbol{r}^{*}\|_2}\leq l\right)\leq\mathbb{P}\left(\min_{k\in A}\|\boldsymbol{r}_{\mathcal{S}}^{k}\|_2=\|\boldsymbol{r}_{\mathcal{S}}^{*}\|_2\right)$$
$$=\mathbb{P}\left(\exists k\in A,\|\boldsymbol{r}_{\mathcal{S}}^{k}\|_2\leq\|\boldsymbol{r}_{\mathcal{S}}^{*}\|_2\right). \tag{A.51}$$

Observe that, for $k\in A$, we have $\|\boldsymbol{r}_{\mathcal{S}}^{k}\|_2\geq\|\boldsymbol{r}^{k}\|_2\geq 1.45\sqrt{\zeta_2(m,\delta)}\sqrt{\frac{d}{m}}\|\boldsymbol{r}^{*}\|_2$. Hence, we conclude that

$$\mathbb{P}\left(\min_{k\in A}\|\boldsymbol{r}_{\mathcal{S}}^{k}\|_2=\|\boldsymbol{r}_{\mathcal{S}}^{*}\|_2,\frac{\|\boldsymbol{r}_{\mathcal{S}}^{*}\|_2}{\|\boldsymbol{r}^{*}\|_2}\leq l\right)\leq\mathbb{P}\left(1.45\sqrt{\zeta_2(m,\delta)}\sqrt{\frac{d}{m}}\|\boldsymbol{r}^{*}\|_2\leq\|\boldsymbol{r}_{\mathcal{S}}^{*}\|_2\right) \tag{A.52}$$
$$\leq\mathbb{P}\left(1.45\sqrt{\zeta_2(m,\delta)}\sqrt{\frac{d}{m}}\|\boldsymbol{r}^{*}\|_2\leq\|\boldsymbol{r}_{\mathcal{S}}^{p}\|_2\right)\leq t. \tag{A.53}$$

□

Corollary 1. *Let $\mathcal{S}$ be a random m-dimensional subspace of $\mathbb{R}^d$. Assume that, for all $k\notin A$, we have*

$$\kappa(\mathscr{B}_k)\|\boldsymbol{r}^{k}\|_2\leq\frac{0.2}{\zeta_2(m,\delta)}\frac{m}{d} \tag{A.54}$$

Then, we have

$$0.875\sqrt{\zeta_1(m,\delta)}\sqrt{\frac{d}{m}}\leq\frac{\|\boldsymbol{r}_{\mathcal{S}}^{*}\|_2}{\|\boldsymbol{r}^{*}\|_2}\leq 1.45\sqrt{\zeta_2(m,\delta)}\sqrt{\frac{d}{m}} \tag{A.55}$$

with probability larger than $1-4(L+2)\delta$.

Proof. Using Theorem 2, we have that for all $k\notin A$, the result in Eq. (A.39) holds. We simplify the result with the assumption $\kappa(\mathscr{B}_k)\|\boldsymbol{r}\|_2\leq\frac{0.2}{\zeta_2(m,\delta)}\frac{m}{d}$. Hence, the bounds of Theorem 2 are given as follows

$$\frac{\zeta_1(m,\delta)}{\alpha^2}\left(1-0.2C_1\right)^2\leq\frac{\|\boldsymbol{r}_{\mathcal{S}}^{k}\|_2^2}{\|\boldsymbol{r}^{k}\|_2^2}\leq\frac{\zeta_2(m,\delta)}{\alpha^2}\left(1+0.2C_2\right)^2, \tag{A.56}$$

which leads to the following bounds:

$$\zeta_1(m,\delta)\frac{d}{m}0.875^2\leq\frac{\|\boldsymbol{r}_{\mathcal{S}}^{k}\|_2^2}{\|\boldsymbol{r}^{k}\|_2^2}\leq\zeta_2(m,\delta)\frac{d}{m}1.45^2, \tag{A.57}$$

with probability exceeding $1-4\delta$.

By using Lemma 5, together with the fact that $t = \delta$, we obtain

$$\mathbb{P}\left(0.875\sqrt{\zeta_1(m,\delta)}\sqrt{\frac{d}{m}} \leq \frac{\|\boldsymbol{r}_{\mathcal{S}}^*\|_2}{\|\boldsymbol{r}^*\|_2} \leq 1.45\sqrt{\zeta_2(m,\delta)}\sqrt{\frac{d}{m}}\right) \geq 1 - 4(L+2)\delta, \tag{A.58}$$

which concludes the proof. □

Useful results

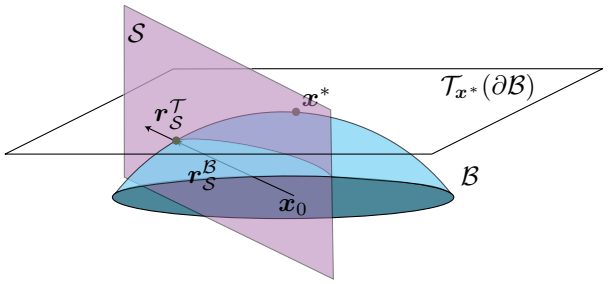


Figure A.4 – The worst-case perturbation in the subspace $\mathcal{S}$ when the decision boundary is $\partial\mathcal{B}$ and $T_{\boldsymbol{x}^*}(\partial\mathcal{B})$ (denoted respectively by $\boldsymbol{r}_{\mathcal{S}}^{\mathcal{B}}$ and $\boldsymbol{r}_{\mathcal{S}}^{\mathcal{T}}$) are collinear.

