



Understanding the role of randomization in adversarially robust classification

Lucas Gnecco Heredia

► To cite this version:

Lucas Gnecco Heredia. Understanding the role of randomization in adversarially robust classification. Computer science. Université Paris sciences et lettres, 2025. English. <NNT : 2025UPSLD009>. <tel-05273037>

HAL Id: tel-05273037

<https://theses.hal.science/tel-05273037v1>

Submitted on 22 Sep 2025

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



HAL Authorization



THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à Université Paris Dauphine - PSL

**Understanding the role of randomization in adversarially
robust classification**

Soutenue par

Lucas GNECCO HEREDIA

Le 21 mai 2025

École doctorale n°543

SDOSE

Spécialité

Informatique

Préparée au

LAMSADE

Dauphine | PSL 
UNIVERSITÉ PARIS

Composition du jury :

Mme Isabelle BLOCH Professeur, Sorbonne Université	<i>Président du jury</i>
M. Teddy FURON Directeur de recherche, Inria Rennes	<i>Rapporteur</i>
M. Varun JOG Professeur, University of Cambridge	<i>Rapporteur</i>
M. Amaury HABRARD Professeur, Université Jean Monnet Saint-Étienne	<i>Examineur</i>
M. Rafael PINOT Chaire de professeur junior (CPJ), Sorbonne U.	<i>Examineur</i>
M. Yann CHEVALEYRE Professeur, Paris Dauphine - PSL	<i>Encadrant</i>
M. Benjamin NEGREVERGNE Maître de conférences, Paris Dauphine - PSL	<i>Co-encadrant</i>

Understanding the role of randomization in adversarially robust classification

Lucas Gnecco Heredia

*Dissertation submitted in partial fulfillment of the requirements for the
degree of Doctor of Philosophy*

Université Paris-Dauphine - PSL

LAMSADE
Machine Intelligence and Learning Systems - MILES
Informatique

2025

Lucas Gnecco Heredia

Understanding the role of randomization in adversarially robust classification

Reviewers: M. Teddy FURON and M. Varun JOG

Supervisors: M. Yann CHEVALEYRE and M. Benjamin NEGREVERGNE

Université Paris-Dauphine - PSL

LAMSADE

Machine Intelligence and Learning Systems - MILES

Informatique

Place du Maréchal de Lattre de Tassigny

75 775, Paris, FRANCE

Abstract

This thesis addresses the challenge of robust classification in the presence of adversarial perturbations at inference time, commonly referred to as adversarial attacks. Adversarial attacks involve imperceptible modifications to input data designed to mislead trained classifiers into making errors. The existence of such attacks has been demonstrated across a wide range of machine learning models, including deep neural networks, and has raised concerns about the security of these models in safety-critical applications. Beyond the immediate security implications, there remains a limited understanding of the fundamental mechanisms that render deep neural networks vulnerable to adversarial attacks and how to design models that are inherently robust to such threats. Among the proposed solutions, *randomization* has emerged as a promising approach. Randomization-based techniques aim to introduce stochasticity during inference to mitigate the effectiveness of adversarial attacks. While conceptually appealing, the practical efficacy of randomization as a defense mechanism remains poorly understood, and its application has so far provided only limited protection against adversarial threats.

In this thesis, we explore the effectiveness of randomization as a defense mechanism against adversarial attacks. Our investigation is structured around three key perspectives: the attacker’s perspective, the defender’s perspective, and the theoretical implications of randomization in both binary and multiclass classification settings.

First, we adopt the attacker’s perspective and analyze the limitations of existing attack strategies against a specific class of randomized classifiers known as finite mixtures. We reformulate the problem of attacking such models as a combinatorial optimization problem and introduce a novel attack algorithm with interesting theoretical guarantees.

Next, we shift to the defender’s perspective and examine the potential benefits of randomization in the context of binary classification. We demonstrate that while randomization can offer certain advantages in specific scenarios, it is always theoretically possible to construct a deterministic classifier that matches or exceeds the

robustness of any given randomized classifier. We discuss the practical implications of this finding for commonly used randomized classifiers, such as finite mixtures and noise injection classifiers.

Finally, we extend our analysis to the multiclass classification setting, where we uncover a striking contrast to the binary case. Unlike in binary classification, where a deterministic classifier can always achieve comparable or superior robustness regardless of the data distribution or randomized classifier, we show that in multiclass settings, there exist data distributions for which no deterministic classifier can match the robustness of the best randomized classifier. We partially characterize the discrete data distributions with finite support for which this phenomenon holds and construct a family of distributions where the performance gap between the best deterministic and randomized classifiers can be made arbitrarily close to $\frac{1}{2}$.

This thesis will conclude with open questions and perspectives for future research in the field of adversarial robustness and randomization.

Résumé

Cette thèse aborde le défi de la classification robuste en présence de perturbations adverses au moment de l'inférence, communément appelées attaques adverses. Les attaques adverses sont des modifications imperceptibles des données d'entrée conçues pour induire en erreur les classifieurs entraînés. L'existence de telles attaques a été démontrée dans un large éventail de modèles d'apprentissage automatique, y compris les réseaux de neurones profonds, ce qui a suscité des inquiétudes quant à la sécurité de ces modèles dans des applications critiques.

Au-delà des implications immédiates en matière de sécurité, il existe encore une compréhension limitée des mécanismes fondamentaux qui rendent les réseaux de neurones profonds vulnérables aux attaques adverses et de la manière de concevoir des modèles intrinsèquement robustes à ces menaces. Parmi les solutions proposées, la randomisation est apparue comme une approche prometteuse. Les techniques basées sur la randomisation visent à introduire une stochasticité lors de l'inférence pour atténuer l'efficacité des attaques adverses. Bien que conceptuellement attrayante, l'efficacité pratique de la randomisation en tant que mécanisme de défense reste mal comprise, et son application n'a jusqu'à présent offert qu'une protection limitée contre les attaques adverses.

Dans cette thèse, nous explorons l'efficacité de la randomisation en tant que mécanisme de défense contre les attaques adverses. Notre investigation est structurée autour de trois perspectives clés : celle de l'attaquant, celle du défenseur, et les implications théoriques de la randomisation dans les contextes de classification binaire et multiclasse.

Tout d'abord, nous adoptons la perspective de l'attaquant et analysons les limites des algorithmes d'attaque existants contre une classe spécifique de classifieurs randomisés connus sous le nom de mixtures finies. Nous reformulons le problème d'attaquer ces modèles en un problème d'optimisation combinatoire et proposons un nouvel algorithme d'attaque avec des garanties théoriques intéressantes.

Ensuite, nous adoptons la perspective du défenseur et examinons les avantages potentiels de la randomisation dans le contexte de la classification binaire. Nous démontrons que, bien que la randomisation puisse offrir certains avantages dans des scénarios spécifiques, il est toujours théoriquement possible de construire un classifieur déterministe qui égalera ou surpassera la robustesse de tout classifieur randomisé donné. Nous discutons des implications pratiques de ce résultat pour

les classifieurs randomisés couramment utilisés, tels que les mixtures finies et les classifieurs à injection de bruit.

Enfin, nous étendons notre analyse au cadre de la classification multiclasse, où nous découvrons un contraste frappant par rapport au cas binaire. Contrairement à la classification binaire, où un classifieur déterministe peut toujours atteindre une robustesse comparable ou supérieure, indépendamment de la distribution des données ou du classifieur randomisé, nous montrons que dans le cas multiclasse, il existe des distributions de données pour lesquelles aucun classifieur déterministe ne peut égaler la robustesse du meilleur classifieur randomisé. Nous caractérisons partiellement les distributions de données discrètes à support fini pour lesquelles ce phénomène se produit, et nous construisons une famille de distributions pour lesquelles l'écart de performance entre le meilleur classifieur déterministe et le meilleur classifieur randomisé peut être rendu arbitrairement proche de $\frac{1}{2}$.

Cette thèse se conclura par des questions ouvertes et des perspectives pour les recherches futures dans le domaine de la robustesse adverses et de la randomisation.

Acknowledgement

First and foremost, I would like to thank Teddy Furon and Varun Jog for agreeing to serve as reviewers of my thesis. Their comments and suggestions were invaluable in improving the quality of this work. My sincere thanks also go to Isabelle Bloch for chairing my defense committee, to Amaury for his participation, and to Rafael for being part of the committee as well as for all the support and guidance he provided during the final months of my PhD.

I would like to extend my deepest gratitude to Yann Chevaleyre and Benjamin Negrevergne for their support throughout these years of work. I have learned so much from both of you and have enjoyed this incredibly formative process. Thank you for your patience and trust, for the countless readings and revisions of my manuscript, for all the notes on my defense slides, and for the many shared coffees. Your good humor has been a constant source of motivation.

During this time, I had the privilege of working at the LAMSADE laboratory. I am grateful to all its members for the three and a half years I spent among them. My thanks go to all the faculty and permanent members: to Florian Sikora for his fascinating curious facts and his Paris metro database, and to Dominik for our engaging discussions on diverse topics, from the number of chickens per human required to sustain regular egg consumption to the varying service quality of different urban transit lines. A special thanks to the administrative staff for their constant support—to Olivier for teaching me about Linux, SSH, and other essential tools for a computer science student, and to Marie-Clothilde for being there for us PhD candidates when we needed it.

Naturally, among all the lab members, it was my fellow PhD students with whom I spent the most time and who occupy the largest space in the tapestry of my doctoral journey. I am deeply grateful to Ariane for her empathy and creativity—her surprises, games, and thoughtful gestures left a lasting impression, showing me the profound impact such actions can have on others and in a work environment, as well as for being my bus companion on the way home. To Alexandre, thanks for mentoring me at the start of this journey, for his infectious energy that introduced me to rock climbing, and for the Greek delicacies he shared. The "cousins" Thibault and Bastien

brought enthusiasm and style, while Matteo contributed memorable moments with Chang beers and plenty scooter kilometers in Thailand. Matthieu, Virginia, and Sébastien made climbing sessions a joy, and Theo introduced me to Supersonic. Sarra brightened each day with her joy and warmth, Nicolas with his humor and spicy sauces, and Charles with his kindness and gentleness. Houria's composure and smile, Gil's calm and steadiness, and Ahmad's chivalry were constants I deeply appreciated. Manolis brought good vibes and optimism about Paris, while Erwan, Blaise, Paul, and Florian shared adventures at conferences, weddings, and PSC. Finally, thank you to all the other PhD students for the wonderful times we shared.

Mi familia también merece un agradecimiento especial. A mi mamá por su dulzura y apoyo incondicional, a mi papá por su ejemplo y sus pertinentes palabras. Son unos padres maravillosos a quienes dedico este título: es poco en comparación a lo que merecen por todo lo que han hecho para criarnos a mi hermano y a mí. Gracias a mi hermano por su compañía desde la distancia: los periodos en Oslo y los viajes a México han marcado etapas importantes de mi doctorado. Me siento muy afortunado de tener una familia tan hermosa, y espero que sigamos encontrando la manera de vernos a menudo, por más de que estemos repartidos en el globo. Gracias también a mis abuelos, tios y primos. Verlos en Colombia siempre es un momento especial, incluso más especial desde que estoy lejos. Una mención especial a mi abuelo Nehil, quien desde pequeño inculcó en mí la importancia de la educación: seguro que está muy orgulloso de este logro. Gracias a Susana y a Javier por todo lo que han hecho por mí durante todos estos años, los considero mi familia también y he disfrutado crecer con Andrea junto a ustedes. Gracias a Gabriela y a Juan Pablo por habernos permitido compartir con ustedes como familia: Andrea y yo hemos disfrutado mucho de esos momentos y estaremos siempre ahí para la pequeña Mathilde, quien tiene un lugar especial en nuestros corazones. Gracias a Daniel, Julian, Mateo, Juan José, Cristhian, Huber y Amigo, con quienes crecí y quienes considero siempre mis amigos, a pesar de la distancia. Espero vengan pronto para viajar por ahí.

Gracias Andrea por compartir tu vida y crecer conmigo a lo largo del camino. Gracias por tu apoyo incondicional y por tu amor, que compartes conmigo a diario de maneras grandes y pequeñas. Gracias por todas las caminatas hechas y por hacer. Espero que podamos seguir caminando por muchos años más, y seguir disfrutando de la vida juntos. Ahora que mi doctorado llega a su fin, estaré a tu lado en esta recta final del tuyo.

Contents

1	Introduction	3
1.1	Context and motivations	3
1.2	Problem statement	5
1.2.1	Attacking mixtures of classifiers	6
1.2.2	The benefits of randomization in binary classification	6
1.2.3	Randomization in multi-class classification	7
2	Adversarial machine learning: Background & Related work	9
2.1	Classification	10
2.1.1	Introduction to classification	10
2.1.2	The risk minimization problem	11
2.1.3	Randomized classifiers	13
2.2	Adversarial classification	14
2.2.1	What are adversarial examples?	14
2.2.2	Adversarial risk minimization	16
2.2.3	Overview on existing attack algorithms	17
2.3	Defense mechanisms	20
2.3.1	Adversarial training	21
2.3.2	Certified robustness	22
2.3.3	Diverse ensembles	24
2.3.4	Bayesian neural networks (BNNs)	26
2.4	Randomized defense mechanisms	27
2.4.1	Random transformations of the input	28
2.4.2	Implicit sub-model selection	29
2.4.3	Mixtures of classifiers	30
2.5	Discussion on the importance of strong attack algorithms	35
2.5.1	Attacking randomized defenses.	36
3	Attacking finite mixtures of classifiers	39
3.1	Failures of existing attacks	40
3.1.1	The problems of PGD against finite mixtures	40
3.1.2	ARC attack and its limitations	42
3.1.3	Desirable properties of an attack for mixtures	46
3.2	Lattice Climber attack	51

3.2.1	Binary linear classifiers	52
3.2.2	Multi-class differentiable classifiers	54
3.2.3	Experiments on synthetic data with linear classifiers	55
3.2.4	Experiments on CIFAR-10	56
3.3	Discussion	60
4	Randomization in binary classification	63
4.1	Any randomized classifier can be matched by a deterministic classifier	64
4.1.1	Decomposing a randomized classifier using the layer-cake representation	64
4.1.2	Bounding the adversarial risk of a randomized classifier with its threshold classifiers	65
4.2	Constructing the deterministic classifier that is at least as robust . . .	67
4.2.1	Determinizing finite mixtures	67
4.2.2	Determinizing noise injection classifiers	68
4.3	The expressivity growth required to determinize a randomized classifier	69
5	Randomization in multi-class classification: A graph-theoretic approach	73
5.1	Computing the randomization gap using graph theoretical tools . . .	74
5.1.1	Conflict hypergraph, conflict graph	75
5.1.2	Set packing problem and adversarial risk minimization	77
5.1.3	Examples of computing the randomization gap using the set packing problem	80
5.2	Link between structural properties of the conflict hypergraph and the randomization gap	82
5.2.1	Decomposition of the randomization gap using the clique hypergraph	82
5.2.2	Necessary structures in the support of the distribution for a positive randomization gap	84
5.3	The randomization gap can be arbitrarily close to $1/2$	86
5.4	Discussion	89
6	Conclusion	91
A	Supplementary material for Chapters 3 and 4	95
A.1	Proofs of Chapter 3	95
A.1.1	Proof of Theorem 3.1.4	95
A.1.2	Proof of Lemma 3.2.1	96
A.1.3	Proof of Theorem 3.2.2	96
A.2	Proofs of Chapter 4	97
A.2.1	Proof of Lemma 4.1.2	97
A.2.2	Proof of Theorem 4.1.3	98
A.3	Example of mixtures with strictly better adversarial risk	103

B	Supplementary material for Chapter 5	109
B.1	Reformulation of the adversarial risk minimization problem for discrete distributions with finite support as a set packing problem . . .	109
B.1.1	Proof of Lemma 5.1.3	109
B.1.2	Proof of Corollary 5.1.4	112
B.1.3	Proof of Corollary 5.1.5	112
B.2	Necessary structures for a positive randomization gap	113
B.2.1	Proof of Lemma 5.2.2	113
B.2.2	Proof of Lemma 5.2.3	115
B.2.3	Proof of Lemma 5.2.4	117
B.2.4	Proof of Theorem 5.2.5	117
B.2.5	Proof of Corollary 5.2.7	118
B.2.6	Constructions of specific structures for different ℓ_p norms . .	119
B.3	Constructing discrete distributions with large randomization gap . .	121
B.3.1	Every graph is the conflict graph of a distribution	122
B.3.2	Constructing non-perfect graphs with large randomization gap using the fibration construction	126
B.3.3	Proof of Theorem 5.3.1	129
B.4	On the hardness of computing the randomization gap.	130
B.4.1	Reminder on NP-hardness and co-NP-hardness	131
B.4.2	Hardness of the RG decision problem	132
C	Experimental setup and additional experiments for Chapter 3	135
C.1	Synthetic experiments	135
C.1.1	ARC and EOL-PGD against two classifiers in the plane at the same distance from the origin	135
C.1.2	LCA, EOL-PGD and ARC with different number of classifiers with respect to the distribution of the bias of the linear classifiers	136
C.2	CIFAR-10 experiments	138
C.2.1	Main results	138
C.2.2	Comparing strong versions of the attacks	139
C.3	On adversarial training for mixtures of classifiers	141
D	Résumé en français	143
D.1	Contexte et motivations	143
D.2	Énoncé du problème	145
D.2.1	Attaquer des mélanges de classifieurs	145
D.2.2	Intérêt de la randomisation en classification binaire	146
D.2.3	Randomisation en classification multi-classe	146
D.3	Apprentissage automatique sous attaques adverses : Contexte et travaux connexes	146
D.3.1	Algorithmes d'attaque	151

D.3.2 Mécanismes de défense	155
D.4 Contributions de la thèse	160
D.4.1 Chapitre 3: Attaque des mélanges finis de classifieurs	160
D.4.2 Chapitre 4: Étude théorique du rôle de la randomisation (cas binaire)	160
D.4.3 Chapitre 5: Extension au cas multiclasse	161
Bibliography	163

Notation, Symbols & Abbreviations

The following is a non-exhaustive list of symbols and abbreviations used in this manuscript.

Mathematical symbols

$\mathbb{R}$	Set of real numbers.
$\mathbb{N}$	Set of natural numbers.
$\mathbb{R}^d$	Set of real valued d -dimensional vectors.
$[K]$	Set of integers between 1 and K .
$x^{(i)}$	i -th component of the vector x .
e_i	Canonic vector that has all zero components, except the i -th which equals 1.
$\mathbf{1}_d$	Vector of size d with all components equal to one.
$\mathbf{0}_d$	Vector of size d with all components equal to zero.
$ x $	Absolute value of the scalar x .
$\ x\ _p$	ℓ_p -norm of the vector x .
$\ x\ _\infty$	Infinite norm of the vector x .
$\ x\ $	Euclidean norm (ℓ_2) of the vector x .
I_d	Identity matrix of size $d \times d$.
$\mathbf{1}_{d \times d}$	Matrix of size $d \times d$ with all entries equal to 1.
$\mathbf{0}_{d \times d}$	Null matrix of size $d \times d$.
$B_p(x, \epsilon)$	Closed ball of center x and radius ϵ under the ℓ_p norm.
$\mathbb{E}_\rho[\cdot]$	Expected value under the distribution ρ .
$\mathbf{1}_{\{C\}}$	Indicator function of the event C .

Abbreviation

e.g.	<i>exempli gratia</i>
Eq.	Equation
Fig.	Figure
i.e.	<i>id est</i>
s.t.	such that
w.r.t.	with respect to
a.k.a.	also known as
RHS	Right Hand Side
LHS	Left Hand Side

Introduction

1.1 Context and motivations

The journey of artificial intelligence (AI) and machine learning (ML) has seen some tremendous advances in the last decades, in particular since the wide spread of deep learning [LBH15], which has found applications in many important fields such as medicine [Lev+19; Pic+21], agriculture [KPB18], pharmaceuticals [Che+18] and transportation [Gri+20], to name a few. This rapid development is also fueled by the incredible amount of data that is being generated every day, and the increasing computational power that is available to process it, allowing for new and more complex tasks to be tackled with AI [YSD23].

This major technological progress has also brought new challenges, and one of the most important ones is the robustness of these models to adversarial perturbations [LJ23]. For example, poisoning attacks occur when an adversary manipulates the training data to degrade the model's performance [BNL12]. Other types include model inversion [FJR15] and membership inference attacks [Sho+17], where the goal is to extract sensitive information about the training data. Lastly, *evasion attacks* involve introducing carefully crafted perturbations to the input data at inference time, aiming to mislead the model into making incorrect predictions [Big+13; Sze+14]. The latter have intrigued the scientific community, and has led to the development of a new field of adversarial attacks research, which by the beginning of 2025 had almost 12.000 academics papers on arxiv¹.