Lemma 6. *Let $\boldsymbol{x}_0 \in \mathbb{R}^d$, and $\boldsymbol{x}^*$ denote the closest point to $\boldsymbol{x}_0$ on the sphere $\partial\mathcal{B}$ (see Fig. A.4). Let $\mathcal{T}_{\boldsymbol{x}^*}(\partial\mathcal{B})$ be the tangent space to $\partial\mathcal{B}$ at $\boldsymbol{x}^*$. For an arbitrary subspace $\mathcal{S}$, let $\boldsymbol{r}_{\mathcal{S}}^{\mathcal{T}}$ and $\boldsymbol{r}_{\mathcal{S}}^{\mathcal{B}}$ denote the worst-case perturbations of $\boldsymbol{x}_0$ on the subspace $\mathcal{S}$, when the decision boundaries are respectively $\mathcal{T}_{\boldsymbol{x}^*}(\partial\mathcal{B})$ and $\partial\mathcal{B}$. Then, the two perturbations $\boldsymbol{r}_{\mathcal{S}}^{\mathcal{T}}$ and $\boldsymbol{r}_{\mathcal{S}}^{\mathcal{B}}$ are collinear.*

Proof. Assuming the center of the ball $\mathcal{B}$ is the origin, the points on the sphere $\partial\mathcal{B}$ satisfy equation: $\|\boldsymbol{x}\|_2 = R$, where R denotes the radius. Hence, the perturbation $\boldsymbol{r}_{\mathcal{S}}^{\mathcal{B}}$ is given by

$$\boldsymbol{r}_{\mathcal{S}}^{\mathcal{B}} = \underset{\boldsymbol{r}\in\mathbb{R}^d}{\arg\min} \|\boldsymbol{r}\|_2^2 \text{ such that } \|\boldsymbol{x}_0 + \mathbf{P}_{\mathcal{S}}\boldsymbol{r}\|_2^2 = R^2. \tag{A.59}$$

By equating the gradient of Lagrangian of the above constrained optimization problem to zero, we obtain the following necessary optimality condition

$$\boldsymbol{r} + \lambda\mathbf{P}_{\mathcal{S}}(\boldsymbol{x}_0 + \mathbf{P}_{\mathcal{S}}\boldsymbol{r}) = 0.$$

It should further be noted that $\mathbf{P}_{\mathcal{S}}\boldsymbol{r}_{\mathcal{S}}^{\mathcal{B}} = \boldsymbol{r}_{\mathcal{S}}^{\mathcal{B}}$. Indeed, if $\boldsymbol{r}_{\mathcal{S}}^{\mathcal{B}}$ had a component orthogonal to $\mathcal{S}$, the projection of $\boldsymbol{r}_{\mathcal{S}}^{\mathcal{B}}$ onto $\mathcal{S}$ would have strictly lower ℓ_2 norm, while still satisfying the condition in Eq.(A.59). Hence, the necessary condition of optimality becomes

$$(1+\lambda)\boldsymbol{r} + \lambda\mathbf{P}_{\mathcal{S}}\boldsymbol{x}_0 = 0,$$

from which we conclude that $\boldsymbol{r}_{\mathcal{S}}^{\mathcal{B}}$ is collinear to $\mathbf{P}_{\mathcal{S}}\boldsymbol{x}_0$.

It should further be noted that $\boldsymbol{r}_{\mathcal{S}}^{\mathcal{T}}$ can be computed in closed form, and is collinear to $\mathbf{P}_{\mathcal{S}}(\boldsymbol{x}^* - \boldsymbol{x}_0)$, which is itself collinear to $\boldsymbol{x}_0$, as the the center of the ball was assumed to be the origin. This concludes the proof. □

Lemma 7. *If* $x \in [0, 2(\sqrt{2}-1)]$,

$$\sqrt{1-x} \geq 1 - \frac{x}{2} - \frac{x^2}{4}. \tag{A.60}$$

Lemma 8. *If* $x \geq 0$,

$$\sqrt{1+x} \geq 1 + \frac{x}{2} - \frac{x^2}{8}. \tag{A.61}$$

B Appendix of Chapter 6

B.1 Proof of Theorem 3

We use Lemma 2 of Appendix A to prove our result.

Theorem 3. *Let $\xi \geq 0, \delta \geq 0$. Let $\mathcal{S}$ be an m dimensional subspace such that $\mathbb{P}_{\boldsymbol{x}\sim\mu}(\|P_{\mathcal{S}}\boldsymbol{r}(\boldsymbol{x})\|_2 \geq 1-\xi) = 1$, where $P_{\mathcal{S}}$ is the projection operator on the subspace. Assume moreover that $\mathscr{L}_s(\boldsymbol{x}, \rho)$ holds for almost all $\boldsymbol{x} \sim \mu$, with $\rho = \frac{\sqrt{em}}{\delta(1-\xi)}$. Then, there exists a universal noise vector $\boldsymbol{v}$, such that*

$$\|\boldsymbol{v}\|_2 \leq \rho$$

and

$$\mathbb{P}_{x\sim\mu}\left(\hat{k}(\boldsymbol{x}+\boldsymbol{v}) \neq \hat{k}(\boldsymbol{x}) \text{ or } \hat{k}(\boldsymbol{x}-\boldsymbol{v}) \neq \hat{k}(\boldsymbol{x})\right) \geq 1-\delta.$$

Proof. Define $\mathbb{S}$ to be the unit sphere centered at 0 in the subspace $\mathcal{S}$. Let $\rho = \frac{\sqrt{em}}{\delta(1-\xi)}$, and denote by $\rho\mathbb{S}$ the sphere scaled by ρ. We have

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{v}\sim\rho\mathbb{S}}\left(\mathbb{P}_{\boldsymbol{x}\sim\mu}\left(\hat{k}(\boldsymbol{x}+\boldsymbol{v}) \neq \hat{k}(\boldsymbol{x}) \text{ or } \hat{k}(\boldsymbol{x}-\boldsymbol{v}) \neq \hat{k}(\boldsymbol{x})\right)\right) \\
&= \mathbb{E}_{\boldsymbol{x}\sim\mu}\left(\mathbb{P}_{\boldsymbol{v}\sim\rho\mathbb{S}}\left(\hat{k}(\boldsymbol{x}+\boldsymbol{v}) \neq \hat{k}(\boldsymbol{x}) \text{ or } \hat{k}(\boldsymbol{x}-\boldsymbol{v}) \neq \hat{k}(\boldsymbol{x})\right)\right) \\
&\geq \mathbb{E}_{\boldsymbol{x}\sim\mu}\left(\mathbb{P}_{\boldsymbol{v}\sim\rho\mathbb{S}}\left(|\boldsymbol{r}(\boldsymbol{x})^T\boldsymbol{v}| - \|\boldsymbol{r}(\boldsymbol{x})\|_2^2 \geq 0\right)\right) \\
&= \mathbb{E}_{\boldsymbol{x}\sim\mu}\left(\mathbb{P}_{\boldsymbol{v}\sim\rho\mathbb{S}}\left(|(P_{\mathcal{S}}\boldsymbol{r}(\boldsymbol{x}) + P_{\mathcal{S}^{\text{orth}}}\boldsymbol{r}(\boldsymbol{x}))^T\boldsymbol{v}| - \|\boldsymbol{r}(\boldsymbol{x})\|_2^2 \geq 0\right)\right),
\end{aligned}$$

where $P_{\mathcal{S}^{\text{orth}}}$ denotes the projection operator on the orthogonal of $\mathcal{S}$. Observe that $(P_{\mathcal{S}^{\text{orth}}}\boldsymbol{r}(\boldsymbol{x}))^T\boldsymbol{v} = 0$. Note moreover that $\|\boldsymbol{r}(\boldsymbol{x})\|_2^2 = 1$ by assumption. Hence, the above

expression simplifies to

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mu}\left(\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\rho\mathbb{S}}\left(|(P_{\mathcal{S}}\boldsymbol{r}(\boldsymbol{x}))^T\boldsymbol{v}|-1\geq 0\right)\right)\\
=&\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mu}\left(\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\mathbb{S}}\left(|(P_{\mathcal{S}}\boldsymbol{r}(\boldsymbol{x}))^T\boldsymbol{v}|\geq \rho^{-1}\right)\right)\\
\geq&\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mu}\left(\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\mathbb{S}}\left(\left|\frac{(P_{\mathcal{S}}\boldsymbol{r}(\boldsymbol{x}))^T}{\|P_{\mathcal{S}}\boldsymbol{r}(\boldsymbol{x})\|_2}\boldsymbol{v}\right|\geq\frac{\delta}{\sqrt{em}}\right)\right),
\end{aligned}$$

where we have used the assumption of the projection of $\boldsymbol{r}(\boldsymbol{x})$ on the subspace $\mathcal{S}$. Hence, it follows from Lemma 2 that

$$\mathop{\mathbb{E}}_{\boldsymbol{v}\sim\rho\mathbb{S}}\left(\mathop{\mathbb{P}}_{\boldsymbol{x}\sim\mu}\left(\hat{k}(\boldsymbol{x}+\boldsymbol{v})\neq\hat{k}(\boldsymbol{x})\text{ or }\hat{k}(\boldsymbol{x}-\boldsymbol{v})\neq\hat{k}(\boldsymbol{x})\right)\right)\geq 1-\delta.$$

Hence, there exists a universal vector $\boldsymbol{v}$ of ℓ_2 norm ρ such that

$$\mathop{\mathbb{P}}_{\boldsymbol{x}\sim\mu}\left(\hat{k}(\boldsymbol{x}+\boldsymbol{v})\neq\hat{k}(\boldsymbol{x})\text{ or }\hat{k}(\boldsymbol{x}-\boldsymbol{v})\neq\hat{k}(\boldsymbol{x})\right)\geq 1-\delta.$$