Adversarial attacks. An adversarial attack is a small, imperceptible perturbation of an input that induces a model into error, even when the model succeeded on the original unmodified input [Big+13; Sze+14]. Since their (re)introduction in the seminal work of Szegedy et al. [Sze+14] during the explosion of deep learning, they have been at the center of a paradox concerning deep neural networks: these models are highly expressive and achieve state of the art performance on complex tasks such as speech and visual recognition, but they can be induced into error by perturbations on the (otherwise well classified) inputs that are *imperceptible*

¹See the following this blog by Nicholas Carlini for an approximate count and listing of the arxiv papers on adversarial attacks: <https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html>

to humans. Adversarial attacks have been designed in many different domains, including image classification [Sze+14], speech recognition² [CW18], and natural language processing [Ebr+18], making them a real concern for the deployment of AI systems in the real world.



Fig. 1.1.: Adversarial examples for the AlexNet model [KSH12] on the ImageNet [Den+09] dataset. The left column of each group presents the original images and the right column shows the adversarial examples crafted by Szegedy et al. [Sze+14]. All the adversarial examples are labeled as “ostrich, *Struthiocamelus*” by the attacked classifier. The middle column presents the difference between the clean and perturbed images. Picture taken from [Sze+14, Section 4.2].

In the last decade, there has been a back-and-forth in research progress between developing increasingly potent attacks [CW17a; CH20] and the creation of practical defense mechanisms through empirical design [Mad+18]. This cat-and-mouse game has led to the development of a wide range of defenses and attacks, each adapting to the other in an iterative cycle, often without yielding a clear understanding of the underlying problem. To gain deeper insights into adversarial attacks, a substantial body of research has approached the problem from a *game theoretical* perspective, modeling the interaction as a two-player game between a learner and an adversary that perturbs inputs at test-time.

A game theoretical perspective on adversarial attacks. The problem of learning robust models against adversarial attacks can be naturally cast as a game between a *learner* that wants to correctly classify inputs, and an *adversary* that can slightly modify inputs at test time to induce the classifier into error [PJ21]. This point of view has led to important theoretical results regarding the existence (or not) of optimal classifiers in the presence of an adversary [TJK24], a better understanding of the behavior of an optimal attacker [TJK23] and even motivated the development of empirical defenses [Pin+20; Meu+21]. One insight of particular interest to this thesis is related to the well-known fact in game theory that some games have no *pure Nash equilibrium*, but that it is usually the case that extending to *mixed strategies* is enough to guarantee the existence of a *mixed Nash equilibrium* [Os+04].

²See the following blog entry for more information and real audio adversarial examples: https://nicholas.carlini.com/code/audio_adversarial_examples/

The concept of mixed strategies in game theory can be transposed to the context of classifiers, which naturally gives rise to *randomized classifiers*. When there is a fixed family of deterministic classifiers of interest, for example linear classifiers of a certain dimension, or neural networks with some fixed architecture, these can be viewed as the set of pure strategies. Then, a new classifier called *mixture* can be constructed: for any input, the mixture classifier randomly selects one of these deterministic classifiers, and outputs the prediction according to it. This provides a theoretically grounded and simple way to introduce randomization in the classification process [Pin+19; Meu+21].

Randomized models in practice. The idea of using randomization to improve adversarial robustness in machine learning and deep learning is not new, and randomized classifiers have been widely used to build defenses using a plethora of techniques. The most common ones are based on random transformations of the input, and they were the first to appear in the literature [Xie+18; HRF19]. Another set of approaches is based in model sub-sampling, usually done by performing activation pruning in an independent fashion [Dhi+18], similar to *dropout* [Sri+14], or with more complex activation functions that promote competition between groups of neurons in a neural network [Pan+21; PCT21].

Intuitively, the added flexibility of randomized classifiers, stemming from their ability to introduce stochasticity, suggests that they should outperform their deterministic counterparts in adversarially robust classification. However, the extent of the improvements brought by randomization remains unclear, primarily because evaluating randomized models is inherently challenging³. In practice, randomized models have often failed to demonstrate robustness against adaptive attacks that explicitly account for the introduced randomization, rendering such defenses ineffective [ACW18; Tra+20]. This suggests that the proposed randomized models are simply harder to attack with traditional attacks, instead of being actually more robust in the strong sense of the term. There is a need for a better understanding of the true benefits of randomization for adversarial robustness.

1.2 Problem statement

In this thesis we analyze the robustness of randomized classifiers from a theoretical perspective, and the improvements they can bring over deterministic classifiers. More specifically, we want to understand to what extent, and under which condi-

³The Robustbench [Cro+21] ranking refuses to accept any type of randomized model. In their own words, “we accept only defenses that are (1) [...] (2) have a fully deterministic forward pass (i.e. no randomness) that (3) [...]. Often, defenses that violate these 3 principles only make gradient-based attacks harder but do not substantially improve robustness”

tions randomized classifiers improve adversarial robustness in classification tasks compared to deterministic classifiers. To answer this question, we will first take the perspective of the attacker against finite mixtures of classifiers, which are arguably one of the simplest type of randomized classifiers. Afterwards, we will take the defenders' point of view to understand the benefits of randomization in the case of binary classification. Lastly, we will generalize to multi-class classification.

1.2.1 Attacking mixtures of classifiers

Taking into account that the evaluation of randomized classifiers is not straightforward, and based on recent work that discovered flaws in the evaluation of finite mixtures of classifiers [DS22], we aim to answer the following question:

Question 1: *How to successfully attack mixtures of classifiers ?*

In Chapter 3 we will provide a geometrical analysis of the problem of attacking a finite mixture, which will allow us to formulate it as the problem of enumerating the maximal elements of a lattice [DP02]. This will provide a general understanding of the complexity of attacking a mixture of classifiers, and position existing attacks with respect to the ideal, optimal attack. We will also derive an attack algorithm that has better theoretical guarantees than existing attacks, and that outperforms the state-of-the-art in practice.

1.2.2 The benefits of randomization in binary classification

In a second time, and motivated by explicit examples that show that randomized classifiers can provide a strict performance improvement over deterministic classifiers, we are going to tackle the following question:

Question 2: *Under which conditions does randomization improve robustness to adversarial attacks in the binary classification setting ?*

In Chapter 4 we will show that any randomization scheme can be outperformed by some deterministic classifier, which suggests that in the binary classification setting, the benefits of randomization are limited. We will show how to build such deterministic classifiers for some commonly used randomization schemes.

1.2.3 Randomization in multi-class classification

A natural following step is to understand the benefits of randomization in the multi-class setting. We aim to answer the following question:

Question 3: *Under which conditions does randomization improve robustness to adversarial attacks in the multi-class classification setting ?*

In stark contrast with the binary case, we will show in Chapter 5 that for some data distributions, the best deterministic classifier can be outperformed by a randomized classifier by a large margin. This shows that, in theory, randomization is necessary for adversarial robustness in multi-class classification. We will provide a partial characterization of the data distributions for which randomization is strictly necessary. Another important difference with the binary case is the set of theoretical tools that we will use to answer this question, which will include graph theory and combinatorial optimization.

Adversarial machine learning: Background & Related work

Contents

2.1	Classification	10
2.1.1	Introduction to classification	10
2.1.2	The risk minimization problem	11
2.1.3	Randomized classifiers	13
2.2	Adversarial classification	14
2.2.1	What are adversarial examples?	14
2.2.2	Adversarial risk minimization	16
2.2.3	Overview on existing attack algorithms	17
2.3	Defense mechanisms	20
2.3.1	Adversarial training	21
2.3.2	Certified robustness	22
2.3.3	Diverse ensembles	24
2.3.4	Bayesian neural networks (BNNs)	26
2.4	Randomized defense mechanisms	27
2.4.1	Random transformations of the input	28
2.4.2	Implicit sub-model selection	29
2.4.3	Mixtures of classifiers	30
2.5	Discussion on the importance of strong attack algorithms	35
2.5.1	Attacking randomized defenses.	36

Adversarial machine learning addresses the challenge of solving usual machine learning tasks, such as classification, in the presence of a malicious adversary capable of manipulating data or models at various stages of training and/or inference. For instance, data poisoning involves subtly altering the training data to create a backdoor, which can later be exploited to manipulate the model into predicting a specific outcome [BNL12]. In contrast, *evasion attacks* focus on imperceptibly modifying inputs during inference to alter a classifier's predictions [Big+13].

While numerous types of adversarial attack exist, this thesis concentrates on understanding evasion attacks and developing new defense mechanisms against them using randomization. The first section of this chapter provides a brief overview of

classification fundamentals, followed by a definition of adversarial attacks in this context and an introduction to algorithms for generating such attacks. The next section reviews existing defense mechanisms, and the final section highlights recent advancements in randomized defense strategies, which form the core focus of this thesis.

2.1 Classification

In this section, we define the problem of classification, which is the task of assigning labels to inputs that belong to a certain class. We will define two important types of classifiers that are of major importance for this thesis, namely *deterministic* and *randomized* classifiers. We will also define the *risk* of a classifier and the problem of *risk minimization* in the context of classification. This will provide the base for the understanding of adversarial classification, which will be introduced in the following section.

2.1.1 Introduction to classification

A classification task involves the following components:

- An input space $\mathcal{X} \subseteq \mathbb{R}^n$ for some $n \in \mathbb{N}$. Typically, a vector $x \in \mathcal{X}$ is used to represent an image, a text or an audio sample. We consider an ℓ_p -norm on $\mathcal{X}$, typically the ℓ_2 (Euclidean), or ℓ_∞ (supremum) norm, which induces the usual topology on $\mathcal{X}$. This norm also endows $\mathcal{X}$ with the structure of a measurable space with Borel σ -algebra $\mathcal{B}(\mathcal{X})$ generated by the open sets.
- A dataset $S = \{(x_1, y_1), \dots, (x_N, y_N)\}$, where each input $x_i \in \mathcal{X}$ is associated with a label $y_i \in \mathcal{Y}$ and $\mathcal{Y}$ is a finite set. For simplicity and without loss of generality, we assume that $\mathcal{Y} = \{1, \dots, K\}$ for some $K \in \mathbb{N}$.
- We assume that there exists an unknown data distribution ρ over $\mathcal{X} \times \mathcal{Y}$, from which the dataset S was drawn.

The goal of classification is to find a function $h : \mathcal{X} \rightarrow \mathcal{Y}$ that assigns a label to each input x in $\mathcal{X}$. It is common practice to define classifiers in a two-step process: first, a

score function, a.k.a *logit function* $f : \mathcal{X} \rightarrow \mathbb{R}^K$ is defined, from which the classifier h is derived by taking the argmax^1 of the per-label scores $f(x)^{(k)}$, as follows:

$$h(x) = \operatorname{argmax}_{k \in \mathcal{Y}} f(x)^{(k)}. \quad (2.1)$$

We sometimes abuse notation and refer to the score function as the classifier, in which case we mean the classifier that is derived from the score function using the argmax operator.

These classifiers are *deterministic* since they always assign the same label to each input x . We use $\mathcal{H}_{\text{det}}$ to represent the set of all deterministic classifiers, that is, all Borel measurable functions $h : \mathcal{X} \rightarrow \mathcal{Y}$. Figure 2.1 shows a representation of a classification task over a finite set of input points, in which the goal is to separate the blue squares from the red circles. In this example, the green line represents the decision boundary of some classifier h , that separates the two classes perfectly.

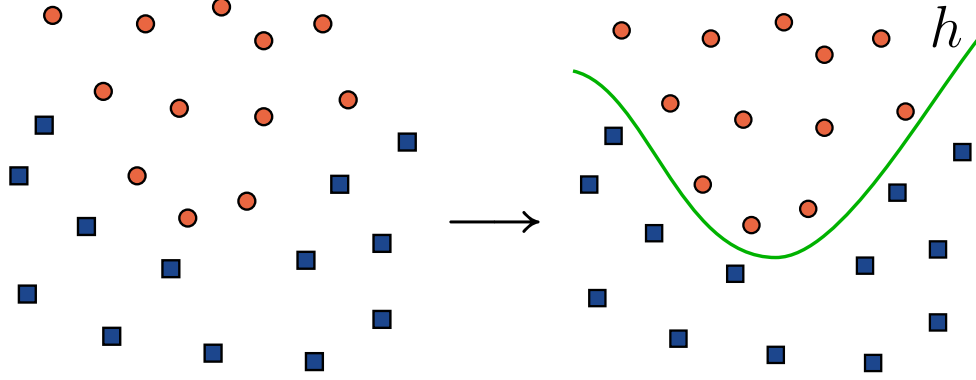


Fig. 2.1.: Representation of a classification task in which the goal is to separate the blue squares from the red circles. In this example, the green line represents the decision boundary of the classifier h .

2.1.2 The risk minimization problem

To find a classifier that correctly predicts the labels of the inputs, the first step is to define a measure of the quality of a classifier. This is done through a measurable *loss function* $\mathcal{L} : \mathbb{R}^K \times \mathcal{Y} \rightarrow [0, \infty)$ that penalizes wrong predictions. In classification we aim to minimize misclassifications, and thus we typically employ the *zero-one loss*, defined for a classifier $f : \mathcal{X} \rightarrow \mathbb{R}^K$ and a point (x, y) as follows

$$\mathcal{L}^{0-1}(f(x), y) := \mathbb{1} \left\{ \operatorname{argmax}_{k \in \mathcal{Y}} f(x)^{(k)} \neq y \right\}. \quad (2.2)$$

¹The argmax operator might output a set. For simplicity, we suppose that there is a unique maximum.

Given a classification task defined by an unknown data distribution ρ over $\mathcal{X} \times \mathcal{Y}$ and a loss function $\mathcal{L}$, we measure the quality of a classifier using the *risk*:

$$\mathcal{R}^{\mathcal{L}}(h, \rho) := \mathbb{E}_{(x,y) \sim \rho} [\mathcal{L}(h(x), y)]. \quad (2.3)$$

Note that when the loss function is the zero-one loss, the risk is exactly the misclassification error, in which case we omit the superscript and simply write $\mathcal{R}(h, \rho)$.

The goal of the learner is to find the classifier that minimizes the risk in (2.3) by defining a set of functions $\mathcal{H} \subseteq \mathcal{H}_{\text{det}}$, called the *hypothesis class*, and solving the *risk minimization* problem over $\mathcal{H}$ with loss $\mathcal{L}$, which is defined as follows:

$$\mathcal{R}^{\mathcal{L}}(\mathcal{H}, \rho) := \inf_{h \in \mathcal{H}} \mathcal{R}^{\mathcal{L}}(h, \rho). \quad (2.4)$$

In practice, the data distribution ρ is unknown and the learner only has access to a finite dataset S of N i.i.d. points drawn from ρ . Thus, the risk is approximated by the *empirical risk*:

$$\hat{\mathcal{R}}^{\mathcal{L}}(h, S) := \frac{1}{N} \sum_{i=1}^N \mathcal{L}(h(x_i), y_i), \quad (2.5)$$

which is minimized as a surrogate to minimizing the true risk.

The choice of the hypothesis class $\mathcal{H}$ is crucial for the success of the learner. It should be rich enough to contain classifiers that can approximate the true data distribution, but it must not be excessively large to prevent *overfitting*, a phenomenon where the classifier performs well on the training data but poorly on new data regardless of the amount of training data available. To illustrate this, consider a binary classification task with data distribution ρ , a dataset $S = \{(x_i, y_i)\}_{i \in [N]}$ and the following classifiers:

$$h_k(x) = \begin{cases} y_i & \text{if } \exists x_i \in S \text{ s.t. } x = x_i, \\ k & \text{otherwise,} \end{cases} \quad (2.6)$$

where $k \in \{0, 1\}$. Both h_1 and h_2 belong to $\mathcal{H}_{\text{det}}$, the class of measurable classifiers. These classifiers achieve zero empirical risk on S , but at least one of them will have a true risk of at least $1/2$ under the reasonable assumption that the dataset S is a negligible set with respect to the data distribution ρ .

The choice of the loss function $\mathcal{L}$ is also very important. The zero-one loss corresponds exactly to misclassifications and is commonly used for theoretical analysis and evaluation. However it is generally difficult to optimize, in particular because its gradients are zero whenever they exist. For this reason, *surrogate* loss functions are used in practice. These are differentiable approximations of the zero-one loss that

enable efficient optimization using gradient-based methods. A widely used surrogate loss is the cross-entropy loss, defined for a score classifier f as follows:

$$\mathcal{L}^{\text{CE}}(f(x), y) = -\log \left(\frac{\exp(f(x)^{(y)})}{\sum_{j \in [K]} \exp(f(x)^{(j)})} \right). \quad (2.7)$$

Note that if the loss function $\mathcal{L}$ is different from the zero-one loss, there is no direct interpretation of the risk in terms of the error of the classifier. However, the risk is still a measure of the quality of the classifier, and the goal of the learner is to find the one that minimizes it. This leads to a key property that is desirable for a surrogate loss, which is *consistency*. A surrogate loss $\mathcal{L}$ is said to be *classification-consistent* if every minimizing sequence of classifiers for the surrogate risk is also a minimizing sequence for the zero-one risk [BJM06]. This property is crucial as it ensures that minimizing the surrogate risk $\mathcal{R}^{\mathcal{L}}(\cdot, \rho)$ also minimizes the misclassification error $\mathcal{R}(\cdot, \rho)$.

2.1.3 Randomized classifiers

In this thesis, we also consider *randomized classifiers*, which map each input x in $\mathcal{X}$ to a categorical probability distribution over the labels in $\mathcal{Y}$. The set of probability distributions over $\mathcal{Y}$ can be represented with the standard K -simplex defined as

$$\Delta^K = \{u \in \mathbb{R}^K \mid u \geq 0, \sum_{k=1}^K u^{(k)} = 1\},$$

where $u^{(k)}$ represents the k -th component of vector u , and a vector $u \in \Delta^K$ represents the probability distribution that assigns probability $u^{(k)}$ to the event of sampling label k . A randomized classifier is therefore a measurable function $h : \mathcal{X} \rightarrow \Delta^K$, and we denote $\mathcal{H}_{\text{rand}}$ the set of all randomized classifiers².

Randomized classifiers produce randomized predictions: given an input x , a label $\tilde{y}$ is drawn from the discrete distribution of parameter $h(x)$. Formally,

$$\tilde{y} \sim \text{Categorical}(h(x)). \quad (2.8)$$

²Note that deterministic classifiers can be seen as a special case of randomized classifiers that assign probability 1 to a single label. That is, one can turn a deterministic classifier h into an equivalent randomized classifier h' that only outputs vectors from the canonical basis $\{e_1, \dots, e_K\} \subset \mathbb{R}^K$. From h , define h' as follows: $h'(x) = e_{h(x)}$.

In an abuse of notation, we will simply write $\tilde{y} \sim h(x)$. To compute the error of a randomized classifier h at (x, y) , we use the *expected zero-one loss*, where the expectation is taken w.r.t. the randomness of $h(x)$:

$$\mathcal{L}^{0-1}(h(x), y) = \mathbb{E}_{\tilde{y} \sim h(x)} [\mathbb{1} \{\tilde{y} \neq y\}] = 1 - h(x)^{(y)} \quad (2.9)$$

We use the same symbol for both the classical and expected zero-one loss because the latter is a natural extension of the former to the randomized setting. With the definition of the zero-one loss for randomized classifiers, the risk of a randomized classifier is also defined using (2.3).

2.2 Adversarial classification

In this section, we generalize the problem of classification to the adversarial setting of *evasion attacks*, in which an adversary can manipulate the inputs at inference to deceive the classifier. We will introduce the concept of *adversarial examples* and go over different types of algorithms that have been developed to find them. We will go over the state-of-the-art in attack algorithms and defense mechanisms, and give particular attention to randomized defenses, which are central to this thesis. We will end the section with a discussion on the importance of using strong attacks to evaluate empirical robustness.