□

B.2 Proof of Theorem 4

Theorem 4. *Let $\kappa>0,\delta>0$ and $m\in\mathbb{N}$. Assume that the quadratic decision boundary model $\mathcal{Q}(\boldsymbol{x},\rho)$ holds for almost all $\boldsymbol{x}\sim\mu$, with $\rho=\sqrt{\frac{2\log(2/\delta)}{m}}\kappa^{-1}+\kappa^{-1/2}$. Let $\mathcal{S}$ be a m dimensional subspace such that*

$$\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\mathbb{S}}\left(\forall\boldsymbol{u}\in\mathbb{R}^2,\alpha_x^{-1}\boldsymbol{u}^T H_{\boldsymbol{z}}^{\boldsymbol{r}(\boldsymbol{x}),\boldsymbol{v}}\boldsymbol{u}\geq\kappa\|\boldsymbol{u}\|_2^2\right)\geq 1-\beta\text{ for almost all }\boldsymbol{x}\sim\mu,$$

where $H_{\boldsymbol{z}}^{\boldsymbol{r}(\boldsymbol{x}),\boldsymbol{v}}=\Pi^T H_{\boldsymbol{z}}\Pi$ with Π an orthonormal basis of span$(\boldsymbol{r}(\boldsymbol{x}),\boldsymbol{v})$, and $\mathbb{S}$ denotes the unit sphere in $\mathcal{S}$. Then, there is a universal perturbation vector $\boldsymbol{v}$ such that $\|\boldsymbol{v}\|_2\leq\rho$ and $\mathop{\mathbb{P}}_{\boldsymbol{x}\sim\mu}\left(\hat{k}(\boldsymbol{x}+\boldsymbol{v})\neq\hat{k}(\boldsymbol{x})\right)\geq 1-\delta-\beta$.

Proof. Let $\boldsymbol{x}\sim\mu$. We have

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\boldsymbol{v}\sim\rho\mathbb{S}}\left(\mathop{\mathbb{P}}_{\boldsymbol{x}\sim\mu}\left(\hat{k}(\boldsymbol{x}+\boldsymbol{v})\neq\hat{k}(\boldsymbol{x})\right)\right)\\
=&\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mu}\left(\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\rho\mathbb{S}}\left(\hat{k}(\boldsymbol{x}+\boldsymbol{v})\neq\hat{k}(\boldsymbol{x})\right)\right)\\
\geq&\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mu}\left(\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\rho\mathbb{S}}\left(\alpha_x^{-1}(\boldsymbol{v}-\boldsymbol{r})^T H_z(\boldsymbol{v}-\boldsymbol{r})+\boldsymbol{r}^T(\boldsymbol{v}-\boldsymbol{r})\geq 0\right)\right)\\
=&\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mu}\left(\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\mathbb{S}}\left(\alpha_x^{-1}(\rho\boldsymbol{v}-\boldsymbol{r})^T H_z(\rho\boldsymbol{v}-\boldsymbol{r})+\boldsymbol{r}^T(\rho\boldsymbol{v}-\boldsymbol{r})\geq 0\right)\right)
\end{aligned}$$

Using the assumptions of the theorem, we have

$$\begin{aligned}
&\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\mathbb{S}}\left(\alpha_x^{-1}(\rho\boldsymbol{v}-\boldsymbol{r})^T H_z(\rho\boldsymbol{v}-\boldsymbol{r})+\boldsymbol{r}^T(\rho\boldsymbol{v}-\boldsymbol{r})\leq 0\right)\\
\leq &\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\mathbb{S}}\left(\kappa\|\rho\boldsymbol{v}-\boldsymbol{r}\|_2^2+\boldsymbol{r}^T(\rho\boldsymbol{v}-\boldsymbol{r})\leq 0\right)+\beta\\
\leq &\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\mathbb{S}}\left(\rho(1-2\kappa)\boldsymbol{v}^T\boldsymbol{r}+\kappa\rho^2+(\kappa-1)\leq 0\right)+\beta\\
\leq &\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\mathbb{S}}\left(\rho(1-2\kappa)\boldsymbol{v}^T\boldsymbol{r}\leq -\epsilon\right)+\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\mathbb{S}}\left(\kappa\rho^2+(\kappa-1)\leq\epsilon\right)+\beta,
\end{aligned}$$

for $\epsilon>0$. The goal is therefore to find ρ such that $\kappa\rho^2+(\kappa-1)\geq\epsilon$, together with $\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\mathbb{S}}\left(\rho(1-2\kappa)\boldsymbol{v}^T\boldsymbol{r}\leq-\epsilon\right)\leq\delta$. Let $\rho^2=\frac{\epsilon+1}{\kappa}$. Using the concentration of measure on the sphere [63], we have

$$\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\mathbb{S}}\left(\boldsymbol{v}^T\boldsymbol{r}\leq\frac{-\epsilon}{\rho(1-2\kappa)}\right)\leq 2\exp\left(-\frac{m\epsilon^2}{2\rho^2(1-2\kappa)^2}\right).$$

To bound the above probability by δ, we set $\epsilon=C\frac{\rho}{\sqrt{m}}$, where $C=\sqrt{2\log(2/\delta)}$. We therefore choose ρ such that

$$\rho^2=\kappa^{-1}\left(C\rho m^{-1/2}+1\right)$$

The solution of this second order equation gives

$$\rho=\frac{C\kappa^{-1}m^{-1/2}+\sqrt{\kappa^{-2}C^2m^{-1}+4\kappa^{-1}}}{2}\leq C\kappa^{-1}m^{-1/2}+\kappa^{-1/2}.$$

Hence, for this choice of ρ, we have by construction

$$\mathop{\mathbb{P}}_{\boldsymbol{v}\sim\mathbb{S}}\left(\alpha_x^{-1}(\rho\boldsymbol{v}-\boldsymbol{r})^T H_z(\rho\boldsymbol{v}-\boldsymbol{r})+\boldsymbol{r}^T(\rho\boldsymbol{v}-\boldsymbol{r})\leq 0\right)\leq\delta+\beta.$$

We therefore conclude that $\mathop{\mathbb{E}}_{\boldsymbol{v}\sim\rho\mathbb{S}}\left(\mathop{\mathbb{P}}_{\boldsymbol{x}\sim\mu}\left(\hat{k}(\boldsymbol{x}+\boldsymbol{v})\neq\hat{k}(\boldsymbol{x})\right)\right)\geq 1-\delta-\beta$. This shows the existence of a universal noise vector $\boldsymbol{v}\sim\rho\mathbb{S}$ such that $\hat{k}(\boldsymbol{x}+\boldsymbol{v})\neq\hat{k}(\boldsymbol{x})$ with probability larger than $1-\delta-\beta$. □

Bibliography

[1] Understanding adversarial training: Increasing local stability of supervised models through robust optimization. *Neurocomputing*, 307:195 – 204, 2018.

[2] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning (ICML)*, 2018.

[3] Michaël Aupetit and Thibaud Catz. High-dimensional labeled data analysis with topology representing graphs. *Neurocomputing*, 63:139–169, 2005.

[4] Shumeet Baluja and Ian Fischer. Learning to attack: Adversarial transformation networks, 2018.

[5] Osbert Bastani, Yani Ioannou, Leonidas Lampropoulos, Dimitrios Vytiniotis, Aditya Nori, and Antonio Criminisi. Measuring neural net robustness with constraints. In *Neural Information Processing Systems (NIPS)*, 2016.

[6] Melika Behjati, Seyed-Mohsen Moosavi-Dezfooli, Mahdieh Soleymani Baghshah, and Pascal Frossard. Universal adversarial attacks on text classifiers. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2019.

[7] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Srndic, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 387–402, 2013.

[8] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *International Conference on Learning Representations*, 2018.

[9] Jacob Buckman, Aurko Roy, Colin Raffel, and Ian Goodfellow. Thermometer encoding: One hot way to resist adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2018.

[10] Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

[11] N. Carlini and D. Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. In *2018 IEEE Security and Privacy Workshops (SPW)*, 2018.

[12] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 2017.

[13] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE, 2017.

[14] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *British Machine Vision Conference*, 2014.

[15] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, and Yann LeCun. Entropy-sgd: Biasing gradient descent into wide valleys. In *International Conference on Learning Representations (ICLR)*, 2017.

[16] Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2014.

[17] Moustapha Cisse, Yossi Adi, Natalia Neverova, and Joseph Keshet. Houdini: Fooling deep structured prediction models. *arXiv preprint arXiv:1707.05373*, 2017.

[18] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *International Conference on Machine Learning (ICML)*, 2017.

[19] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Siwei Li, Li Chen, Michael E. Kounavis, and Duen Horng Chau. Shield: Fast, practical defense and vaccination for deep learning using jpeg compression. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018.

[20] Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003.

[21] Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2933–2941, 2014.

[22] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *International Conference on Machine Learning (ICML)*, 2017.

[23] Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. Hotflip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018.

[24] Gamaleldin Elsayed, Dilip Krishnan, Hossein Mobahi, Kevin Regan, and Samy Bengio. Large margin deep networks for classification. In *Advances in Neural Information Processing Systems*, 2018.

[25] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[26] Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Analysis of classifiers' robustness to adversarial perturbations. *CoRR*, abs/1502.02590, 2015.

[27] Alhussein Fawzi and Pascal Frossard. Manitest: Are classifiers really invariant? In *British Machine Vision Conference (BMVC)*, pages 106.1–106.13, 2015.

[28] Alhussein Fawzi and Pascal Frossard. Measuring the effect of nuisance variables on classifiers. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2016.

[29] Alhussein Fawzi*, Seyed Moosavi-Dezfooli*, and Pascal Frossard. Robustness of classifiers: from adversarial to random noise. In *Neural Information Processing Systems (NIPS)*, 2016.

[30] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. The robustness of deep networks: A geometrical perspective. *IEEE Signal Processing Magazine*, 34(6):50–62, 2017.

[31] Alhussein Fawzi*, Seyed-Mohsen Moosavi-Dezfooli*, Pascal Frossard, and Stefano Soatto. Empirical study of the topology and geometry of deep networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[32] Reuben Feinman, Ryan R Curtin, Saurabh Shintre, and Andrew B Gardner. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017.

[33] Jean-Yves Franceschi, Alhussein Fawzi, and Omar Fawzi. Robustness of classifiers to uniform ℓ_p and Gaussian noise. In *21st International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018.

[34] C Daniel Freeman and Joan Bruna. Topology and geometry of half-rectified network optimization. In *International Conference on Learning Representations (ICLR)*, 2016.

[35] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[36] Shixiang Gu and Luca Rigazio. Towards deep neural network architectures robust to adversarial examples. *CoRR*, abs/1412.5068, 2014.