2.2.1 What are adversarial examples?

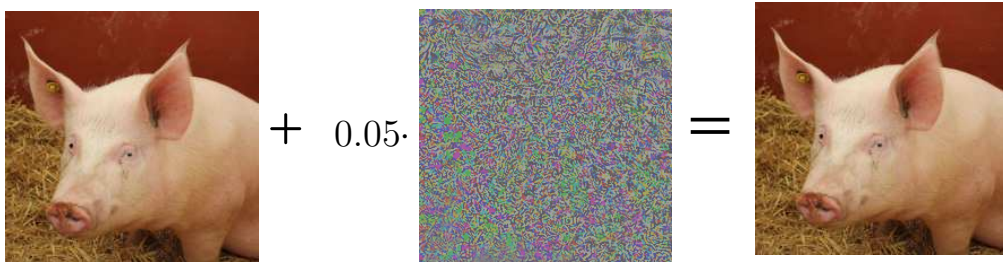


Fig. 2.2.: Adversarial perturbation of a hog from ImageNet [Den+09] for the pretrained ResNet50 from torchvision. The left image is correctly predicted as “hog” with high confidence, while the right image is misclassified as “airliner” with high confidence. The perturbation is imperceptible to the human eye. See <https://adversarial-ml-tutorial.org/introduction/> for more details and the code that generated this adversarial example. The perturbation has been scaled by a factor of 20 for visualization purposes, because otherwise it would appear as a grey square to the naked eye.

Adversarial examples are small perturbations added to the input that cause a classifier to make a wrong prediction with high confidence. Figure 2.2 depicts the process of crafting an adversarial example for a deep neural network. The image on the

left shows a hog from the ImageNet [Den+09] dataset, which is correctly classified by a deep neural network as a “hog” with high confidence. The image on the right shows the same pig image after adding a small perturbation (center), which causes the network to misclassify the image as an airliner with high confidence. The perturbation is imperceptible to the human eye, but it is enough to fool the network into making a wrong prediction. Moreover, a similar perturbation can be crafted for many other target labels.

To formally define adversarial examples, we must begin by defining the set of valid adversarial perturbations of an input x , also referred to as *adversarial neighborhood* of x . A common choice is the closed ℓ_p -ball of radius ϵ around x , denoted $B_p(x, \epsilon)$, where ϵ represents the adversary’s budget. The parameter ϵ is chosen to be small enough such that all points within the ℓ_p -ball are considered imperceptibly different from x , and common choices for the p -norm are the ℓ_∞ and ℓ_2 norms. In this work, we will focus on the closed ℓ_p -ball of radius ϵ around x for some $p \in (1, \infty]$.

For any given loss $\mathcal{L}$, an *optimal attacker* receives a data point (x, y) and finds $x' \in B_p(x, \epsilon)$ that maximizes the loss of the classifier. The loss under attack of a classifier h at (x, y) writes as follows:

$$\mathcal{L}_{\epsilon,p}(h(x), y) := \sup_{x' \in B_p(x, \epsilon)} \mathcal{L}(h(x'), y) \quad (2.10)$$

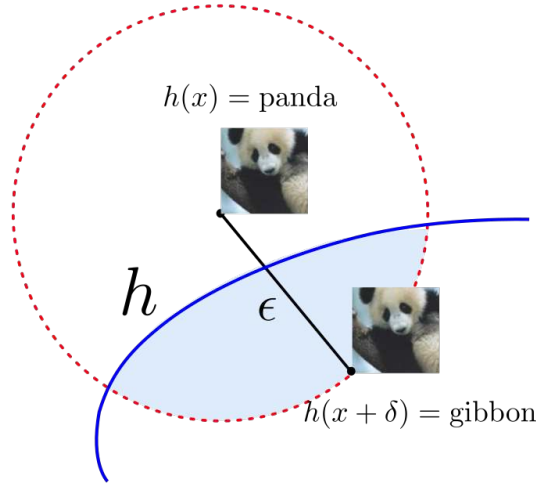


Fig. 2.3.: Illustration of the original image x from the ImageNet dataset [Den+09] and its adversarial neighborhood $B_2(x, \epsilon)$. This time, the perturbed panda is classified as a gibbon. See [GSS15] for more details.

Research on adversarial machine learning, particularly on test time attackers that can temper the inputs of a classifier, can be traced back as early as 2004, when Dalvi et al. [Dal+04] introduced the concept of adversarial classification in the context

of spam detection. Later, in 2006, Globerson and Roweis [GR06] considered the setting in which the adversary can delete some features of the input. It was in 2013 when Biggio et al. [Big+13] introduced the term *evasion attacks* to refer to the problem of manipulating the inputs to deceive a classifier at inference, and proposed a gradient-descent evasion attack algorithm.

Adversarial examples for deep learning models were popularized by Szegedy et al. [Sze+14], who showed that despite achieving state-of-the-art performance in tasks like image classification, these models could consistently misclassify images that had been subtly and imperceptibly altered from their original, correctly classified versions. Notably, this discovery came just a year and a half after Krizhevsky et al. [KSH12] revolutionized image classification with AlexNet, which surpassed the previous state-of-the-art on ImageNet [Den+09] by a remarkable 10% margin in the 2012 competition. In their seminal work, Szegedy et al. [Sze+14] were able to craft adversarial examples for AlexNet, as shown in Figure 1.1.

2.2.2 Adversarial risk minimization

In this thesis, we consider adversarial classification against an optimal adversary, as described in (2.10). For a given loss function $\mathcal{L}$, a p -norm and a budget $\epsilon > 0$, the *adversarial risk* of a classifier h w.r.t. the data distribution ρ against an optimal data-perturbing adversary is defined as:

$$\mathcal{R}_{\epsilon,p}^{\mathcal{L}}(h, \rho) := \mathbb{E}_{(x,y) \sim \bar{\rho}} \left[\sup_{x' \in B_p(x, \epsilon)} \mathcal{L}(h(x'), y) \right]^3. \quad (2.11)$$

The *adversarial risk minimization problem* over hypothesis class $\mathcal{H}$ at level ϵ and norm p writes as follows:

$$\mathcal{R}_{\epsilon,p}^{\mathcal{L}}(\mathcal{H}, \rho) := \inf_{h \in \mathcal{H}} \mathcal{R}_{\epsilon,p}^{\mathcal{L}}(h, \rho) \quad (2.12)$$

When the loss function is the zero-one loss, we will omit it and write $\mathcal{R}_{\epsilon,p}$. The *empirical adversarial risk* over a finite set of data points S is defined analogously to the empirical risk in (2.5):

$$\hat{\mathcal{R}}_{\epsilon,p}^{\mathcal{L}}(h, S) := \frac{1}{N} \sum_{i=1}^N \sup_{x' \in B_p(x_i, \epsilon)} \mathcal{L}(h(x'), y_i) \quad (2.13)$$

³Note that the expectation is taken w.r.t. $\bar{\rho}$, which is the completion of the Borel measure ρ restricted to the universal σ -algebra $\mathcal{U}(\mathcal{X} \times \mathcal{Y})$. This technical detail is important because the loss function under attack might not be a Borel measurable function, even when h is [AFM21; PJ23]. In particular, this is the case with the zero-one loss. All these technical details are discussed in Appendix A.2.2, as they are required to prove Theorem 4.1.3 in Chapter 4.

Coming back to the illustration in Figure 2.1, where the task was to separate the blue squares from the red circles, we can visualize in Figure 2.4 the effect of considering an adversarial neighborhood around the original points. The adversarial neighborhood is represented as the small balls around the original points, and the goal of the classifier is to correctly classify the points even when they are slightly perturbed. The classifier h that was able to separate all the data points is now vulnerable to attacks on all those that are close to the decision boundary, as they can be easily perturbed to be misclassified by it. More generally, when some points of different classes are very close to each other and their adversarial neighborhoods overlap, it becomes impossible to separate them under adversarial attacks.

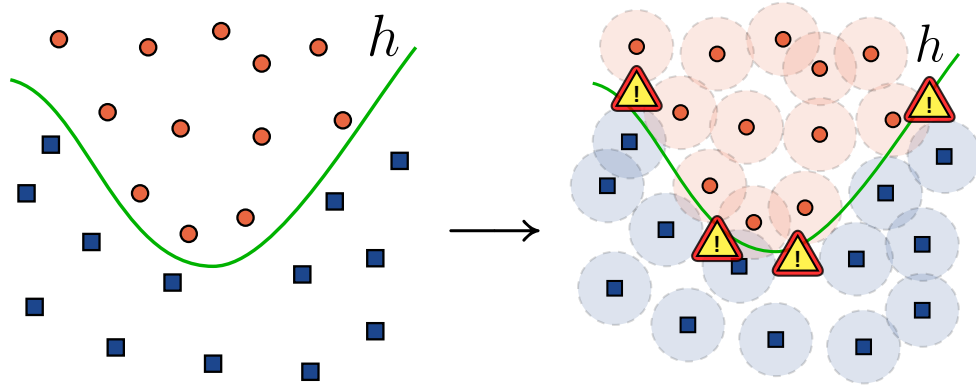


Fig. 2.4.: Representation of an adversarial classification task in which the goal is to separate the blue squares from the red circles while being robust to adversarial attacks.

2.2.3 Overview on existing attack algorithms

The problem of finding an adversarial example at inference can be formulated as the problem of finding

$$x_{\text{adv}} \in \underset{x' \in B_p(x, \epsilon)}{\operatorname{argmax}} \mathcal{L}(h(x'), y), \quad (2.14)$$

for a given point (x, y) , classifier h and loss function $\mathcal{L}$. An attack algorithm is a method that attempts to solve (2.14) under a given set of assumptions. These assumptions include the adversary's capabilities and their knowledge of the target classifier, which define the *threat model* and influences the attack design. *White-box* attacks assume complete knowledge of the target model, including its architecture, parameters, and gradients, enabling the design of highly effective attacks. *Black-box* attacks, by contrast, rely solely on querying the target model to infer information, often approximating gradients using techniques like zeroth-order methods. It is also worth distinguishing between *non-targeted* attacks, which aim to cause misclassification regardless of the new predicted class, and *targeted* attacks, which aim to induce a misclassification toward a specific target class.

The process of crafting attacks involves various optimization techniques. There are two main approaches: to approximately solve (2.10) using a surrogate loss and projected gradient descent, or to use a Lagrangian relaxation of the problem to find a minimal norm perturbation that forces a misclassification. In the following, we focus on attack algorithms within the white-box threat model.

Initial approach by Szegedy et al. [Sze+14]. For a classifier h based on the logit function f , an input point $(x, y) \in \mathbb{R}^d \times [K]$ and a target label $y_{\text{adv}} \in [K]$, Szegedy et al. [Sze+14] tried to solve the following ℓ_2 targeted attack problem in their original work:

$$\begin{aligned} \min \quad & \|\delta\|_2 \\ \text{s.t.} \quad & h(x + \delta) = y_{\text{adv}}, \\ & x + \delta \in \mathcal{X}. \end{aligned} \tag{2.15}$$

However, given the difficulty of solving (2.15), authors approximate it by using box-constrained L-BFGS. More specifically, authors consider a continuous loss function $\mathcal{L}$ (for example the cross-entropy) and propose the following ℓ_1 attack problem:

$$\begin{aligned} \min \quad & c\|\delta\|_1 + \mathcal{L}(f(x + \delta), y_{\text{adv}}) \\ \text{s.t.} \quad & x + \delta \in \mathcal{X}. \end{aligned} \tag{2.16}$$

They perform line-search to find the minimum c for which the solution to (2.16) satisfies $h(x + \delta) = y_{\text{adv}}$. A large value of c forces the perturbation to have small norm, in which case it might be hard to change the prediction of the classifier. A small value of c allows for a larger perturbation, which favors the change of the prediction at the risk of making the perturbation perceptible. In practice, the perturbations found by the authors were always imperceptible to the human eye.

FGM. The *fast gradient method* (FGM) is a simple and efficient white-box attack based on the assumption that neural networks are locally linear around their inputs. If we consider a surrogate loss function $\mathcal{L}$ and its first order approximation around an input x , we obtain that for a small perturbation δ ,

$$\mathcal{L}(f(x + \delta), y) \approx \mathcal{L}(f(x), y) + \nabla_x \mathcal{L}(f(x), y)^T \delta. \tag{2.17}$$

We can look for the perturbation that maximizes this first order approximation constrained to having a norm of at most ϵ by considering the following optimization problem:

$$\begin{aligned} \max \quad & \delta^T \nabla_x \mathcal{L}(f(x), y) \\ \text{s.t.} \quad & \|\delta\|_p \leq \epsilon. \end{aligned} \tag{2.18}$$

In the case of the ℓ_2 -norm, the solution to (2.18) is given by the well-known maximizer of the dot product:

$$\delta_{\text{FGM}} = \epsilon \cdot \frac{\nabla_x \mathcal{L}(f(x), y)}{\|\nabla_x \mathcal{L}(f(x), y)\|_2}. \quad (2.19)$$

In the case of the ℓ_∞ -norm, which was the problem considered by Goodfellow et al. [GSS15], there is a closed form solution: the famous *fast gradient sign method* (FGSM) attack:

$$\delta_{\text{FGSM}} = \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(f(x), y)). \quad (2.20)$$

Note that if the classifier is linear, then the following holds in the ℓ_∞ norm: if the classifier is vulnerable, then the sign of the gradient is guaranteed to be an adversarial example. This is not the case with the gradient itself. Figure 2.5 presents a visualization of the link between the gradient and the sign of the gradient in the ℓ_∞ norm for a linear classifier f .

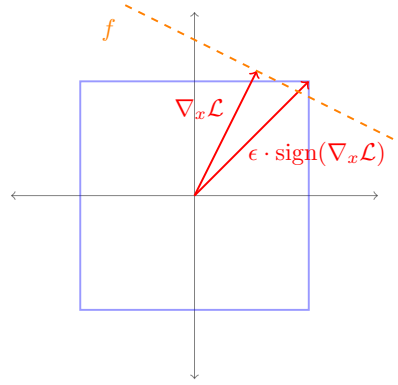


Fig. 2.5.: Illustration of the FGSM attack. The perturbation is computed as the sign of the gradient of the loss function with respect to the input, scaled by the adversarial budget ϵ . Note that following the gradient of the loss to create an adversarial example will produce one that is outside of the ℓ_∞ -ball of radius ϵ around the original point x , while following the sign of the gradient will produce a valid perturbation that inside the ℓ_∞ -ball.

PGD. The *projected gradient descent* (PGD) attack is an extension of the FGM that considers multiple steps of gradient ascent. The PGD attack is defined recursively as follows:

$$x_{t+1} = \Pi_{B_p(x, \epsilon)}(x_t + \alpha \cdot \nabla_x \mathcal{L}(f(x_t), y)), \quad (2.21)$$

where $\Pi_{B_p(x, \epsilon)}$ denotes the projection into the set of admissible adversarial perturbations $B_p(x, \epsilon)$, and α is a parameter of the attack that controls the step size of each iteration. Note that the linearity hypothesis that was used to derive the FGSM attack is less critical for the PGD attack. It is worth noting that the name PGD was employed by Madry et al. [Mad+18], while at a similar time Kurakin et al. [KGB17]

proposed an almost identical attack that they called the *basic iterative method* (BIM). Note that the PGD attack can also be used in the ℓ_∞ norm, in which case it is worth replacing the gradient in (2.21) with the sign of the gradient, to benefit from the closed form solution described in (2.20).

Carlini-Wagner. The *Carlini-Wagner* (CW) attack [CW17b] revisits the initial minimal-norm formulation by Szegedy et al. [Sze+14] and focuses on the choice of loss function while also generalizing to different ℓ_p norms. Consider a classifier h based on the score function f . Carlini and Wagner [CW17b] consider the following optimization problem:

$$\begin{aligned} \min \quad & \|\delta\|_p + c \cdot g(x + \delta) \\ \text{s.t.} \quad & x + \delta \in [0, 1]^d. \end{aligned} \quad (2.22)$$

where g is a function such that $g(x + \delta) \leq 0$ if and only if $h(x + \delta) = y_{\text{adv}}$ for a target label y_{adv} , for example

$$g(z) = \max \left(\max_{k \neq y_{\text{adv}}} f(z)^{(k)} - f(z)^{(y_{\text{adv}})}, 0 \right). \quad (2.23)$$

After testing different ameliorations, authors propose the following unconstrained optimization problem for the ℓ_2 norm:

$$\min_w \quad \left\| \frac{1}{2}(\tanh(w) + 1) - x \right\|_2^2 + c \cdot g\left(\frac{1}{2}(\tanh(w) + 1)\right), \quad (2.24)$$

where the change of variable $\delta^{(i)} = \frac{1}{2}\tanh(w^{(i)} + 1) + x^{(i)}$ is applied, and the function g is the one defined in (2.23). For the ℓ_∞ norm, they propose the following optimization problem:

$$\min_{\delta} \quad c \cdot g(x + \delta) + \sum_i \max(\delta^{(i)} - \tau, 0), \quad (2.25)$$

where τ is a parameter that controls which components of δ are penalized. It starts at 1 and is decreased by a factor of 0.9 at each iteration if $\delta^{(i)} < \tau$ for all i . The CW attack obtained by solving either (2.24) or (2.25) depending on the norm used, is considered one of the strongest white-box attacks, and it is widely used to evaluate the robustness of classifiers.

2.3 Defense mechanisms

A large body of research has been dedicated to developing defense mechanisms against adversarial attacks. The most famous approach involves adding adversarial examples during the training phase, a technique known as *adversarial training*. Other types of defenses involve ensembles of classifiers, adding stochasticity to the

model, or using architectures that are inherently robust to adversarial attacks. In the following, we present some of the most popular defense mechanisms that are considered effective and that are relevant to this thesis.

2.3.1 Adversarial training

The most popular defense mechanism is *adversarial training*, which was introduced by Goodfellow et al. [GSS15] and later formalized and popularized by Madry et al. [Mad+18]. The idea is to train the classifier with adversarial examples in addition to the regular training data. This way, the classifier learns to be robust to the adversarial perturbations, and it is able to generalize better to unseen adversarial data. Ideally, adversarial training tries to minimize the empirical adversarial risk in (2.13), and this is often done in practice by approximating the inner supremum in (2.13) with an attack algorithm like PGD or FGSM. A very general pipeline of vanilla adversarial training is presented in Algorithm 1.

Algorithm 1: General pipeline of adversarial training

Require: Training data $\mathcal{D}$, classifier h to train, attack algorithm $A(h, x, y)$.

- 1: **for** number of epochs **do**
 - 2: **for** mini-batch of training data **do**
 - 3: Pick (x, y) from mini-batch
 - 4: $x_{adv} \leftarrow A(h, x, y)$
 - 5: Update parameters of h using x and/or x_{adv} and y
 - 6: **return** Trained classifier h
-

Adversarial training is simple to implement, but it is computationally expensive, as it requires generating adversarial examples for each example of the training data. Bartoldson et al. [Bar+24] use a 10 step PGD adversary during training and estimate a 9x increase in the total FLOPs compared to normal training. Many works have tried to find ways to accelerate adversarial training, like Shafahi et al. [Sha+19], who recycled gradient information computed during the update of the model parameters to perform what they call “free” adversarial training, or Zhang et al. [Zha+19] who restricted both forward and backward passes of the adversarial update to the initial layer of the network.

Being simple to implement and effective in practice, adversarial training is very often paired with other defense strategies. However, the effectiveness of adversarial training together with other defense techniques has been heavily criticized by Tramer et al. [Tra+20]. For example, when evaluating the k-Winners-Take-All activation defense proposed by Xiao et al. [XZZ20], they find that adding adversarial training results in *lower* robustness. Similar results were found by Athalye et al. [ACW18],

in which some evaluated defenses showed no improvement when paired with adversarial training.

An important recent trend in adversarial training is the use of synthetic data generated by diffusion models to augment the training data, as proposed by Gowal et al. [Gow+21] and Rebuffi et al. [Reb+21]. The coupling of adversarial training with synthetic data augmentation has created some of the most robust models according to the Robustbench benchmark [Cro+21]. More recently, Wang et al. [Wan+23] made yet another improvement by simply using a diffusion model with a better FID, arguing that the performance of the diffusion model used to generate the synthetic data is crucial to the success of the defense.

2.3.2 Certified robustness

One branch of research on defense mechanisms tried to get away from the adversarial training paradigm, but more specifically from the back-and-forth oscillation between attacks and defenses that has characterized the field. Instead of proposing a defense that could be broken by an adaptive attack, as had happened with many defenses in the past, the idea was to create a defense that was provably robust to a certain degree. This idea created the subfield of *certified defenses*, which aim to provide a certificate that the classifier is robust to a certain level of perturbation.

Lipschitz neural networks. Hein and Andriushchenko [HA17] provide a theoretical analysis that exhibits a connection between the Lipschitz constant of each class score function and the robustness of the classifier. This connection is also studied by Tsuzuku et al. [TSS18], who are able to prove the following result: let $f : \mathbb{R}^d \rightarrow \mathbb{R}^K$ be a score-based classifier with Lipschitz constant L and (x, y) be a labeled data point. Define the *margin* of f at x as

$$m_f(x, y) = \max_{j \neq y} \left[f(x)^{(y)} - f(x)^{(j)} \right].$$

For any $\epsilon > 0$, if $m_f(x, y) > \sqrt{2}L\epsilon$, then for every $x' \in B_2(x, \epsilon)$,

$$\arg \max_k f(x')^{(k)} = y.$$

In other words, the Lipschitz constant of a classifier provides an immediate way of providing point-wise robustness guarantees (called certificates) based on the margin of the classifier.

Based on previous links between robustness to norm bounded perturbations and generalization developed by Xu and Mannor [XM12], Cisse et al. [Cis+17] derive an

upper bound on the adversarial risk of a classifier with respect to any loss function that depends on the standard risk and the Lipschitz constant of both the loss function and the classifier. This motivated Cisse et al. [Cis+17] to propose *Parseval networks*, which are neural networks that are designed and trained to have a small Lipschitz constant by constraining the Lipschitz constant of each layer to be smaller than 1.

Numerous following works have tackled the task of designing more general neural networks with controlled Lipschitz constant, like convolutional or residual networks. In the following, we make a broad overview of recent works in this direction. Meunier et al. [Meu+22] propose a framework for constructing 1-Lipschitz residual neural networks by analyzing residual networks through a continuous-time dynamical systems perspective, introducing the Convex Potential Layer. Araujo et al. [Ara+21] introduce a tight and computationally efficient upper bound for the Lipschitz constant of convolutional layers using Toeplitz matrix theory, enabling effective training of Lipschitz-regularized convolutional neural networks. Prach and Lampert [PL22] introduce a rescaling-based weight matrix parametrization, Almost-Orthogonal Lipschitz (AOL), which guarantees a Lipschitz constant of at most 1 for any linear layer while being efficient, easy to implement, and compatible with any training objective or optimization method. Delattre et al. [Del+23] propose a precise, fast, and differentiable upper bound for the spectral norm of convolutional layers using circulant matrix theory and Gram iteration, demonstrating superior precision, efficiency, and scalability for Lipschitz regularization of convolutional neural networks. Recently, Araujo et al. [Ara+23] present a unifying algebraic framework for designing 1-Lipschitz neural networks, connecting diverse approaches through a common semidefinite programming condition.

Randomized smoothing. A second approach takes an arbitrary classifier h and creates a new smoothed classifier g by convolving h with a Gaussian function. This approach was originally proposed by Lecuyer et al. [Lec+19], who took inspiration from differential privacy and presented the first broadly applicable certified defense that could scale to large networks and datasets.

The smoothed classifier g returns the class that the base classifier h is most likely to return when x is perturbed by a small Gaussian perturbation. More formally, the smoothed classifier is defined as:

$$g(x) = \operatorname{argmax}_{y \in \mathcal{Y}} \mathbb{P}_{\eta \sim \mathcal{N}(\mathbf{0}_d, \sigma^2 I_d)}[h(x + \eta) = y]. \quad (2.26)$$

Cohen et al. [CRK19] prove that if g predicts the correct class y_a with probability p_a and the second most probable class with probability p_b , then the smoothed classifier g is robust in the ball $B_2(x, \epsilon_\sigma)$ with ϵ_σ given by

$$\epsilon_\sigma = \frac{\sigma}{2} (\Phi^{-1}(p_a) - \Phi^{-1}(p_b)),$$

where Φ is the cumulative distribution function of the standard normal distribution. In practice, the smoothed classifier is approximated using Monte Carlo sampling, and instead of computing p_a and p_b , a lower and upper bound $\underline{p}_a$ and $\overline{p}_b$ are computed respectively, to obtain a certified radius ϵ_σ that is valid with high probability.

Salman et al. [Sal+19] designed an adapted attack for smoothed classifiers, and use it to perform adversarial training and further improve the *provable* robustness of the final smoothed model. Recently, Delattre et al. [Del+24] propose a novel certification procedure for improving the certified robustness of classifiers through randomized smoothing, leveraging enhanced Lipschitz bounds, variance-margin trade-offs, and Bernstein's inequality. This work is an interesting mix of the ideas presented by Tsuzuku et al. [TSS18] and Cohen et al. [CRK19].

2.3.3 Diverse ensembles

One remarkable property of adversarial attacks that was noticed as early as 2014 by Szegedy et al. [Sze+14] is that adversarial examples generated for one classifier are often misclassified by other classifiers. This *transferability* of adversarial attacks is particularly problematic for techniques such as model ensembles, which rely on the idea that different classifiers have independent errors. This is the starting point for the development of *diverse ensembles* as a defense mechanism against adversarial attacks, which try to enforce diversity between the classifiers in the ensemble to reduce the transferability of adversarial attacks and create an ensemble that is more robust than any of its individual components. There are two common approaches that have been taken to improve diversity: 1) enforcing the classifiers to produce different scores for the same input on the non-true labels, 2) reducing the alignment of the gradients of different classifiers on a given input to reduce the effectiveness of gradient-based attacks.

Pang et al. [Pan+19] proposed the *adaptive diversity promoting* (ADP) regularizer that encourages the score vectors of non-true labels of the different classifiers to be almost orthogonal and uniformly distributed. They consider an ensemble of m score functions composed with the softmax operator $f_1, \dots, f_m : \mathcal{X} \rightarrow \Delta^K$, and they define the $(K - 1 \times m)$ -matrix $F^{-y}(x)$ in which the j th column is the vector

obtained after removing the y -th component of $f_j(x)$ and normalizing (in ℓ_2 norm) for $j = 1, \dots, m$.

Let $f(x) = \frac{1}{m} \sum_{i=1}^m f_i(x)$ be the average of the scores of the ensemble. The ADP regularizer is defined as:

$$\text{ADP}_{\alpha, \beta}(x, y) := \alpha \cdot H(f(x)) + \beta \cdot \log(\mathbb{ED}(x, y)), \quad (2.27)$$

where $H(f(x)) = -\sum_{j=1}^K f(x)^{(j)} \log(f(x)^{(j)})$ is the entropy of the categorical probability distribution with parameter $f(x) \in \Delta^K$ and $\mathbb{ED}(x, y)$ is the *ensemble diversity* metric, defined as follows:

$$\mathbb{ED}(x, y) := \det(F^{-y}(x)^T F^{-y}(x)). \quad (2.28)$$

The important intuition is that the determinant in (2.28) is equal to the square of the volume of the parallelepiped spanned by the columns of $F^{-y}(x)$, which is maximal when the columns are orthogonal. The ADP regularizer pushes the score vectors (without the y -th component) to be orthogonal, while also pushing the whole score vector to be uniformly distributed. The intuition behind this choice is that different models should match their prediction on the true label, but should be as different as possible on the other labels while remaining uniform to avoid being targeted by an attacker. This regularizer is used in conjunction with adversarial training to improve the robustness of the classifier, but it was proven to be ineffective by Tramer et al. [Tra+20], who were able to drop the accuracy under attack of this defense to 0%.

Kariyappa and Qureshi [KQ19] consider another regularizer using the alignment of the gradients of the classifiers, trying to induce orthogonality between them in order to reduce the transferability of gradient-based attacks. For each pair of classifiers, the cosine similarity of the gradients is computed:

$$\text{cs}(i, j) := \frac{\nabla_x \mathcal{L}(f_i(x), y) \cdot \nabla_x \mathcal{L}(f_j(x), y)}{\|\nabla_x \mathcal{L}(f_i(x), y)\|_2 \|\nabla_x \mathcal{L}(f_j(x), y)\|_2}. \quad (2.29)$$

The GAL regularizer is defined as a smooth approximation of the maximum cosine similarity between the gradients of the classifiers:

$$\text{GAL}(x, y) := \log \left(\sum_{1 \leq i < j \leq m} \exp(\text{cs}(i, j)) \right). \quad (2.30)$$

Yang et al. [Yan+21] propose a similar idea, with the addition of enforcing model smoothness. They define the Transferability Reduced Smooth (TRS) ensemble training strategy using the following regularization term for each pair of models:

$$\text{TRS}(x, y, i, j, \tau) := \text{cs}(i, j) + \left[\max_{\|\hat{x}-x\|_\infty \leq \tau} \|\nabla_{\hat{x}} \mathcal{L}(f_i(\hat{x}), y)\|_2 + \|\nabla_{\hat{x}} \mathcal{L}(f_j(\hat{x}), y)\|_2 \right] \quad (2.31)$$

Yang et al. [Yan+20] propose DVERGE, a regularization technique particularly crafted for convolutional neural networks that isolates the adversarial vulnerability in each sub-model by distilling non-robust features.

The results reported by Yang et al. [Yan+21] on CIFAR-10 and CIFAR-100 suggest that none of these methods were able to improve the robustness of the classifier under attack. ADP and GAL are reported to have $\approx 0\%$ robust accuracy on both datasets, and all methods have a robust accuracy of less than 5% on the CIFAR-100 dataset (see [Yan+21, Table 1, Table 12]). Added to results obtained by Tramer et al. [Tra+20] for ADP, this suggests that these methods are not entirely effective at improving the robustness of the ensemble of classifiers.

2.3.4 Bayesian neural networks (BNNs)

Bayesian neural networks (BNNs) naturally arise when the weights of a neural network are treated as random variables. In this framework, it is more adapted to consider a neural network as a function $f : \mathbb{R}^d \times \mathbb{R}^p \rightarrow \Delta^K$, where d is the dimension of the input space and p the dimension of the weight space. The initial belief about the weights is encoded in a prior distribution $p(w)$. The goal is to update our belief on the weights that could have produced the observed labeled data, and for that the posterior distribution of the weights given the data is computed using Bayes' theorem:

$$p(w|\mathcal{D}) = \frac{p(\mathcal{D}|w)p(w)}{p(\mathcal{D})}, \quad (2.32)$$

where $p(\mathcal{D}|w)$ is the likelihood of the data given the weights of the model. The posterior distribution is then used to make predictions, and for a given input x , the probability of it being of label y can be obtained by averaging the predictions of the model over the posterior distribution of the weights:

$$p(y|x, \mathcal{D}) = \int p(y|x, w)p(w|\mathcal{D})dw. \quad (2.33)$$

In practice, the quantity $p(\cdot|x, w)$ is modeled using the neural network $f(x, w)$ that outputs a vector in Δ^K and represents a probability distribution over labels associated to input x and weights w . Thus, the prediction of a BNN is given by

$$f_{\text{BNN}}(x) = \int f(x, w)p(w|\mathcal{D})dw. \quad (2.34)$$

As the exact form of the posterior distribution is intractable, approximate inference methods are used to compute an approximate posterior distribution $\hat{p}(w|\mathcal{D})$. Then,

the prediction is approximated using Monte Carlo sampling from the approximate posterior. Thus, the prediction of a BNN in practice is given by

$$\hat{f}_{\text{BNN}}(x) = \frac{1}{n} \sum_{i=1}^n f(x, w_i), \quad w_i \sim \hat{p}(w|\mathcal{D}). \quad (2.35)$$

A bayesian neural network can be thought as an infinite ensemble of classifiers, in which the weights used to aggregate the model predictions are given by the posterior distribution. Similar to ensembles, the robustness of a BNN is also related to the transferability of attacks between the different classifiers $f(\cdot, w)$ when $w \sim p(w|\mathcal{D})$.

Carbone et al. [Car+20] and Bortolussi et al. [Bor+24] analyze the geometry of adversarial attacks in the context of BNNs, demonstrating that vulnerability to gradient-based attacks arises from data degeneracy, and show that BNNs exhibit robustness to such attacks in the large-data, overparameterized limit. They state that under the correct assumptions on the width of the neural network, the amount of data and the convergence of the training algorithm, the average gradient over the posterior distribution of the BNN is null.

This property implies that BNNs are protected against gradient-based attacks, and suggests a direct link between the quality of the approximation of the posterior and the robustness of the final model in (2.34). This apparent theoretical guarantee has been at the root of a recent trend of using BNNs as a defense mechanism against adversarial attacks. However, these results are often demonstrated on small datasets and models, and have been recently questioned by Feng et al. [Fen+24], who detected problems in the original robustness evaluation and was able to decrease the robust accuracy of the models proposed by Carbone et al. [Car+20] from 97% to less than 2.5% in the MNIST dataset.

2.4 Randomized defense mechanisms

Randomized defenses are a class of defenses that involve adding randomness at inference time, with the objective of creating a less predictable classifier that is harder to attack⁴. In this section, we are going to present some of the most popular randomized defenses and discuss their effectiveness. We are going to give particular

⁴Randomized smoothing and Bayesian neural networks (Section 2.3) are commonly considered to be randomized. In theory, both methods are deterministic, and it is the practical implementations that turn them into randomized defenses. For example, the theoretical definition randomized smoothing involves convolving a deterministic classifier with a Gaussian distribution to create a new smoothed **deterministic** classifier. However, as this is untractable in practice, sampling is used in order to approximate the theoretical smoothed classifier. In this work, we consider the theoretical definitions of both randomized smoothing and BNNs, and therefore we do not include them in this section

attention to mixtures of classifiers, which are the main focus of this thesis and constitute a general framework that cover many randomized defenses proposed in practice.

2.4.1 Random transformations of the input

Two of the first uses of randomization to improve robustness to adversarial attacks can be traced back to the 2017 Neurips *Adversarial Attacks and Defences* Competition. Zhang and Liang [ZL19] obtained the first place in the competition (out of 107 submissions) by injecting Gaussian noise to the input and then quantizing it, a technique they named “randomized discretization”. The second place in the competition was obtained by Xie et al. [Xie+18], who proposed to use random resizing and random padding of input images. This defense was later shown to be ineffective by Athalye et al. [ACW18]. This idea of applying random transformations to the input was later revisited by Raff et al. [Raf+19], who used 25 different transformations, achieving what was believed at the time to be an impressive robustness on the ImageNet dataset [Den+09]. However, their defense was shown to be ineffective three years later by Sitawarin et al. [SGSW22].

A particular random transformation of the input that has been widely explored is *noise injection*, which adds random noise to the input of the model before performing the prediction. Randomized classifiers based on noise injection have the advantage of being simple to implement and computationally efficient, and also have been backed by theoretical work. Pinot et al. [Pin+22] provide a formal framework for understanding the robustness of general randomized classifiers, based on the definition of (ϵ, η, D) -robustness, defined as follows. Fix a p -norm, and a probability metric D (e.g. Total Variation). Then, a randomized classifier $h : \mathcal{X} \rightarrow \Delta^K$ is said to be (ϵ, η, D) -robust if for all $x, x' \in \mathcal{X}$

$$\|x - x'\|_p \leq \epsilon \implies D(h(x), h(x')) \leq \eta. \quad (2.36)$$

The first contribution of Pinot et al. [Pin+22] is to show a connection between the notion of robustness in (2.36) and the gap between the standard and adversarial risk. Explicitly, let TV denote the total variation distance, and $\mathcal{H}_{\epsilon, \eta}^{\text{TV}}$ denote the set of $(\epsilon, \eta, \text{TV})$ -robust classifiers for a fixed p -norm. Then for any data distribution ρ ,

$$h \in \mathcal{H}_{\epsilon, \eta}^{\text{TV}} \implies \mathcal{R}_{\epsilon, p}(h, \rho) - \mathcal{R}(h, \rho) \leq \eta. \quad (2.37)$$

In other words, Pinot et al. [Pin+22] are able to describe a class of randomized classifiers for which the gap between robust and standard accuracy can be bounded, providing an optimistic view on the question regarding the tradeoff between standard and robust accuracy. This is not trivial, as earlier work by Tsipras et al. [Tsi+19]

argued that robust and standard accuracy were at odds, backing their claims with examples in which it was provably impossible to achieve both. The result in (2.37) is complemented with a bound on the generalization gap of classifiers in $\mathcal{H}_{\epsilon,\eta}^{\text{TV}}$. Together, these two results provide a bound on the *adversarial generalization gap* for this family of randomized classifiers. This suggests that minimizing the empirical risk in this hypothesis class is a good strategy to minimize the true adversarial risk.

The second contribution of Pinot et al. [Pin+22] is a practical way to design the hypothesis class $\mathcal{H}_{\epsilon,\eta}^D$ by convolving a fixed deterministic classifier f with Gaussian noise. For the case of the TV, they prove that any deterministic classifier f can be transformed into a $(\epsilon, 2\Phi(\frac{\epsilon}{2\sigma}), \text{TV})$ -robust randomized classifier against ℓ_2 adversaries by considering the convolved classifier

$$h(x) = \mathbb{E}_{\delta} [f(x + \delta)], \quad \delta \sim \mathcal{N}(\mathbf{0}_d, \sigma^2 I_d). \quad (2.38)$$

Lastly, Pinot et al. [Pin+22] prove in [Pin+22, Proposition 3] a connection between $(\epsilon, \eta, \text{TV})$ -robust randomized classifiers and randomized smoothing. Note that the randomized smoothing classifier is obtained by taking the most probable label from the classifier defined in (2.38). For some fixed x , let $\hat{y}_1$ and $\hat{y}_2$ be the two labels with highest predicted probability under $h(x)$. If h is $(\epsilon, \eta, \text{TV})$ -robust and $h(x)^{(\hat{y}_1)} - h(x)^{(\hat{y}_2)} > 2\eta$, the following holds:

$$\forall x' \in B_2(x, \epsilon), \quad \operatorname{argmax}_{k \in \mathcal{Y}} h(x')^{(k)} = \operatorname{argmax}_{k \in \mathcal{Y}} h(x)^{(k)}. \quad (2.39)$$

While the noise injection mechanism has not been independently refuted, numerous studies indicate that it offers limited robustness. For instance, Gao et al. [Gao+22] examine the limitations of stochastic pre-processing defenses and successfully circumvent a defense strategy that employs random input transformations, including noise injection. Given that this defense is more comprehensive than solely applying noise injection, their findings suggest that noise injection alone is ineffective in enhancing classifier robustness. As a second example, Sitawarin et al. [SGSW22] show that the *barrage of random transformations* defense proposed by Raff et al. [Raf+19], which incorporates noise injection, is less effective than initially believed.

2.4.2 Implicit sub-model selection

To create a randomized classifier, one can fix a base classifier and then sample a sub-model by pruning some of the neurons during inference, similar to the idea of dropout [Hin+12]. The first of such randomized defenses was proposed by Dhillon et al. [Dhi+18], who randomly prune activations at each layer based on the magnitudes of their activations using a multinomial distribution. An illustration

of this procedure is depicted in Figure 2.6. This defense was later shown to be ineffective by Athalye et al. [ACW18].

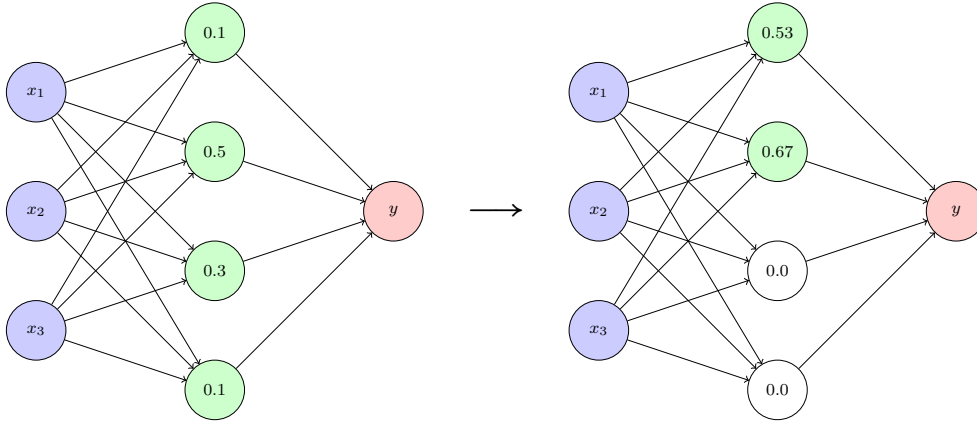


Fig. 2.6.: Illustration of the pruning procedure proposed by Dhillon et al. [Dhi+18]. The left figure shows the original neural network after the activations of the hidden layer (green) have been computed, and the right figure shows the pruned network.

In a somehow similar fashion, Panousis et al. [Pan+21] propose stochastic k -winner-take-all activation functions, based on previous work that had explored this kind of competitive activations in the deterministic setting [XZZ20]. Instead of pruning neurons at each layer independently, Panousis et al. [Pan+21] group the neurons of a given layer into competitive groups from which only k of them are selected at random, where the probability of being selected is proportional to the activation of the neuron. They also conceive a method that can prune connections between neurons of consecutive layers, opening the possibility for much stronger pruning strategies. These defense techniques have been circumvented: the pruning technique by Dhillon et al. [Dhi+18] was broken by Athalye et al. [ACW18], and the k -winner-take-all activation functions by Panousis et al. [Pan+21] was not accepted to the Robustbench benchmark after their model was allegedly found to be much less robust than originally claimed by using a small adaptation (see the following discussion: <https://github.com/fra31/auto-attack/issues/58>).