[37] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens van der Maaten. Countering adversarial images using input transformations. In *International Conference on Learning Representations (ICLR)*, 2018.

[38] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[39] Jan Hendrik Metzen, Mummadi Chaithanya Kumar, Thomas Brox, and Volker Fischer. Universal adversarial perturbations against semantic image segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2755–2764, 2017.

[40] Ruitong Huang, Bing Xu, Dale Schuurmans, and Csaba Szepesvári. Learning with a strong adversary. *CoRR, abs/1511.03034*, 2015.

[41] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

[42] Daniel Jakubovitz and Raja Giryes. Improving dnn robustness to adversarial attacks using jacobian regularization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[43] Saumya Jetley, Nicholas Lord, and Philip Torr. With friends like these, who needs adversaries? In *Neural Information Processing Systems (NIPS)*, 2018.

[44] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM International Conference on Multimedia (MM)*, pages 675–678. ACM, 2014.

[45] Can Kanbak, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Measuring robustness of classifiers to geometric transformations. 2017.

[46] Can Kanbak, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Geometric robustness of deep networks: analysis and improvement. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[47] Valentin Khrulkov and Ivan Oseledets. Art of singular vectors and universal adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8562–8570, 2018.

[48] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[49] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems (NIPS)*, pages 1097–1105, 2012.

[50] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *International Conference on Learning Representations (ICLR)*, 2017.

[51] Quoc V Le, Will Y Zou, Serena Y Yeung, and Andrew Y Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3361–3368. IEEE, 2011.

[52] Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[53] Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. Object recognition with gradient-based learning. In *Shape, contour and grouping in computer vision*, pages 319–345. 1999.

[54] Jeffrey M Lee. *Manifolds and differential geometry*, volume 107. American Mathematical Society Providence, 2009.

[55] Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Defense against adversarial attacks using high-level representation guided denoiser. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[56] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. 2014.

[57] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. In *International Conference on Learning Representations (ICLR)*, 2017.

[58] Yujia Liu, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. A geometry-inspired decision-based attack. *arXiv preprint arXiv:1903.10826*, 2019.

[59] Jiajun Lu, Theerasit Issaranon, and David Forsyth. Safetynet: Detecting and rejecting adversarial examples robustly. In *International Conference on Computer Vision (ICCV)*, 2017.

[60] Yan Luo, Xavier Boix, Gemma Roig, Tomaso A. Poggio, and Qi Zhao. Foveation-based mechanisms alleviate adversarial examples. *CoRR*, abs/1511.06292, 2015.

[61] Chunchuan Lyu, Kaizhu Huang, and Hai-Ning Liang. A unified gradient regularization family for adversarial examples. In *IEEE International Conference on Data Mining*, 2015.

[62] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018.

[63] Jiri Matousek. *Lectures on discrete geometry*, volume 108. Springer New York, 2002.

[64] Ofer Melnik and Jordan Pollack. Using graphs to analyze high-dimensional classifiers. In *Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on*, volume 3, pages 425–430. IEEE, 2000.

[65] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. In *International Conference on Learning Representations (ICLR)*, 2017.

[66] Apostolos Modas, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Sparsefool: a few pixels make a big difference. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[67] Guido F Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. In *Advances In Neural Information Processing Systems*, pages 2924–2932, 2014.

[68] Seyed-Mohsen Moosavi-Dezfooli*, Alhussein Fawzi*, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[69] Seyed-Mohsen Moosavi-Dezfooli*, Alhussein Fawzi*, Omar Fawzi, Pascal Frossard, and Stefano Soatto. Robustness of classifiers to universal perturbations: A geometric perspective. In *Sixth International Conference on Learning Representations*, 2018.

[70] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[71] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. Robustness via curvature regularization, and vice versa. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[72] Seyed-Mohsen Moosavi-Dezfooli, Ashish Shrivastava, and Oncel Tuzel. Divide, denoise, and defend against adversarial attacks. *arXiv preprint arXiv:1802.06806*, 2018.

[73] Konda Reddy Mopuri, Utsav Garg, and R Venkatesh Babu. Fast feature fool: A data independent approach to universal adversarial perturbations. In *British Machine Vision Conference (BMVC)*, 2017.

[74] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, 2011.

[75] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS P)*, 2016.

[76] Ben Poole, Subhaneil Lahiri, Maithreyi Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. In *Advances In Neural Information Processing Systems*, pages 3360–3368, 2016.

[77] Jonas Rauber and Wieland Brendel. Robust vision benchmark. https://github.com/bethgelab/robust-vision-benchmark.

[78] Andrew Slavin Ross and Finale Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *AAAI*, 2018.

[79] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[80] Andrzej P Ruszczyński. *Nonlinear optimization*, volume 13. Princeton university press, 2006.

[81] Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J Fleet. Adversarial manipulation of deep representations. In *International Conference on Learning Representations (ICLR)*, 2016.

[82] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1528–1540. ACM, 2016.

[83] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2014.

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

[85] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[86] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014.

[87] Pedro Tabacof and Eduardo Valle. Exploring the space of adversarial images. *IEEE International Joint Conference on Neural Networks*, 2016.

[88] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1701–1708, 2014.