2.4.3 Mixtures of classifiers

Inspired from the idea of mixed strategies in zero-sum games, mixtures of classifiers constitute a general framework to describe many randomized approaches. The core idea is to define a probability distribution over a set of deterministic classifiers $\mathcal{H}$, usually with a small finite support, and at each inference step, sample one model from $\mathcal{H}$ independently to make a prediction. Mathematically speaking, mixtures of classifiers are defined as follows:

Definition 2.4.1 (Mixture of classifiers [Pin+20; Meu+21]).

Given a classification task over the space $\mathcal{X} \times \mathcal{Y}$, a set of deterministic classifiers $\mathcal{H}$, and a probability distribution μ over $\mathcal{H}$, the mixture h_μ is a randomized classifier defined as follows: for a point $x \in \mathcal{X}$ and a label $y \in \mathcal{Y}$, we have

$$h_\mu(x)^{(y)} = \mathbb{P}_{h \sim \mu} [h(x) = y]. \quad (2.40)$$

Recall that deterministic classifiers can also be defined as functions $h : \mathcal{X} \rightarrow \Delta^K$. In this case, the definition of the mixture h_μ can be simplified to

$$h_\mu(x) = \mathbb{E}_{h \sim \mu} [h(x)]. \quad (2.41)$$

Motivating finite mixtures against data perturbing adversaries. Mixtures of classifiers as in Equation (2.41) were proposed in the context of adversarial robustness by Pinot et al. [Pin+20], who showed that under certain conditions, any deterministic binary classifier could be outperformed by a mixture of two classifiers. In their work, authors model the adversary's action as a pair of measurable functions $\phi = (\phi_{-1}, \phi_1)$ that map each point $(x, y) \in \mathcal{X} \times \{-1, 1\}$ into a perturbed point $(\phi_y(x), y)$ that respects the adversary's budget constraints, that is, $\phi_y(x) \in B_p(x, \epsilon)$. Let Ψ_ϵ be the set of all such pairs of functions that the adversary could use, and define the hypothesis class $\mathcal{H}$ as follows:

$$\mathcal{H} = \{h : x \rightarrow \text{sign}(g(x)) \mid g : \mathcal{X} \rightarrow \mathbb{R} \text{ continuous}\}. \quad (2.42)$$

With this model of the adversary, the adversarial examples problem becomes a standard two-player zero-sum game, in which the defender proposes a classifier $h \in \mathcal{H}$ to minimize the adversarial risk, and the adversary chooses a pair of functions $\phi \in \mathcal{F}_\epsilon$ to maximize it. This game can be formalized as the following minmax problem:

$$\inf_{h \in \mathcal{H}} \sup_{\phi \in \Psi_\epsilon} \mathcal{R}_\Psi(h, \phi) := \mathbb{E}_{(x, y) \sim \rho} [\mathcal{L}^{0-1}(h(\phi_y(x)), y)]. \quad (2.43)$$

Before going into the results of Pinot et al. [Pin+20], we need to introduce some game-theoretical terminology. For a fixed strategy ϕ of the adversary, a classifier h^* is said to be a *best response* to ϕ if $\mathcal{R}_\Psi(h^*, \phi) = \min_{h \in \mathcal{H}} \mathcal{R}_\Psi(h, \phi)$. Analogously, a strategy ϕ^* is a best response to a classifier h if $\mathcal{R}_\Psi(h, \phi^*) = \max_{\phi \in \Psi_\epsilon} \mathcal{R}_\Psi(h, \phi)$. A pair of strategies (h^*, ϕ^*) is a *Pure Nash equilibrium* if h^* is a best response to ϕ^* and ϕ^* is a best response to h^* . Such a pair of strategies represent a stable point of the game, in which neither player can improve their adversarial risk by changing their strategy. This also means that no attack can be more harmful for the proposed classifier, and also that the classifier is optimal against the worst possible attack.

Pinot et al. [Pin+20] start their analysis by showing that the game in (2.43) can have trivial Nash equilibria, and that these undesirable equilibria arise from the fact that the adversary has unlimited computational power, and also no constraint apart from the maximal norm of the perturbation. In a practical example of bypassing a security system (security camera, malware detection, spam filtering), this means that the adversary will perturb *every* input, and that she can modify every input as *much as possible* without incurring into any kind of cost or risk of being detected. As discussed by the authors, this is highly unrealistic and contradicts the more reasonable assumption that an attacker wants to be as stealthy as possible, attacking only key inputs and doing so with the minimal perturbation needed in an effort to avoid detection. This intuition leads Pinot et al. [Pin+20] to slightly modify the game in (2.43) by introducing a penalty $\Omega \in \{\Omega_{\text{norm}}, \Omega_{\text{mass}}\}$ for the attacker and consider the modified game

$$\inf_{h \in \mathcal{H}} \sup_{\phi \in \Psi_\epsilon} \mathcal{R}_\Psi^\Omega(h, \phi) := \mathcal{R}_\Psi(h, \phi) + \lambda \Omega(\phi). \quad (2.44)$$

The two explored penalties are the following:

$$\Omega_{\text{norm}}(\phi) := \mathbb{E}_{(x,y) \sim \rho} [\|\phi_y(x) - x\|_p], \quad \Omega_{\text{mass}}(\phi) := \mathbb{E}_{(x,y) \sim \rho} [\mathbf{1}[\phi_y(x) \neq x]]. \quad (2.45)$$

Pinot et al. [Pin+20] show that the game in (2.44) does not admit a Pure Nash equilibrium for any of the two penalties and any $\lambda \in (0, 1)$, which implies that 1) no attack can be simultaneously optimal for a given classifier *and* transferable to other classifiers, and more importantly 2) no deterministic classifier can be robust against every attack. Intuitively, the non-existence of a pure Nash equilibrium means that there is no way out of the back-and-forth between the defender and the adversary: training a robust classifier will always lead to a new attack that is optimal to it, and training a classifier that is robust against the new attack will lead to yet another new attack, and so on.

Inspired by this negative result concerning pure strategies, Pinot et al. [Pin+20] turn their attention to the immediate extension of *mixed strategies*, or mixtures of classifiers. They are able to prove that for any deterministic classifier h , there exists a mixture of two classifiers that is strictly better than h for the game in (2.44). This result is formalized in the following theorem:

Theorem 2.4.2 (Adapted from Pinot et al. [Pin+20], Theorem 2).

Fix an arbitrary data distribution ρ , p -norm, and $\epsilon > 0$ and $\Omega \in \{\Omega_{\text{norm}}, \Omega_{\text{mass}}\}$. For any $h \in \mathcal{H}$, there exist a classifier $h' \in \mathcal{H}$ and a mixture h_μ over h, h' such that:

$$\max_{\phi \in \Psi_\epsilon} \mathcal{R}_\Psi^\Omega(h_\mu, \phi) < \max_{\phi \in \mathcal{F}_\epsilon} \mathcal{R}_\Psi^\Omega(h, \phi). \quad (2.46)$$

This result implies that for the regularized problem in (2.44), extending the hypothesis class $\mathcal{H}$ to finite mixtures over $\mathcal{H}$ can provide a strict improvement in robustness. On a practical level, Pinot et al. [Pin+20] propose a boosting algorithm to train robust mixtures. Nevertheless, this method was shown to be ineffective by Dbouk and Shanbhag [DS22], which leaves the question of how to train robust finite mixtures of classifiers open.

Mixed Nash equilibria in the adversarial examples game. Extending on the work of Pinot et al. [Pin+20], Meunier et al. [Meu+21] study mixtures of classifiers against a *non-regularized, randomized* adversary. They propose another zero-sum game reformulation of the adversarial risk minimization problem, similar to (2.43) and demonstrate that under certain assumptions on the loss function, this game has no duality gap and therefore it always admits an approximate *mixed* Nash equilibria.

Meunier et al. [Meu+21] allow the adversary to use randomized strategies by considering *adversarial distributions* instead of adversarial functions. To define adversarial distributions, we need to introduce measure theoretic concepts. Let $\mathcal{M}_1^+(\mathcal{Z})$ denote the set of Borel probability distributions over $\mathcal{Z}$. Given two measurable spaces $\mathcal{W}, \mathcal{Z}$, a measurable function $g: \mathcal{W} \rightarrow \mathcal{Z}$ and a distribution $\gamma \in \mathcal{M}_1^+(\mathcal{W})$, the push-forward measure $g\#\gamma$ is a distribution in $\mathcal{M}_1^+(\mathcal{Z})$ defined as follows:

$$\forall A \subseteq \mathcal{Z} \text{ measurable, } (g\#\gamma)(A) = \gamma(g^{-1}(A)). \quad (2.47)$$

A coupling between two distributions $\rho \in \mathcal{M}_1^+(\mathcal{W}), \pi \in \mathcal{M}_1^+(\mathcal{Z})$ is a distribution $\gamma \in \mathcal{M}_1^+(\mathcal{W} \times \mathcal{Z})$ such that $\text{proj}_1\#\gamma = \rho$ and $\text{proj}_2\#\gamma = \pi$, where proj_i denotes the projection on the i -th coordinate. We denote $\Gamma(\rho, \pi)$ the set of couplings between ρ and π . Lastly, for a given $\gamma \in \mathcal{M}_1^+(\mathcal{Z})$, we say an event $A \subseteq \mathcal{Z}$ happens γ -almost surely (γ -a.s.) if it happens with probability 1 w.r.t. γ , i.e. $\gamma(A) = 1$.

For a given data distribution $\rho \in \mathcal{M}_1^+(\mathcal{X} \times \mathcal{Y})$ and a budget $\epsilon > 0$ for the adversary, the set of *adversarial distributions* is defined as follows:

$$\mathcal{A}_\epsilon(\rho) = \{ \pi \in \mathcal{M}_1^+(\mathcal{X} \times \mathcal{Y}) \mid \exists \gamma \in \Gamma(\rho, \pi) \text{ with } x' \in B_p(x, \epsilon), y = y' \gamma\text{-a.s.} \}. \quad (2.48)$$

Note that the definition of the adversary using an adversarial distribution $\pi \in \mathcal{A}_\epsilon(\rho)$ allows the adversary to split the mass of a point (x, y) into different points (x', y) . Intuitively, the adversary can choose a distribution defined via a conditional density $q(\cdot \mid (x, y)) \in \mathcal{M}_1^+(\mathcal{X} \times \mathcal{Y})$ that has support on $B_p(x, \epsilon) \times \{y\}$, and then sample from this distribution to obtain the adversarial example (x', y) .

Regarding the hypothesis class and loss function, Meunier et al. [Meu+21] consider parametric classifiers h_θ with parameter space Θ , and a Borel measurable, bounded

loss function $\mathcal{L} : \Theta \times (\mathcal{X} \times \mathcal{Y}) \rightarrow [0, \infty)$ such that $\mathcal{L}(\theta, \cdot)$ is upper-semi continuous for every θ . They consider randomized classifiers as mixtures over Θ , and with the adversary in (2.48) they provide a new zero-sum game reformulation of the adversarial risk minimization problem:

$$\inf_{\mu \in \mathcal{M}_1^+(\Theta)} \sup_{\pi \in \mathcal{A}_\epsilon(\rho)} \mathcal{R}_{\epsilon, p}(h_\mu, \pi) \quad (2.49)$$

Meunier et al. [Meu+21] show that this game has no duality gap, and if the assumptions of 1) compactity of the parameter space Θ , and 2) lower semi-continuity of the function $\mathcal{L}(\cdot, (x, y))$ for all (x, y) are added, then there always exist a mixed Nash equilibrium (μ^*, π^*) . Such a stable state is desirable for both players, and from the defender's perspective, it means that there is a randomized classifier that is optimal against the worst possible attack.

On a practical level, Meunier et al. [Meu+21] introduce a learning algorithm designed to train robust mixtures of classifiers using a boosting-like approach. However, their method has not been independently evaluated by other researchers. For their own evaluation, they rely solely on two variants of PGD with EOT [Ath+18], as provided by the AutoAttack library [CH20]. This limited evaluation raises concerns, especially given that Dbouk and Shanbhag [DS22] demonstrated that PGD-like algorithms are not effective for attacking finite mixtures. Consequently, it is reasonable to question whether the robustness of the mixtures proposed by Meunier et al. [Meu+21] is as high as they claim.

On the generality of mixtures of classifiers. Mixtures of classifiers cover a wide range of randomized defenses proposed in the literature. Take for example defenses based on random input transformations. If h is the base deterministic classifier used as the backbone of the randomized defense and $t_1, \dots, t_T$ is the set of T input transformations that are randomly selected at inference, then we can define a family of deterministic classifiers $\mathcal{H} = \{h_t \mid h_t(x) = h(t(x))\}$ and consider μ to be the uniform distribution over $\mathcal{H}$. Then, the mixture h_μ is exactly the randomized classifier proposed by Xie et al. [Xie+18] or Raff et al. [Raf+19]. An analogous reasoning can be done to classifiers based on noise injection, and more generally to any transformation of the input, model parameters or activations, that is independently sampled at each inference step.

Randomized classifiers based on pruning could also be modeled as mixtures, in which case the set $\mathcal{H}$ consists of all possible sub-models that can be obtained by pruning neurons from a backbone model h . In this case, however, the pruning process that happens layer-wise will depend on the activations of the neurons, making the sampling process dependent on x . This is a more complex scenario that

is not covered by our current definition of mixtures of classifiers, but that could be adapted if we allowed a family of distributions μ_x that depended on the input x .

2.5 Discussion on the importance of strong attack algorithms

The definition of robustness to adversarial attacks based on (2.11) requires the classifier to be robust to any possible perturbation that respects the specified constraints. However, as the supremum in (2.10) is often intractable to compute, empirical evaluations of the adversarial risk using any attack algorithm can only offer a lower bound on the real empirical adversarial risk. This lower bound will be tight if the attack algorithm is strong enough to find the worst-case perturbation for the classifier at every point.

The impossibility of computing the real empirical adversarial risk has led the community into ambiguous and confusing results concerning the robustness of certain defense mechanisms. It became common to see defenses that were claimed to be robust, but that were later shown to be ineffective against stronger attacks that adapted to them. In this section we go over a non-exhaustive list of defenses that were shown to be ineffective and the works that made those discoveries. At the end of the section, we will discuss the importance of strong attacks for the case of randomized classifiers, specifically finite mixtures of classifiers.

A notable early effort to rigorously test a defense mechanism was conducted by Carlini and Wagner [CW17a], who evaluated defenses based on detecting adversarial examples and were able to circumvent them. When Carlini and Wagner [CW17b] introduced the CW attack, they also demonstrated that defensive distillation was ineffective at improving classifier robustness. Since these foundational studies, numerous works have adopted a similar approach, testing proposed defenses against strong, carefully designed attacks.

Athalye et al. [ACW18] were able to break seven out of nine defenses proposed at ICLR 2018 by using attack improvements that countered the phenomenon known as *gradient obfuscation*. This was done by approximating non-differentiable operations in the proposed defenses, accounting for randomness in the forward passes, and by reparametrizing the models that had vanishing or exploding gradients due to implementation choices. This was an important step in the field of adversarial robustness, as it provided a systematic review of defense techniques in a top machine learning conference and counter-measures to overcome them. It also showed that

the field was still in its infancy and that many defenses were not robust, but simply harder to attack.

Later, in 2020, Tramer et al. [Tra+20] were able to circumvent thirteen defenses presented at top machine learning conferences (ICML, ICLR, Neurips), including the diversity ensemble technique proposed by Pang et al. [Pan+19]. Very recently, Sitawarin et al. [SGSW22] were able to break the defense proposed by Raff et al. [Raf+19] using a complex attack that combined multiple techniques. Concerning Bayesian Neural Networks, very recent work by Feng et al. [Fen+24] questions previous evaluations by Carbone et al. [Car+20] and Bortolussi et al. [Bor+24].

All these works that take on the endeavor of evaluating defense mechanisms proposed in the literature have highlighted the lack of rigorous evaluation and benchmarking of defenses. Some initiatives have been taken to address this issue, like the RobustBench benchmark by Croce et al. [Cro+21], which aims to provide a standardized evaluation of the robustness of classifiers using a mix of strong attack algorithms. Other efforts aim at establishing a correct methodology for evaluating the robustness of classifiers. For example, Carlini et al. [Car+19] enumerate multiple good practices when evaluating a defense, and Athalye et al. [ACW18] also established guidelines to check that a defense did not depend on obfuscated gradients. However, as Popovic et al. [Pop+22] highlighted, these guidelines are not exhaustive, and many defenses cite them as justification without delivering genuine robustness improvements.

2.5.1 Attacking randomized defenses.

Randomization techniques have been widely used to bypass existing attacks, providing great robustness overestimates. However, the community is more aware of the importance of strong attacks to evaluate randomized defenses, as shown by the works of Gao et al. [Gao+22], who find that most defenses based on stochastic pre-processing are weaker than previously thought. Lucas et al. [Luc+23] even suggests that randomized defenses are less secure than their deterministic counterparts when the adversary can attack repeatedly.

The defense based on local competitive activation functions by Panousis et al. [PCT21] was debated on an Github issue⁵ by the authors of Robustbench [Cro+21], who found that their model was non-robust using a simple adaptation of AutoAttack [CH20]. Among the seven defenses broken by Athalye et al. [ACW18] were the stochastic pruning defense by Dhillon et al. [Dhi+18], and the random resizing and padding defense by Xie et al. [Xie+18] that was on the podium at the 2017 Neurips

⁵Available at <https://github.com/fra31/auto-attack/issues/58>

competition. The extension to multiple random image transformations by Raff et al. [Raf+19] was broken very recently by Sitawarin et al. [SGSW22] with a complex attack. This non-exhaustive list only shows that the subfield of randomized defenses is also subject to the same issues as deterministic defenses, and that strong attacks are necessary to evaluate them properly.

Attacks against finite mixtures of classifiers. Arguably one of the simplest randomized defenses are finite mixtures of classifiers, and yet they are not trivial to attack. When evaluating their mixtures, Pinot et al. [Pin+20] clearly tried to adapt the current state-of-the-art attacks to their setting by accounting for the randomness of the classifier, but it was later shown by Dbouk and Shanbhag [DS22] that their attack remained weak and that their results were overestimating the robustness of the defense. This is a clear example that even the simplest randomized defenses are not trivial to attack, and will be the starting point of this thesis: in Chapter 3 we are going to tackle the problem of attacking finite mixtures of classifiers both theoretically and in practice, with the goal of providing a strong attack that has better theoretical guarantees and that can provide a tighter estimate of the robustness of finite mixtures.

Attacking finite mixtures of classifiers

Contents

3.1	Failures of existing attacks	40
3.1.1	The problems of PGD against finite mixtures	40
3.1.2	ARC attack and its limitations	42
3.1.3	Desirable properties of an attack for mixtures	46
3.2	Lattice Climber attack	51
3.2.1	Binary linear classifiers	52
3.2.2	Multi-class differentiable classifiers	54
3.2.3	Experiments on synthetic data with linear classifiers	55
3.2.4	Experiments on CIFAR-10	56
3.3	Discussion	60

As discussed in Section 2.5, the absence of strong attacks targeting specific defenses poses a significant limitation to research in adversarial robustness, as it prevents accurate evaluation of a defense’s true effectiveness. This challenge has persisted throughout the field, and randomized defenses are no exception. For instance, the defense based on multiple random transformations proposed by Raff et al. [Raf+19] was initially considered robust—until *three years later*, when Sitawarin et al. [SGSW22] successfully broke it.

Strong attacks play a critical role not only in exposing the limitations of existing defenses but also in driving progress by motivating the development of more robust solutions. Without rigorous and adaptive attacks, the true strength of a defense remains uncertain, potentially creating a false sense of security. Furthermore, strong attacks serve a dual purpose: beyond evaluating defenses, they can be directly utilized to enhance robustness through adversarial training.

In this chapter, we approach randomized classifiers from the attacker’s perspective, focusing on the question of *how to effectively attack a finite mixture of classifiers*. In Section 3.1, we highlight the shortcomings of traditional attack algorithms, such as PGD, when applied to mixtures. We then review the state-of-the-art attacks for mixtures, identifying key areas for improvement. Building on these insights, in

Section 3.2, we propose a novel attack that addresses these limitations and advances the current landscape of adversarial attacks.

3.1 Failures of existing attacks

The development of an effective attack algorithm for finite mixtures of classifiers begins with a thorough understanding of the limitations of existing methods. In this section, we first analyze early adaptive attacks for mixtures, which include the application of the Expectation Over Transformation (EOT) principle proposed by Athalye et al. [Ath+18], and a related approach that targets the average output of the mixture used by Pinot et al. [Pin+20]. We will see that these approaches rely on the assumption that all classifiers in the mixture can be attacked simultaneously—a limitation that was later highlighted by Dbouk and Shanbhag [DS22] in their development of ARC, a more advanced attack against mixtures. While ARC represents a significant step forward, we will discuss its remaining limitations, which provide valuable insights for identifying potential improvements.

3.1.1 The problems of PGD against finite mixtures

Expectation Over Transformation (EOT). One important step in the development of attacks against randomized classifiers was the introduction of *Expectation Over Transformation* (EOT) by Athalye et al. [Ath+18], which intended to craft adversarial attacks that remained adversarial even when the input was subject to random transformations. EOT introduces an expectation over the transformations $t \sim T$ applied to the input, and optimizes the expected loss in a targeted or untargeted fashion:

$$\mathbb{E}_{t \sim T} [\mathcal{L}(h(t(x')), y)]. \quad (3.1)$$

The EOT principle has been transposed to the problem of attacking a finite mixture of classifiers h_μ over $\mathcal{H} = \{h_1, \dots, h_m\}$ by Pinot et al. [Pin+20] and Meunier et al. [Meu+21]. Instead of considering random transformations, the idea is to account for the sampling of the classifier as the source of randomness. Thus, our objective function becomes the *Expectation Of the Loss* (EOL), defined as follows:

$$\mathbb{E}_{h \sim \mu} [\mathcal{L}(h(x), y)] = \sum_{i=1}^m \mu_i \mathcal{L}(h_i(x), y), \quad (3.2)$$

where $\mu_i = \mu(\{h_i\})$ is the weight of classifier h_i in the mixture. Note that if we set the loss to be the zero-one loss, then maximizing the objective in (3.2) directly corresponds to maximizing the classification error of the mixture. Therefore, *maximizing the EOL with the zero-one loss is the correct objective for attacking a finite mixture.*

However, as we will demonstrate in the following, replacing the zero-one loss with alternative loss functions, such as the cross-entropy loss, can lead to unintended and undesirable consequences.

Practical adaptations for attacking finite mixtures. There are two approaches to attack finite mixtures in practice based on (3.2). The first and most natural approach is to keep the original formulation and directly maximize the EOL (3.2):

$$\sup_{x' \in B_p(x, \epsilon)} \sum_{i=1}^m \mu_i \mathcal{L}(h_i(x'), y). \quad (3.3)$$

The second approach, which we call *Loss Of Expectation* (LOE), modifies the objective in (3.2) and maximizes the loss computed on the expected output of the mixture. This was done by Pinot et al. [Pin+20] following the good practices of Tramer et al. [Tra+20] to improve the effectiveness of the attack against multiple classifiers. The optimization problem is then

$$\max_{x'} \mathcal{L} \left(\sum_{i=1}^m \mu_i h_i(x'), y \right). \quad (3.4)$$

In practice, both of these problems are solved using first-order optimization methods like PGD. We refer to these attacks as EOL-PGD and LOE-PGD, respectively. We will now show that both of these approaches rely on the implicit assumption that all classifiers can be attacked simultaneously and that their vulnerabilities are *aligned*, a concept we illustrate in Figures 3.1 and 3.2. To show this, we will analyze the gradients of the objective functions of both problems w.r.t. the input.

First, the gradient of EOL in (3.3) can be directly computed due to the linearity of the gradient operator:

$$\nabla_x \sum_{i=1}^m \mu_i \mathcal{L}(h_i(x'), y) = \sum_{i=1}^m \mu_i \nabla_x \mathcal{L}(h_i(x), y). \quad (3.5)$$

Secondly, the gradient of LOE in (3.4) is equal to:

$$\nabla_x \mathcal{L} \left(\sum_{i=1}^m \mu_i h_i(x'), y \right) = \sum_{i=1}^m \gamma_i \nabla_x \mathcal{L}(h_i(x), y), \quad (3.6)$$

where the coefficients γ_i are given by

$$\gamma_i = \frac{\mu_i \cdot \mathcal{L}'(\sum_{j=1}^m \mu_j h_j(x), y)}{\mathcal{L}'(h_i(x), y)}. \quad (3.7)$$

Note that in both cases, the gradients are a linear combination of the gradients of the loss of the individual classifiers. Moreover, under reasonable assumptions, all the coefficients are non-zero. Thus, using first order methods to solve problems (3.3) or

(3.4) is intuitively trying to attack all classifiers simultaneously, and the success of such attack relies on the assumption that this linear combination is a good attack direction. This can be effective in a scenario like the one depicted in Figure 3.1, in which all classifiers are vulnerable and their vulnerabilities are aligned.

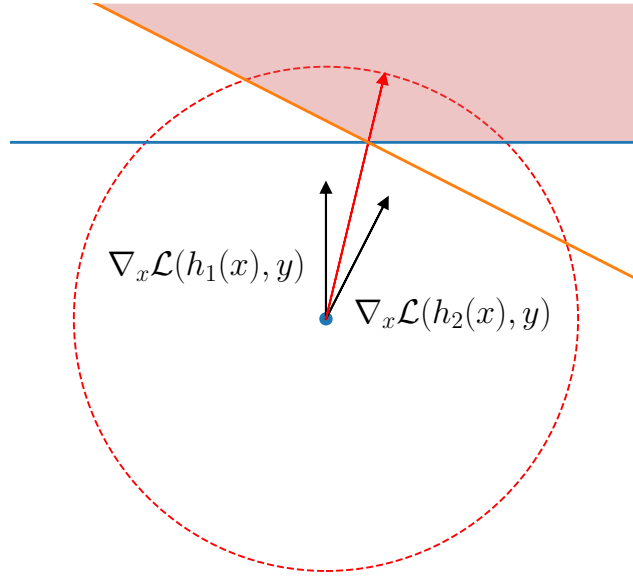


Fig. 3.1.: Example of two linear classifiers with high alignment. The red arrow represents the gradient of the loss of the mixture, while the black arrows represent the gradient of the loss of individual classifiers. The red region in the top-right corner represents the points that are adversarial for both classifiers simultaneously. In this case, the linear combination of the individual gradients is a successful attack direction.

However, following a linear combination of the gradients of individual models can guide the attack towards a perturbation that induces zero error. This happens when the common vulnerabilities of the classifiers lie outside of the set of admissible perturbations, as demonstrated in Figure 3.2. In this case, and as highlighted by Dbouk and Shanbhag [DS22], EOL-PGD and LOE-PGD will fail to find a perturbation that increases the error, even though such perturbations exist. For example, simply attacking one of the classifiers increases the error from 0 to 0.5, and this attack is optimal in Figure 3.2. This issue with EOL-PGD and LOE-PGD is the starting point for the development of ARC, a stronger attack against mixtures of classifiers, which we will discuss in the following section.

3.1.2 ARC attack and its limitations

Dbouk and Shanbhag [DS22] criticize EOL-PGD and LOE-PGD because of their lack of *consistency*¹ [DS22, Theorem 5.1], meaning that even if there exist adversarial

¹This is the term used by authors in the original work. We rather use the term *effectiveness*. See Section 3.1.3

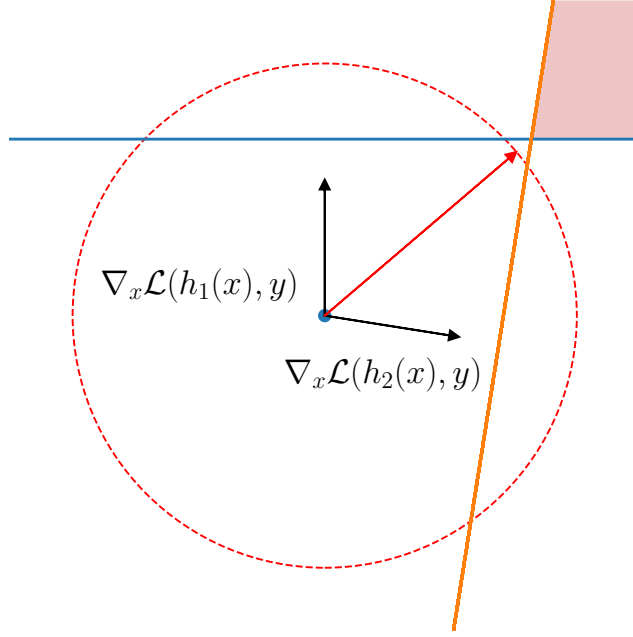


Fig. 3.2.: Example of two linear classifiers with low alignment. The red arrow represents the gradient of the loss of the mixture, while the black arrows represent the gradient of the loss of individual classifiers. The red region in the top-right corner represents the points that are adversarial for both classifiers simultaneously. Note that that this red region lies *outside* of the ϵ -ball. In this case, the linear combination of the individual gradients is *not* an effective attack direction as it leads to a perturbation that induces zero error, even though there exist perturbations that induce an error of 0.5.

attacks that increase the error of the mixture, EOL-PGD or LOE-PGD can miss them (i.e. Figure 3.2). This motivates the authors to create *Attacking Randomized ensembles of Classifiers* (ARC) [DS22], an attack against finite mixtures of classifiers that is guaranteed to be consistent in binary classification against linear classifiers. In the rest of this section, we will compare ARC to EOL-PGD and discuss the limitations of ARC. We omit the discussion of LOE-PGD, but our analysis applies to it as well.

A simplified version of ARC in the case of binary linear classifiers is presented in Algorithm 2 (see [DS22, Algorithm 1] for full details). The key difference between ARC and EOL-PGD is that ARC attacks the classifiers one by one, instead of trying to attack all of them simultaneously. At each iteration, ARC follows the direction that attacks the one classifier at hand, and at the end of the iteration, the new perturbation will be kept only if the error of the whole mixture was strictly increased. The greedy approach of ARC makes it provably consistent when attacking binary linear classifiers (see [DS22, Theorem 5.1]). If we revisit a situation akin to Figure 3.2, we can see that ARC is able to attack one of the classifiers successfully, while EOL-PGD fails to find a perturbation that induces an error greater than zero. This is shown in Figure 3.3.

Algorithm 2: ARC attack for binary linear classifiers (simplified) [DS22]

Require: Mixture h_μ over set of linear binary classifiers $\{h_1, \dots, h_m\}$, point (x, y) .

- 1: Sort classifiers by decreasing order of their weights $\mu^{(i)}$.
 - 2: Initialize perturbation $\delta = 0$.
 - 3: **for** each classifier h_i **do**
 - 4: Compute optimal perturbation δ_i against h_i .
 - 5: Compute β_i (See [DS22]).
 - 6: **if** $\mathcal{L}^{0-1}(h_\mu(x + \delta + \beta_i \delta_i), y) > \mathcal{L}^{0-1}(h_\mu(x + \delta), y)$ **then**
 - 7: $\delta = \delta + \beta_i \delta_i$.
 - 8: **else**
 - 9: Discard δ_i .
 - 10: **return** perturbation δ guaranteed to be consistent.
-

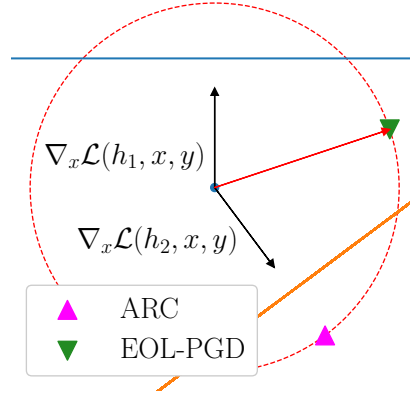


Fig. 3.3.: Comparison between the perturbation proposed by EOL-PGD and ARC for a mixture of two linear classifiers with low alignment. ARC first attacks one of the classifiers successfully (orange, lower right), and then tries to attack the other one without success. ARC keeps the first perturbation that was able to increase the zero-one loss, and ignores the second one. This capability of ARC allows it to outperform EOL-PGD in this type of situations.

Dbouk and Shanbhag [DS22] train mixtures of two classifiers using the method proposed by Pinot et al. [Pin+20] and find that when attacked with ARC², the robustness of these mixtures drops significantly compared to when they are attacked with EOL-PGD or LOE-PGD, which was the attack used by Pinot et al. [Pin+20] for their evaluation. This behavior remains consistent across neural network architectures, datasets, and norms considered. Thus, ARC has proven to be a much stronger attack than EOL-PGD and LOE-PGD and remains, to this day, the state-of-the-art attack against finite mixtures of classifiers.

Although ARC resolves the main issue with EOL-PGD, it does so at the cost of losing the ability to attack multiple classifiers simultaneously in situations of medium

²ARC is born from an analysis in the binary linear case, but it is generalized to the case of multi-class differentiable classifiers like neural networks. In their experiments comparing to Pinot et al. [Pin+20], they used the latter.

alignment. Experimentally, one can verify that even for two linear classifiers in $\mathbb{R}^2$, ARC may fail to find a perturbation that misclassified by both classifiers simultaneously when the region of common vulnerabilities within the ϵ -ball is very small. An example of such scenario is shown in Figure 3.4.

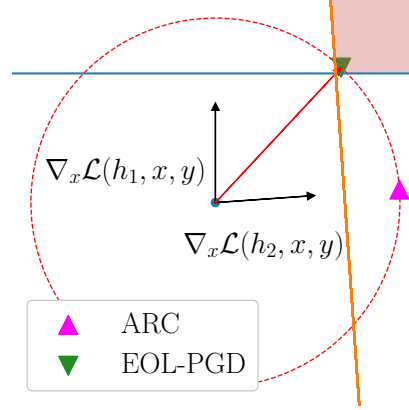


Fig. 3.4.: Comparison between the perturbation proposed by EOL-PGD and ARC on a mixture of two linear classifiers that have their intersection very close to the border of the ϵ -ball. In this case, EOL-PGD outperforms ARC. Note that ARC remains consistent.

To further demonstrate this issue, we test both EOL-PGD and ARC in situations that range from high to low alignment by changing the angle between the normal vectors of the linear classifiers from 0 degrees (perfect alignment) to 180 degrees (opposite normal vectors, low alignment). When the angle is 0, both classifiers are equal, and the optimal attack direction is the same. Then, as the angle increases, the intersection point of both linear classifiers starts moving towards the boundary of the ϵ -ball (northeast in Figure 3.4). At some point, this intersection goes outside the ϵ -ball, at which moment the classifiers stop being simultaneously vulnerable, like in Figure 3.2.

In Figure 3.5, we plot the error induced by the perturbations found by both EOL-PGD and ARC and compare it to the optimal error, which is determined by checking if the intersection of the two linear classifiers lies within the ϵ -ball or not. It can be seen that no attack dominates the other and that each outperforms the other in some regime that depends on the alignment of the decision boundaries. When classifiers are almost equal, both attacks are optimal, but ARC drops in performance as the classifiers become less aligned. Then, at around 90 degrees, the classifiers stop being simultaneously vulnerable so the optimal error becomes $1/2$. In this regime, ARC is able to attack one of the classifiers and ignore the other one, achieving an optimal error. EOL-PGD, on the other hand, fails and achieves an error of 0.

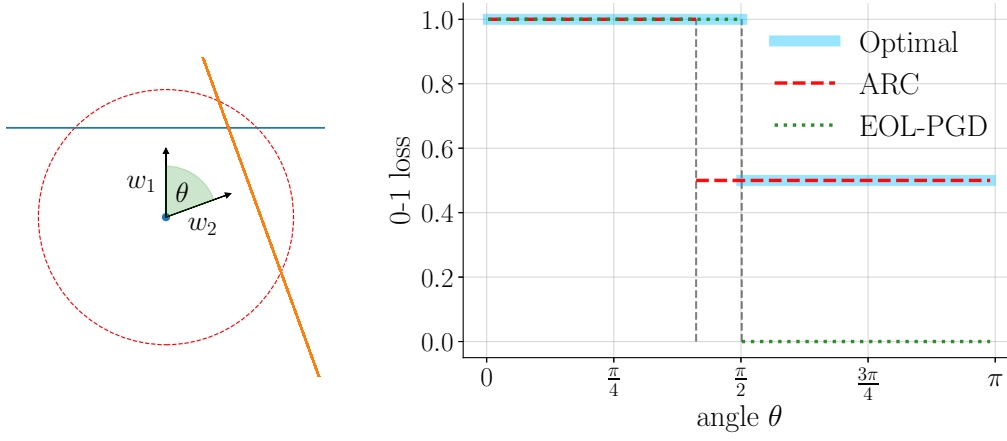


Fig. 3.5.: Comparison between EOL-PGD and ARC w.r.t the level of alignment of the linear classifiers. On the left figure, a schema of the experiment that we run for a fixed angle θ . On the right figure, the error of the perturbations found by EOL-PGD and ARC compared to the optimal error w.r.t. θ . The black vertical lines indicate the change of regime, the first one being when ARC stops being optimal and the second the moment in which the classifiers stop being simultaneously vulnerable. It can be seen that no attack dominates the other. Each outperforms the other in some regime that depends on the alignment of the decision boundaries.

We end this section with a comparison of ARC, EOL-PGD and LOE-PGD through the lens of the directions used to craft the perturbation at each iteration of the attack. Recall that the gradient directions analyzed in (3.5) and (3.6) were linear combinations of the gradients of the loss of individual classifiers. In the case of EOL-PGD and LOE-PGD, the coefficients of this linear combination were non-zero for *all* classifiers. In the case of ARC, these coefficients are all zero except for one, which corresponds to the classifier being considered at the current step (See [DS22, Algorithm 2] for full details).

3.1.3 Desirable properties of an attack for mixtures

After having discussed the limitations of existing attacks, in this section we will formalize the problem of attacking a mixture of classifiers as a combinatorial optimization problem, more explicitly as the problem of enumerating maximal elements of a certain lattice. Using this formalism, we will discuss the desirable properties of an attack algorithm that faces mixtures of classifiers. This will provide a new understanding of the consistency guarantee of ARC, and will allow us to identify a clear improvement opportunity: *maximality*.

Formalizing the problem of attacking a finite mixture. Note that a given deterministic classifier h partitions the input space into at most K non-overlapping regions $h^{-1}(j) \subseteq \mathcal{X}$, $j \in [K]$, where each region corresponds to a class it predicts. Moreover,

an optimal untargeted attacker only cares about forcing a prediction *different* from the true label y inside the set of admissible perturbations $B_p(x, \epsilon)$. In that sense, each classifier h induces a partition of the set of admissible perturbation $B_p(x, \epsilon)$ into two disjoint regions of interest:

$$B_p(x, \epsilon) = \underbrace{B_p(x, \epsilon) \cap h^{-1}(y)}_{\text{non-adversarial}} \sqcup \underbrace{B_p(x, \epsilon) \cap \bigcup_{j \neq y} h^{-1}(j)}_{\text{Vulnerability region } V(h)}. \quad (3.8)$$

We call *vulnerability region* of h the set $B_p(x, \epsilon) \cap \bigcup_{j \neq y} h^{-1}(j)$, and denote it as $V(h)$. An example is shown in Figure 3.6 for two classifiers, in which the coloured regions represent the vulnerability regions of each classifier.

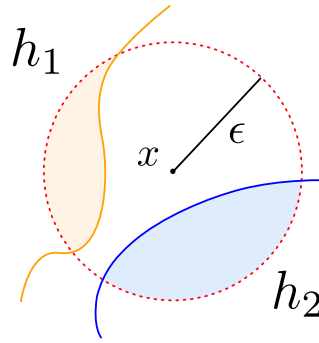


Fig. 3.6.: Example of the vulnerability regions of two classifiers around a point x with label y . The regions $V(h_1)$ and $V(h_2)$ are represented in orange and blue, and contain all the points $x' \in B_p(x, \epsilon)$ for which $h_1(x') \neq y$ and $h_2(x') \neq y$, respectively.