[89] Thomas Tanay and Lewis Griffin. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv preprint arXiv:1608.07690*, 2016.

[90] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations (ICLR)*, 2018.

[91] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2019.

[92] Jonathan Uesato, Brendan O'Donoghue, Aaron van den Oord, and Pushmeet Kohli. Adversarial risk and the dangers of evaluating against weak attacks. In *International Conference on Machine Learning (ICML)*, 2018.

[93] Andrea Vedaldi and Karel Lenc. Matconvnet: Convolutional neural networks for matlab. In *ACM International Conference on Multimedia (MM)*, pages 689–692, 2015.

[94] Homer F Walker and Layne T Watson. Least-change secant update methods for underdetermined systems. *SIAM Journal on numerical analysis*, 27(5):1227–1262, 1990.

[95] David Warde-Farley, Ian Goodfellow, T Hazan, G Papandreou, and D Tarlow. Adversarial perturbations of deep neural networks. *Perturbations, Optimization, and Statistics*, 2016.

[96] Chaowei Xiao, Jun-Yan Zhu, Bo Li, Warren He, Mingyan Liu, and Dawn Song. Spatially transformed adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2018.

[97] Kai Y. Xiao, Vincent Tjeng, Nur Muhammad (Mahi) Shafiullah, and Aleksander Madry. Training for faster adversarial robustness verification via inducing reLU stability. In *International Conference on Learning Representations*, 2019.

[98] Zhewei Yao, Amir Gholami, Qi Lei, Kurt Keutzer, and Michael W Mahoney. Hessian-based analysis of large batch training and robustness to adversaries. In *Advances in Neural Information Processing Systems*, pages 4949–4959, 2018.

[99] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2016.

[100] Wen Zhou, Xin Hou, Yongjun Chen, Mengyun Tang, Xiangqi Huang, Xiang Gan, and Yong Yang. Transferable adversarial perturbations. In *The European Conference on Computer Vision (ECCV)*, September 2018.

Seyed Moosavi

June 2019

+41 78 849 8157

http://smoosavi.me

moosavi.sm@gmail.com

Interests

Analysing adversarial vulnerability of machine learning systems
Understanding geometric characteristics of deep image classifiers
Worst-case analysis for interpreting deep networks
Security of machine learning systems

Education

Sep 2014 - May 2019 Ph.D. in Computer and Communication Sciences
École Polytechnique Fédérale de Lausanne (EPFL)
Thesis advisor: Prof. Pascal Frossard
Thesis title: Geometry of adversarial robustness of deep networks: methods and applications

Sep 2012 - Aug 2014 M.Sc. in Communication Systems
École Polytechnique Fédérale de Lausanne (EPFL)
GPA: 5.72/6 via 120 credits
Specialization: Signals, images and interfaces
Supervisors: Prof. Martin Vetterli, EPFL
Dr. Yvonne-Anne Pignolet, ABB.

Sep 2008 - Aug 2012 B.Sc. in Electrical Engineering (Telecommunications)
Amirkabir University of Technology (Tehran Polytechnic)
GPA: 19.35/20 via 140 credits

Work Experience

Apr 2018 - Jun 2018 Apple Inc., Cupertino, CA.
PhD Research Intern
Hosted by: Dr. Oncel Tuzel

Sep 2017 - Dec 2017 Apple Inc., Cupertino, CA.
PhD Research Intern
Hosted by: Dr. Oncel Tuzel

Jul 2017 University of California, Los Angeles (UCLA)
Visiting Researcher
Hosted by: Prof. Stefano Soatto

Feb 2014 - Aug 2014 ABB Corporate Research, Baden-Daettwil
Research Intern
Hosted by: Dr. Yvonne-Anne Pignolet
Dr. Dacfey Dzung

Honors and Awards

2014 Awarded the Fellowship in Computer and Communication Sciences, EPFL, Switzerland.

2012 Recipient of Faculty of Communication and Computer Sciences Research Scholarship (top-5% of admissions), EPFL, Switzerland.

2012 Best B.Sc. Graduate Award in Electrical Engineering, Amirkabir University of Technology.

Outreach

2019 My work has been mentioned in:
Le Monde

2018 I am interviewed by:
EPFL News, Die Republik (in German)

2017 My work has been mentioned in:
The Verge, BBC, iProgrammer (1), (2), and (3).

Seyed Moosavi

June 2019

+41 78 849 8157

http://smoosavi.me

moosavi.sm@gmail.com

Student Supervision

Jul 2018 - Sep 2018	Yujia Liu Title: Query-efficient black-box adversarial examples.	M.Sc. Research Intern
Jul 2018 - Sep 2018	Melika Behjati Title: Universal perturbations for text classifiers.	M.Sc. Research Intern
Feb 2018 - Nov 2018	Apostolos Modas Title: Fast methods to compute sparse adversarial perturbations.	Doctoral Project
Feb 2017 - Nov 2017	Can Kanbak Title: Measuring adversarial invariance of deep networks.	Master Thesis
Sep 2016 - Jan 2017	Eric Bezzam Title: DeepFool applied to commercial classifiers.	M.Sc. Semester Project
Jul 2016 - Sep 2016	Christos Nikolaou Title: Parallel implementation of DeepFool in Caffe.	B.Sc. Research Intern
Feb 2016 - Jul 2016	Zhongqi Shi Title: The effect of architecture on adversarial robustness.	M.Sc. Semester Project