With the definition of vulnerability region, attacking a deterministic classifier h becomes equivalent to finding a point $x' \in V(h)$. This will allow us to rewrite the error of a finite mixture in the following way: Consider a finite mixture h_μ over $\mathcal{H}$ and a point (x, y) . The error under attack of h_μ can be written as the following sum:

$$\mathcal{L}_{\epsilon, p}^{0-1}(h_\mu(x), y) = \sup_{x' \in B_p(x, \epsilon)} \sum_{h \in \mathcal{H}} \mu(\{h\}) \mathbb{1}\{h(x') \neq y\}, \quad (3.9)$$

and including the vulnerability regions yields

$$\mathcal{L}_{\epsilon, p}^{0-1}(h_\mu(x), y) = \sup_{x' \in B_p(x, \epsilon)} \sum_{h \in \mathcal{H}} \mu(\{h\}) \mathbb{1}\{x' \in V(h)\}. \quad (3.10)$$

Equation (3.10) highlights that the optimal attack belongs to the intersection of the vulnerability regions of the classifiers that maximizes the total mass according to μ . Let us define the *common vulnerability region* of a subset of classifiers as

$$CV(\mathcal{H}') = \bigcap_{h \in \mathcal{H}'} V(h) \setminus \bigcup_{h \notin \mathcal{H}'} V(h). \quad (3.11)$$

In simple terms, $CV(\mathcal{H}')$ is the set of points that are adversarial for exactly the classifiers in $\mathcal{H}'$. With this definition, the error of the mixture on a point x' that belongs to $CV(\mathcal{H}')$ is exactly

$$\mathcal{L}^{0-1}(h_\mu(x'), y) = \sum_{h \in \mathcal{H}'} \mu(\{h\}). \quad (3.12)$$

and if $\mathcal{H}_1 \subseteq \mathcal{H}_2$ and $CV(\mathcal{H}_2) \neq \emptyset$, then the following holds:

$$\forall x_1 \in CV(\mathcal{H}_1), \forall x_2 \in CV(\mathcal{H}_2), \mathcal{L}^{0-1}(h_\mu(x_1), y) \leq \mathcal{L}^{0-1}(h_\mu(x_2), y), \quad (3.13)$$

which means that an attacker prefers to attack in the common vulnerability region of a larger set of classifiers whenever it is not empty.

The number of non-empty common vulnerability regions for a mixture of m classifiers is upper bounded by 2^m . Figure 3.7 shows the four possible configurations for a mixture composed of two distinct classifiers, from the most convenient for the defender (configuration (a) on the left) to the most convenient for the attacker (configuration (d) on the right).

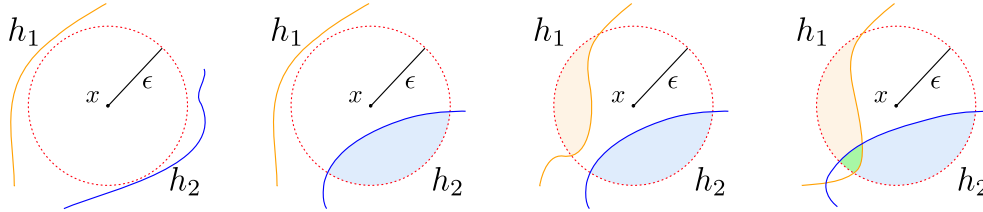


Fig. 3.7.: Four possible configurations for a mixture with two classifiers h_1 and h_2 assuming x is correctly classified by both. a) Both h_1 and h_2 are non-vulnerable, the mixture is robust. b) Only one classifier is vulnerable. c) Both h_1 and h_2 are vulnerable, but they can not be attacked simultaneously. d) Both h_1 and h_2 are vulnerable, and they can be attacked on the same region. Best viewed in color.

Equation (3.13) suggests that subsets of classifiers can be ordered by their preference for the attacker. Formally speaking, we can define a partial order relation $\leq$ as

$$\mathcal{H}_1 \leq \mathcal{H}_2 \iff \mathcal{H}_1 \subseteq \mathcal{H}_2 \text{ and } CV(\mathcal{H}_2) \neq \emptyset. \quad (3.14)$$

The order relation in Equation (3.14) induces a *lower semi-lattice structure* [DP02] in the family of subsets

$$\mathcal{S} = \{\mathcal{H}' \subseteq \mathcal{H} \mid CV(\mathcal{H}') \neq \emptyset\} \cup \{\emptyset\}. \quad (3.15)$$

For simplicity, we refer to it as the *adversarial lattice* of h_μ at (x, y) . Figure 3.8 illustrates two examples of an adversarial lattice for a mixture of two classifiers. In both cases, it is assumed that the classifiers correctly classify the point x , meaning

x is not in any vulnerability region. Consequently, the lattice's minimal element is the empty set, which corresponds to points that are not adversarial for any classifier. With the lattice structure in mind, in the following we will use the lattice object to discuss desirable properties of an attack algorithm that faces mixtures of classifiers.

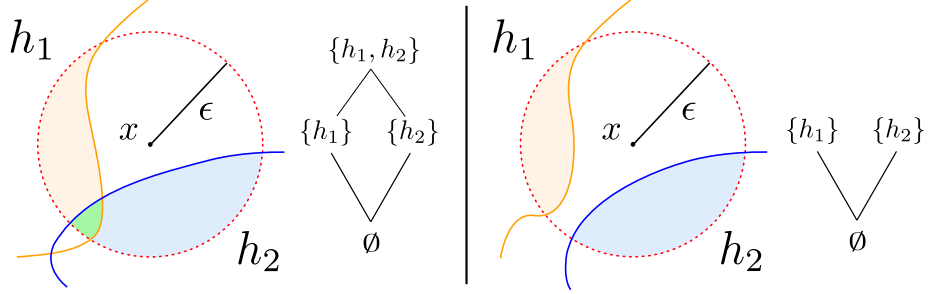


Fig. 3.8.: Examples of adversarial lattice for two of the configurations described in Figure 3.7. Best viewed in color.

Definition 3.1.1 (Effectiveness property).

An attack algorithm is effective for the hypothesis class $\mathcal{H}$ if, for any finite mixture over $\mathcal{H}$ and any point (x, y) , it can generate an adversarial example increasing the error of the mixture whenever such a point exists.

In the examples in Figure 3.8, where the attacker starts at point x , corresponding to the empty set in the adversarial lattice, an effective attack should be able to find an adversarial example in *some* common vulnerability region at *any* level of the adversarial lattice. This property, termed consistency by Dbouk and Shanbhag [DS22], is a defining characteristic of ARC for binary linear classifiers—one that EOL-PGD lacks.

Figure 3.9 illustrates an example of an adversarial lattice for a mixture of four classifiers. An attack with an *effectiveness* guarantee is ensured to produce an adversarial attack x' that corresponds to *some* highlighted node. However, there is no guarantee regarding the level of the lattice that is reached, and the effectiveness guarantee becomes less meaningful as the size of the lattice increases.

Definition 3.1.2 (Maximality property).

We say an attack algorithm is maximal for the hypothesis class $\mathcal{H}$ if, for any finite mixture over $\mathcal{H}$ and any point (x, y) , it can generate an adversarial example for a maximal subset of classifiers of the adversarial lattice.

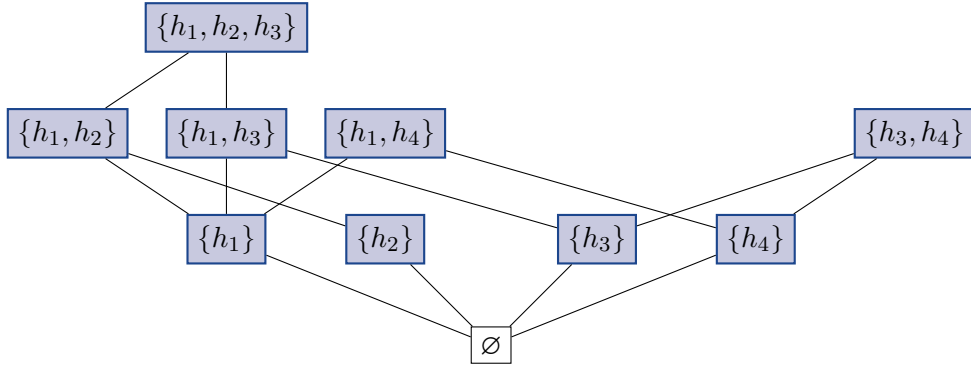


Fig. 3.9.: Example of an adversarial lattice for a mixture of four classifiers with uniform weights. The highlighted nodes represent the sets of classifiers that can be attacked by an attack with the *effectiveness* guarantee.

In Figure 3.10, we present the same lattice from Figure 3.9 but highlighted the maximal nodes, emphasizing that the maximality guarantee is a stronger property than the effectiveness guarantee. Note that in configuration (c) of Figure 3.7, the effectiveness and maximality guarantees coincide because the adversarial lattice has only one level. In configuration (d), however, the maximality guarantee becomes stronger. In such configuration, neither EOL-PGD nor ARC is maximal for binary linear classifiers, as counterexamples can be constructed where they fail to find a maximal subset of classifiers to attack (Figures 3.3 and 3.4).

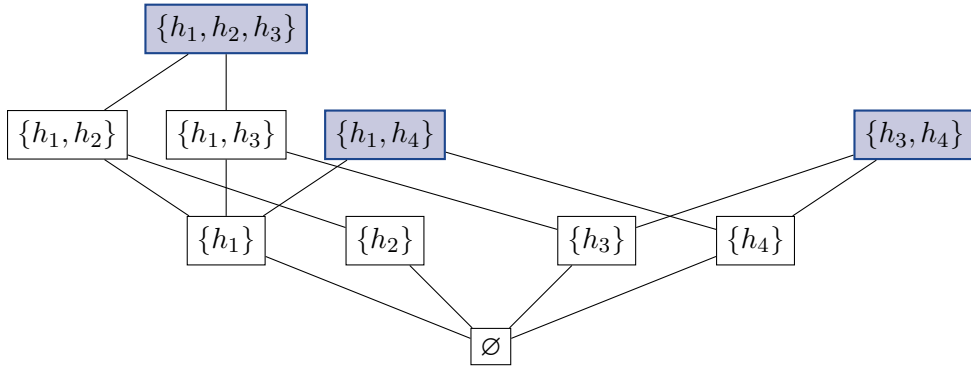


Fig. 3.10.: Example of an adversarial lattice for a mixture of four classifiers with uniform weights. The highlighted nodes represent the sets of classifiers that can be attacked by an attack with the *maximality* guarantee.

Definition 3.1.3 (Optimality property).

An attack algorithm is optimal for the hypothesis class $\mathcal{H}$ if, for any finite mixture over $\mathcal{H}$ and any point (x, y) , it can generate an adversarial example that achieves the highest possible error.

The most desirable property of an algorithm attacking mixtures would be *optimality*. The optimal attack corresponds to a *subset* of the maximal elements of the adversarial lattice, depending on the weights of the mixture. In the example of Figure 3.10 with

uniform weights, the optimal attack would correspond to the set $\{h_1, h_2, h_3\}$, as it has the highest weight in the mixture. However, if the weights of the mixture were set to $\mu(\{h_1\}) = \mu(\{h_2\}) = 0.1$, and $\mu(\{h_3\}) = \mu(\{h_4\}) = 0.4$, the optimal attack would correspond to the set $\{h_3, h_4\}$.

Unfortunately, no polynomial-time algorithm is guaranteed to achieve optimality, even when using linear classifiers in binary classification, due to the following result (full proof in Appendix A.1.1.):

Theorem 3.1.4 (Hardness of attacking linear classifiers).

Consider a binary classification setting, a p -norm with $p > 1$ and $\epsilon > 0$ to be defined. Given a labeled point (x, y) , a set of m linear classifiers $x \mapsto \mathbb{1}\{\theta_i^\top x + b_i \geq 0\}$ where $(\theta_i, b_i) \in \mathbb{R}^{d+1}$, a uniform mixture h composed of these linear classifiers and a value $\beta > 0$, there exists a noise budget $\epsilon > 0$ such that the problem of checking if there exists $\delta \in B_p(0, \epsilon)$ such that $\mathcal{L}^{0-1}(h(x + \delta), y) \geq \beta$ is NP-hard.

As a consequence of Theorem 3.1.4, the attack algorithm that we will propose will not aim at achieving optimality. Instead, we will focus on *maximality*, which can be guaranteed in the case of binary linear classifiers and will give us effectiveness for free.

3.2 Lattice Climber attack

In this section, we introduce a new attack called Lattice Climber Attack (LCA) inspired from the maximality property. We first present a version of our attack that has maximality guarantees against mixtures of binary linear classifiers under mild assumptions, and then extend it to multi-class differentiable classifiers. Afterwards, we will present experimental results that compare LCA with EOL-PGD and ARC on synthetic and real-world datasets.

The idea behind LCA is to arrive at a maximal element of the lattice by climbing one level at a time, as illustrated in Figure 3.11. Similar problems have been studied under the name of *most specific sentences* or *maximal frequent itemsets* [AS+94; GMS97] in the domain of data mining and knowledge discovery, where algorithms like Apriori [AS+94], MaxMiner [BJ98] or AllMSS [GMS97] have been proposed. Our approach of climbing the lattice is akin to A_Random_MSS in [GMS97].

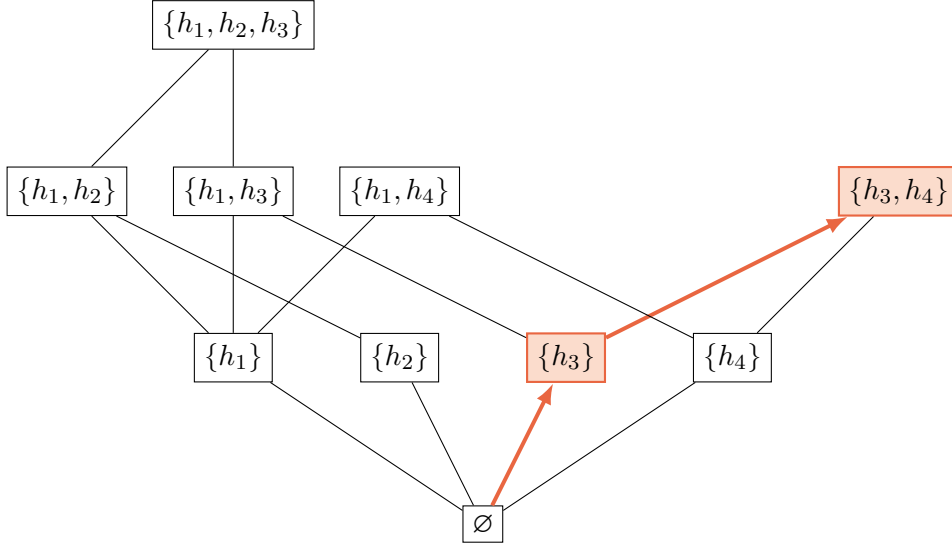


Fig. 3.11.: Main idea behind LCA with a given order $(3, 2, 4, 1)$. In this example, we attack first the classifier h_3 successfully. Then we try to add h_2 without success (h_3 and h_2 have no common vulnerability region). Then we add h_4 with success and finally we try to add h_1 without success. In the end, the attack was able to attack h_3 and h_4 simultaneously, inducing an error of $\mu(\{h_3\}) + \mu(\{h_4\})$. The other maximal elements in this example are $\{h_1, h_2, h_3\}$ and $\{h_1, h_4\}$, and each one of them could be optimal depending on μ .

3.2.1 Binary linear classifiers

In this section and for simplicity, we consider class labels $y \in \{-1, 1\}$ and classifiers $h : \mathbb{R}^d \rightarrow \{-1, 1\}$ of the form $h(x) = \text{sign}(f(x))$ for some linear function $f : \mathbb{R}^d \rightarrow \mathbb{R}$. In this setting, h correctly classifies the data point (x, y) if $yf(x) > 0$. Therefore, attacking h translates to minimizing $yf(x)$. The optimal attack direction and margin to the decision boundary of a single linear classifier are known for all ℓ_p norm with $p \geq 1$ [DS22, Appendix A].

The first component of our attack algorithm is a procedure to attack a subset of classifiers $\mathcal{H}'$ and find $x + \delta \in \text{CV}(\mathcal{H}')$ whenever $\text{CV}(\mathcal{H}') \neq \emptyset$. Similar to Perdomo and Singer [PS19], we consider the *reverse hinge loss* $\mathcal{L}^{\text{rev}}(yf(x)) = \max(yf(x), 0)$ as an objective to *minimize* in order to attack *one* binary linear classifier. Attacking a binary linear classifier with the traditional hinge loss would imply maximizing a convex function, for which there is no guarantee of convergence to a global optimum using algorithms like PGD. On the other hand, attacking a binary linear classifier with the reverse hinge loss is equivalent to *minimizing* a bounded convex function, for which PGD is guaranteed to converge to a global optimum.

Another useful property of the reverse hinge loss is that the expected reverse hinge loss over a finite mixture of classifiers is convex and equal to zero if and only if all

Algorithm 3: LCA for binary linear classifiers

Require: Set $\mathcal{H}$ of m binary linear classifiers in some order $(h_1, \dots, h_m)$, starting point $(x, y) \in \mathbb{R}^d \times \{-1, 1\}$. T number of iterations and η step size for PGD.

- 1: Initialize pool $\mathcal{H}_{\text{pool}} = \emptyset$, $\delta = 0_d$
 - 2: **for** $i = 1, 2, \dots, m$ **do**
 - 3: $\mathcal{H}_{\text{pool}} = \mathcal{H}_{\text{pool}} \cup \{h_i\}$ {Add h_i to the pool}
 - 4: Attack $\text{SRH}(\mathcal{H}_{\text{pool}}, \cdot, y)$ starting at $x + \delta$ with PGD (T, η) to find new perturbation $\hat{\delta}$
 - 5: **if** $\text{SRH}(\mathcal{H}_{\text{pool}}, x + \hat{\delta}, y) = 0$ i.e. succeeded **then**
 - 6: $\delta = \hat{\delta}$ {Update current attack, keep h_i in pool}
 - 7: **else**
 - 8: $\mathcal{H}_{\text{pool}} = \mathcal{H}_{\text{pool}} \setminus \{h_i\}$ {Reset pool to last state}
 - 9: **return** Adversarial example $x + \delta$
-

the classifiers in the mixture misclassify x . Then, in order to attack a set of classifiers $\mathcal{H}'$, we can minimize the *sum of reverse hinge losses* [PS19]

$$\text{SRH}(\mathcal{H}', x, y) = \frac{1}{|\mathcal{H}'|} \sum_{h \in \mathcal{H}'} \mathcal{L}^{\text{rev}}(yf(x)). \quad (3.16)$$

If there exists x' such that $\text{SRH}(\mathcal{H}', x', y) = 0$, then PGD with an appropriately chosen step size and sufficient number of iterations will converge to some x'' that is misclassified by all classifiers in $\mathcal{H}'$ (see [PS19, Theorem 3] for details).

The second component of our attack algorithm is a way to navigate the adversarial lattice. We propose a bottom-up navigation mechanism to climb the adversarial lattice up a branch, one step at a time, as illustrated in Figure 3.11. This mechanism, linear in the number of classifiers in the mixture, consists of keeping a pool of fooled classifiers, attempting to add each classifier in a fixed order. A classifier joins the pool only if it can be fooled alongside all current members; otherwise, it is discarded. The algorithm terminates after evaluating all classifiers.

The order in which classifiers are considered is a parameter of the algorithm, and similar to Dbouk and Shanbhag [DS22], we find that considering them in decreasing order of their associated probability yields good performance in general. For example, in the case $m = 2$, and with suitable parameters for the internal PGD, it ensures that LCA is optimal. Another reasonable option is to choose the order at random, which Dbouk and Shanbhag [DS23] found to be useful for ARC. The pseudocode of LCA in the binary linear setting is shown in Algorithm 3.

Maximality of LCA in the binary linear setting. In the binary linear classifier setting, PGD(T, η) is guaranteed to minimize the function $\text{SRH}(\mathcal{H}_{\text{pool}}, \cdot, y)$ because it is a convex optimization problem [PS19]. Thus, line 4 of Algorithm 3 ensures that we

will find an adversarial attack $x + \hat{\delta} \in \text{CV}(\mathcal{H}_{\text{pool}})$ if $\text{CV}(\mathcal{H}_{\text{pool}}) \neq \emptyset$. This enables climbing to the top of the adversarial lattice along a branch determined by the classifier order, as shown in Figure 3.11. This is formalized in the following lemma:

Lemma 3.2.1.

Consider a set of m binary linear classifiers $\mathcal{H}$. Fix $\epsilon > 0$ the attack budget and a point (x, y) . If there exists a point $x' \in B_p(x, \epsilon)$ such that $\text{SRH}(\mathcal{H}, x', y) = 0$, then there exist parameters T and η such that minimizing $\text{SRH}(\mathcal{H}, x, y)$ w.r.t. x with PGD(T, η) will return x'' such that $\text{SRH}(\mathcal{H}, x'', y) = 0$.

The proof of Lemma 3.2.1 is given in Appendix A.1.2. Lemma 3.2.1 allows us to prove that LCA is maximal for the set of binary linear classifiers (full proof in Appendix A.1.3):

Theorem 3.2.2 (LCA is maximal in the binary linear setting).