Teaching Experience

Fall 2015-16	Digital Signal Processing (B.Sc., EPFL) Instructors: Prof. Pascal Frossard Prof. Jean-Philippe Thiran	Teaching Assistance
Fall 2014	Audio Signal Processing (M.Sc., EPFL) Instructors: Dr. Christof Faller Dr. Dirk Schroeder	Teaching Assistance
Spring 2012	Probability and Statistics (B.Sc., AUT) Instructor: Dr. Gholamreza Moradi	Teaching Assistance
Spring 2011	Signals and Systems (B.Sc., AUT) Instructor: Dr. Farzaneh Abdollahi	Teaching Assistance

Research Talks

May 2019	A geometric perspective on the robustness of deep networks University of Amsterdam, Netherlands.
Jul and Aug 2018	Geometric robustness of deep networks Google Zurich, Switzerland, Deep Learning Summer School, University of Tehran, Iran.
May and Jul 2018	Robustness meets geometry in deep networks Vector Institute, University of Toronto, Canada, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran.
Dec 2017	Robustness of image classifiers Sharif University of Technology, Tehran, Iran.
Nov 2016	Robustness of classifiers beyond random noise Swiss Machine Learning Day (SMLD), Lausanne, Switzerland.
Nov 2015	Fast computation of adversarial examples in convolutional networks Swiss Machine Learning Day (SMLD), Lausanne, Switzerland.

Seyed Moosavi

June 2019

+41 78 849 8157

http://smoosavi.me

moosavi.sm@gmail.com

Publications

1. Robustness via curvature regularization, and vice versa, S. M. Moosavi-Dezfooli, A. Fawzi, J. Uesato and P. Frossard, *Conference on Computer Vision and Pattern Recognition (CVPR 2019)*, June 2019, California, US.
2. SparseFool: a few pixels make a big difference, A. Modas, S. M. Moosavi-Dezfooli and P. Frossard, *Conference on Computer Vision and Pattern Recognition (CVPR 2019)*, June 2019, California, US.
3. Geometric robustness of deep networks: analysis and improvement, C. Kanbak, S. M. Moosavi-Dezfooli and P. Frossard, *Conference on Computer Vision and Pattern Recognition (CVPR 2018)*, June 2018, Utah, US.
4. Empirical study of the topology and geometry of deep networks, A. Fawzi, S. M. Moosavi-Dezfooli, P. Frossard and S. Soatto, *Conference on Computer Vision and Pattern Recognition (CVPR 2018)*, June 2018, Utah, US. (Spotlight)
5. Robustness of Classifiers to Universal Perturbations: A Geometric Perspective, S. M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard and S. Soatto, *Sixth International Conference on Learning Representations (ICLR 2018)*, May 2018, Vancouver, Canada.
6. Adaptive Quantization for Deep Neural Network, Y. Zhou, S. M. Moosavi-Dezfooli, N. M. Cheung and P. Frossard, *AAAI Conference 2018*, Louisiana, US.
7. The Robustness of Deep Networks: A Geometrical Perspective, A. Fawzi, S. M. Moosavi-Dezfooli and P. Frossard, *IEEE Signal Processing Magazine* 34 (6), 50-62, 2017.
8. Universal adversarial perturbations, S. M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi and P. Frossard, *Conference on Computer Vision and Pattern Recognition (CVPR 2017)*, July 2017, Hawaii, US. (Oral)
9. Robustness of classifiers: from adversarial to random noise, A. Fawzi, S. M. Moosavi-Dezfooli and P. Frossard, *Advances in Neural Information Processing Systems (NIPS 2016)*, December 2016, Barcelona, Spain.
10. DeepFool: a simple and accurate method to fool deep neural networks, S. M. Moosavi-Dezfooli, A. Fawzi and P. Frossard, *Conference on Computer Vision and Pattern Recognition (CVPR 2016)*, June 2016, Nevada, US.
11. Simultaneous Acoustic Localization of Multiple Smartphones with Euclidean Distance Matrices, S. M. Moosavi-Dezfooli, Y.-A. Pignolet-Oswald and D. Dzung, *International Conference on Embedded Wireless Systems and Networks (EWSN 2016)*, February 2016, Graz, Austria.

* Pre-prints:

12. A geometry-inspired decision-based attack, Y. Liu, S. M. Moosavi-Dezfooli, P. Frossard.
13. Divide, Denoise, and Defend against Adversarial Attacks, S. M. Moosavi-Dezfooli, A. Shrivastava and O. Tuzel, arXiv pre-print, 2018.

Reviewing Service

Conferences	IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2018), Advances in Neural Information Processing Systems (NeurIPS 2016, 2018), International Conference on Machine Learning (ICML 2019), Data Compression Conference (DCC 2018), European Signal Processing Conference (EUSIPCO 2018).
Journals	IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), IEEE Transactions on Neural Networks and Learning Systems (TNNLS)

Languages

Persian	Native
English	Fluent
French	Intermediate (A2/B1)
German	Beginner (A1/A2)

**Ce document a été imprimé au Centre d'impression EPFL,
imprimerie climatiquement neutre, certifiée myClimate.**