Let $\mathcal{H}$ be a finite set of binary linear classifiers and h_μ a mixture over $\mathcal{H}$. Fix $\epsilon > 0$ the attack budget and a p -norm with $p > 1$. For any $(x, y) \in \mathbb{R}^d \times \{-1, 1\}$, there exist parameters T and η for the inner PGD such that Algorithm 3 returns an adversarial example $x + \delta \in \text{CV}(\mathcal{H}')$, where $\mathcal{H}'$ is a maximal element of the adversarial lattice of h_μ at (x, y) .

3.2.2 Multi-class differentiable classifiers

The ideas developed in Section 3.2.1 for binary linear classifiers need to be adapted to the multi-class case with general differentiable classifiers, like neural networks, because we can no longer have the guarantees provided by Lemma 3.2.1 and Theorem 3.2.2. Moreover, the reverse hinge loss was defined for binary classifiers and not for multi-class classifiers.

In order to attack a classifier h based on the score function f at an arbitrary point (x, y) , we choose the target label $y_{\text{adv}} \in [K] \setminus \{y\}$ with the largest score according to $f(x)$ and minimize the reverse hinge loss of the margin between $f(x)^{(y)}$ and $f(x)^{(y_{\text{adv}})}$, i.e. $\mathcal{L}^{\text{rev}}(f(x)^{(y)} - f(x)^{(y_{\text{adv}})})$. This is a common choice that has been used by Perdomo and Singer [PS19] and also by Carlini and Wagner [CW17b], who found it to perform well as an objective function for crafting adversarial attacks. We denote SRH_m the sum of reverse hinge losses applied to a set of multi-class score classifiers using the margin described before.

Algorithm 4: LCA for multi-class classifiers

Require: Set of m classifiers h in some order $(h_1, \dots, h_m)$. Starting point (x, y) . T number of iterations and η step size for PGD.

- 1: Initialize pool $\mathcal{H}_{\text{pool}} = \emptyset$, $\delta = 0$
 - 2: **for** $i = 1, 2, \dots, m$ **do**
 - 3: $\mathcal{H}_{\text{pool}} = \mathcal{H}_{\text{pool}} \cup \{h_i\}$
 - 4: Attack $\text{SRH}_m(\mathcal{H}_{\text{pool}}, \cdot, y)$ starting at $x + \delta$ with PGD (T, η) to find new perturbation $\hat{\delta}$
 - 5: **if** $\mathcal{L}^{0-1}(h_\mu(x + \hat{\delta}), y) > \mathcal{L}^{0-1}(h_\mu(x + \delta), y)$ **then**
 - 6: $\delta = \hat{\delta}$
 - 7: Recompute pool $\mathcal{H}_{\text{pool}}$ according to current δ
 - 8: **return** $x + \delta$
-

Recall from (3.5) and (3.6) that the step used by EOL-PGD and LOE-PGD can be expressed as linear combinations of the gradients of the loss of individual classifiers. For LCA, similar to ARC, the coefficients of this linear combination will be sparse: only the models within the pool $\mathcal{H}_{\text{pool}}$ will have non-zero coefficients. This behavior positions LCA as an intermediate approach between EOL-PGD and ARC: in LCA, the coefficients initially resemble those in ARC, but as the pool grows, they transition to resemble the denser pattern of EOL-PGD. This behavior makes it possible to adapt to different types of adversarial lattice: when classifiers are not simultaneously vulnerable, LCA will behave like ARC, which was created to ensure *effectiveness* for binary linear classifiers, and when they are simultaneously vulnerable, it will behave like EOL-PGD which attacks all classifiers simultaneously.

Our proposed attack is presented in Algorithm 4. As there is no guarantee on the convergence of the attack to $\text{SRH}_m(\mathcal{H}_{\text{pool}}, \cdot, y)$ in line 4, we change the criteria to update the pool of classifiers: each time we find a perturbation δ with a higher error for the mixture, we keep it and update the pool to be the classifiers that misclassify the current adversarial example $x + \delta$.

3.2.3 Experiments on synthetic data with linear classifiers

In this section, we show the experimental results that support the theoretical guarantees of LCA in the binary linear setting. We also compare the performance of LCA with ARC and EOL-PGD in the multi-class differentiable setting, in which guarantees are not provided, and show that LCA is competitive and even outperforms the state-of-the-art attacks. More details on the experimental setup and additional experiments can be found in Appendix C.

Binary linear classifiers in high dimension. In this experiment, we assess how the performance of EOL-PGD, ARC and LCA scale with the number of classifiers m in a higher dimension d . For each value m and a fixed value of $\epsilon = 1$, we repeat 1000 times the following experiment: we fix our point $(x, y) = (0_d, -1)$ and sample m i.i.d linear classifiers (w_i, b_i) where w_i is uniformly sampled from the unit sphere in $\mathbb{R}^d$ and $b_i \sim -|\mathcal{N}(\alpha, \beta)|$ follows a folded Gaussian distribution. This choice for b_i is motivated by the fact that we want all classifiers to correctly predict the class -1 for the point x . For each sampling of the mixture of linear classifiers, we test the three attacks. The hyperparameter α controls the expected distance from x to the decision boundary of the classifiers, and β the variance of such distance.

Figure 3.12 shows, on the left, the average of the zero-one loss that each attack was able to achieve as a function of the number of models. The colored regions around each plot represent a 99% confidence interval for the average. As we can see, the average performance of all attacks deteriorates as the number of models increases, but LCA remains superior for all the values of m . It is worth noting that depending on the difficulty of the configurations (α and β), the performance of the attacks can change drastically. Nevertheless, LCA remained superior in all configurations that we tested, regardless of the dimension. More combinations of α and β can be found in Appendix C.1. The right figure shows the average time that each attack took to generate an adversarial example as a function of the number of classifiers. ARC is particularly fast as it only requires one gradient step per classifier, while EOL-PGD and LCA require multiple ones. The number of gradient steps of EOL-PGD was set to be m times the one used for the inner PGD of LCA, in order to avoid a computational advantage for LCA. It can be seen that this choice makes LCA faster than EOL-PGD for all values of m , while keeping it strictly more performant.

3.2.4 Experiments on CIFAR-10

Experimental setup. To test our attack against differentiable multi-class classifiers, we measure the accuracy of mixtures built using models trained with different ensemble diversity techniques. To have a wide variety of models and training methods, we take the pre-trained models from the DVERGE³ [Yan+20] repository, which compares baseline models trained without any defense (BASE) against ensemble diversity defenses such as ADP [Pan+19], GAL [KQ19], plain adversarial training (AT) [Mad+18] and DVERGE (DV) [Yan+20]. The authors made available 3 independent runs of each method (except for BASE), and we report the average accuracy over these 3 independent runs. We also trained a mixture of 5 models using the MRBoost [Zha+22] framework, using the ResNet20 and the ResNet18 architectures [He+16]. All these methods are designed to promote ensemble diversity and reduce the joint

³<https://github.com/zjysteven/DVERGE/tree/main>

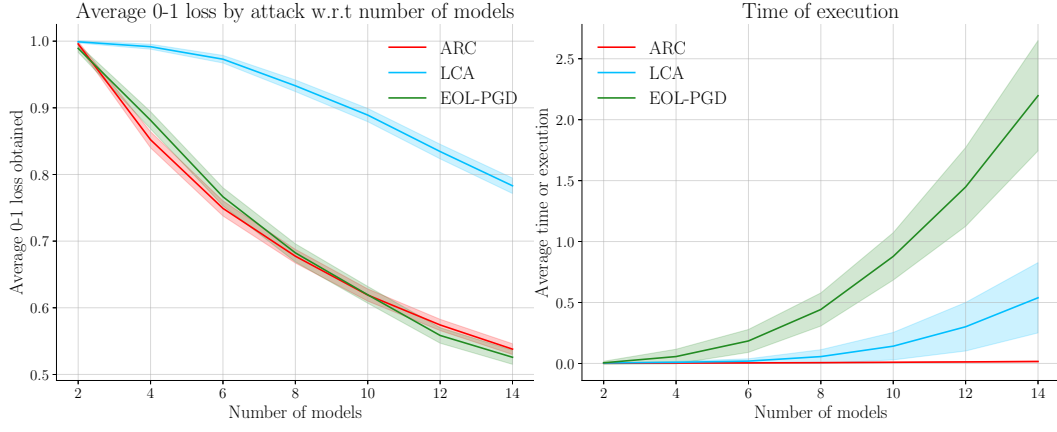


Fig. 3.12.: On the left figure, the average error obtained by the attacks in $\mathbb{R}^{128}$ as a function of the number of classifiers in the mixture with 99% confidence intervals (colored area around the average). The right figure shows the average time in seconds that each attack took to generate an adversarial example as a function of the number of classifiers with one standard deviation intervals. For each number of models, the experiment was repeated 1000 times. Note that the y -axis limits have been adapted for visualization purposes. In this case, $\alpha = 0.25$, $\beta = 0.2$. Best viewed in color.

adversarial vulnerability of the sub-models, so we consider them appropriate to evaluate the performance of attacks against mixtures. We also take pre-trained models from the BARRE⁴ [DS23] repository, in which authors propose a boosting algorithm to specifically train robust mixtures. All these models use the ResNet20 architecture as the base classifiers, except for MRBoost with ResNet18. We use the ℓ_∞ threat model with $\epsilon = 0.03$ as in standard practice, and also the $\epsilon = 0.01$ setting to compare with [Yan+20].

In order to provide a fair comparison, we adjusted the number of iterations T for each attack to make sure that LCA was not given any advantage. Note that the number of gradient computations is $T \cdot m$ for EOL-PGD, at most $4 T \cdot m$ for ARC [DS22], and at most $T \cdot \frac{m(m+1)}{2}$ for LCA. In our case $m \in \{3, 5, 8\}$. Taking this into account, we set the number of iterations to 50 for LCA, and gave ARC and EOL-PGD up to 200 and 500 iterations respectively. We also tested ARC and EOL-PGD with fewer iterations, and report only the best result. We further give more advantage to EOL-PGD by allowing it to perform random initialization and 5 restarts. In contrast, ARC and LCA do not use random initialization nor restarts.

Results. Table 3.1 shows the results of our evaluation. Firstly, we can confirm the observations made by Dbouk and Shanbhag [DS22]: EOL-PGD, even with the advantage given, tends to overestimate the robustness of mixtures, and it is more notorious for BASE models with $\epsilon = 0.01$ and DV models. In these cases, ARC dramatically outperforms EOL-PGD (as was seen in [DS22]), and LCA dramatically

⁴<https://github.com/hsndbk4/BARRE/tree/main>

Tab. 3.1.: Expected accuracy of different mixtures against EOL-PGD, ARC and LCA under the ℓ_∞ threat model. Attacker want to minimize accuracy, so lower is better. All models use ResNet20 as the base architecture, except $\dagger$ which uses ResNet18.

		$\epsilon = 0.01$			$\epsilon = 0.03$		
Model	Nat	EOL	ARC	LCA	EOL	ARC	LCA
BASE/3	91.8%	15.0%	4.3%	1.7%	0.1%	0.2%	0.0%
BASE/5	91.9%	12.3%	6.1%	2.7%	0.0%	0.2%	0.1%
BASE/8	91.8%	10.6%	7.9%	4.6%	0.0%	0.2%	0.1%
ADP/3	89.4%	15.7%	14.6%	15.1%	0.4%	5.2%	1.8%
ADP/5	[Pan+19]	88.8%	15.4%	16.1%	0.2%	2.5%	1.5%
ADP/8		88.5%	15.2%	18.4%	0.3%	4.2%	3.3%
GAL/3		87.1%	43.0%	18.0%	14.3%	14.5%	5.8%
GAL/5	[KQ19]	88.7%	46.2%	45.1%	36.0%	11.5%	9.4%
GAL/8		89.7%	44.5%	52.4%	50.2%	3.9%	11.7%
AT/3		77.3%	68.6%	68.3%	65.3%	47.2%	47.7%
AT/5	[Mad+18]	77.9%	69.0%	69.1%	65.2%	47.2%	49.2%
AT/8		77.8%	69.0%	69.0%	64.9%	47.5%	50.3%
DV/3		89.8%	52.6%	44.4%	31.2%	39.2%	5.8%
DV/5	[Yan+20]	89.9%	59.0%	55.1%	42.1%	35.3%	10.1%
DV/8		88.6%	62.1%	63.2%	52.8%	31.7%	18.4%
DV+AT/3		81.7%	71.4%	71.3%	69.0%	44.4%	45.2%
DV+AT/5	[Yan+20]	84.0%	72.8%	73.1%	70.2%	42.2%	44.6%
DV+AT/8		84.0%	73.2%	73.4%	70.9%	44.3%	46.5%
BARRE/5	[DS23]	76.3%	68.9%	69.4%	66.2%	50.6%	49.9%
MR/5	[Zha+22]	73.7%	65.7%	66.4%	62.3%	46.3%	46.7%
MR/5 †	[Zha+22]	81.3%	72.7%	74.1%	70.2%	47.1%	51.4%

outperforms both ARC and EOL-PGD. Note however that EOL-PGD is better than both ARC and LCA when models are simultaneously vulnerable and the individual gradients used for the attack are highly aligned [KQ19; DS22], which seems to be the case for ADP models. This could be explained by the fact that ADP, contrary to methods like GAL, does not explicitly reduce gradient alignment during training.

GAL models show a peculiar behavior: robustness against EOL-PGD tends to *decrease* as the number of models increase, suggesting that the models become more aligned. A similar behavior was also reported by Yang et al. [Yan+20, Table 4], and it suggests that this method does not scale well with the number of models. However, robustness against ARC or LCA increases with the number of models. This contrast might suggest that GAL models are indeed more diverse locally, but still simultaneously vulnerable inside the ϵ -ball so that EOL-PGD is able to exploit this vulnerability by naively attacking all classifiers enough times.

For the more robust models (AT, DV+AT, BARRE and MR), LCA shows a clear improvement in performance when compared to both ARC and EOL-PGD. The average gap between LCA and ARC in the $\epsilon = 0.03$ setting is 4.9%, with a minimum gap of 3.8%.

Adversarial training for mixtures of classifiers. Table 3.1 shows that most diversity inducing methods fall short in the task of producing a robust mixture. The only

methods that do increase robustness against all the tested attacks use some form of adversarial training, either on individual models independently (AT, DV+AT), by using the EOL-PGD attack for the whole mixture (BARRE) or on the ensembles during training (MRBoost). All these methods explicitly or implicitly increase the individual robustness of the individual classifiers. Mixing models that are already robust is a good starting point, but the real goal of a mixture is to outperform the best individual base model, which is only possible if the models are vulnerable but not simultaneously vulnerable on a non-negligible portion of the input set.

An intuitive approach to increase the joint robustness would be to train the mixture in a dependent way, as is done with BARRE [DS23], by crafting an attack that particularly targets the common vulnerability regions (ARC or LCA). We have empirically found that performing adversarial training with such adapted attacks does not improve (and sometimes even decreases) the robustness of the final mixture. More details in Appendix C.3. This behavior was also observed by Dbouk and Shanbhag [DS23, Appendix B.1], where authors end up using EOL-PGD instead of ARC for BARRE training. In traditional adversarial training, a similar phenomenon has also been observed by Wang et al. [Wan+19], where authors identify that attacks of higher quality (better convergence) are better at the later stages of training, and using them at the early stages is not necessary. Our main hypothesis is that obtaining better solutions for the inner maximization problem (adversarial attack) leads to a significantly more complicated outer problem (risk minimization), an idea that is supported by Wang et al. [Wan+19].

Optimality gap: the impact of the number of restarts. Given that PGD has no optimality guarantee in the case of general multi-class differentiable classifiers, we cannot compute the real optimality gap of LCA. We can however compare LCA against a much stronger version of it that tries all the possible orderings of classifiers to perform the lattice climb. Recall from Figure 3.11 that the ordering in which the classifiers are considered determines the path that is used to climb the lattice, and thus the maximal region that is attained. In practice, this ordering is even more important for Algorithm 4, in which the pool of classifiers is adapted depending on the score, and can therefore change in a non-increasing manner.

To have a better understanding of the limits of LCA, we launched LCA with different numbers of random restarts, ranging from 1 (normal LCA) to 10, which means that 10 different random orderings are tested, and at the end of each ordering, only the best perturbation is kept in a greedy way. As an optimality baseline, we launched a brute-force version of LCA, denoted LCA_{all} , that tests all the possible orderings and keeps, at each time, the best perturbation. We used the same mixtures from Table 3.1, but only for the models AT/5, DV+AT/5, BARRE/5 and MR/5 that showed a

good level of robustness. As all these mixtures have 5 sub-models, there is a total of 120 possible orderings tested by LCA_{all} .

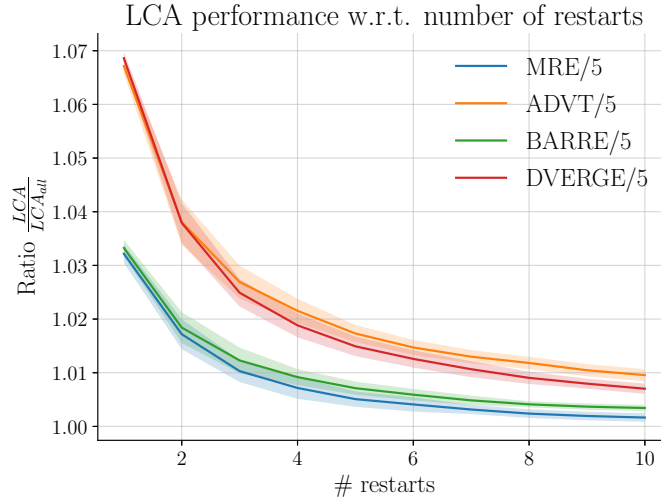


Fig. 3.13.: Impact of restarts in the performance of LCA. The curves show the ratio of the accuracy after attack by LCA and the brute force LCA_{all} (lower the better), averaged over 20 runs with different seeds. Colored regions represent three standard deviations from the average. Best viewed in color.

Figure 3.13 shows the ratio of the robust accuracy obtained by LCA and LCA_{all} as a function of the number of restarts. The reported values are the average over 20 different seeds, and the shaded region around each curve represents three standard deviations from the mean. It can be seen that with 10 restarts, the ratio between LCA and LCA_{all} is of at most 1% which translates in practice to an absolute accuracy gap of at most 0.38% for the model AT/5 that has the highest gap. This experiment also shows that if more computation time is available to afford random restarts, LCA can improve even further with respect to the main results shown in Table 3.1 where it was already superior to ARC and EOL-PGD. In our experiment, with 5 restarts for models AT/5 and DV+AT/5, LCA can decrease the accuracy by an extra 2% with respect to the baseline without restarts. For the models MR/5 and BARRE/5, the improvement is of 1%. Note that all the other parameters for LCA were left unchanged from the settings used for the main results (50 iterations for the inner PGD, no random initialization of the perturbation).

3.3 Discussion

In this chapter, we have discussed several properties that an attack for randomized mixtures should satisfy. We introduced a new attack algorithm called LCA that is *maximal* in the binary linear setting, and also demonstrates good empirical performances against multi-class differentiable classifiers. Regarding the results in

Table 3.1, some diversity inducing techniques like DVERGE [Yan+20] with adversarial training were able to boost the clean accuracy of mixtures while keeping a robust accuracy comparable to plain adversarial training. However, there is no clear improvement in robustness, and our evaluation suggests that the adversarial robustness of mixtures is still mainly a consequence of individual model robustness. Recent work on the training of mixtures also suggests this: when proposing the BARRE method for training mixtures, Dbouk and Shanbhag [DS23] report a robustness of a 5-ResNet20 mixture that is similar to the robustness of a single ResNet20 reported in their previous work [DS22]. Furthermore, while adversarial training represents the current state-of-the-art approach for enhancing robustness, its efficacy is limited in the context of mixtures: using stronger, more adapted attacks for mixtures does not lead to more robust ones. There is a need for new theoretical insights to better train robust mixtures of classifiers and fully capitalize on their theoretical potential.

Randomization in binary classification

Contents

4.1	Any randomized classifier can be matched by a deterministic classifier	64
4.1.1	Decomposing a randomized classifier using the layer-cake representation	64
4.1.2	Bounding the adversarial risk of a randomized classifier with its threshold classifiers	65
4.2	Constructing the deterministic classifier that is at least as robust .	67
4.2.1	Determinizing finite mixtures	67
4.2.2	Determinizing noise injection classifiers	68
4.3	The expressivity growth required to determinize a randomized classifier	69

In this chapter, we will take the defender's perspective and investigate how and when randomization can improve adversarial robustness over deterministic classifiers. In Section 4.1, we will prove Theorem 4.1.3 which states that for binary classification, the robustness of any randomized classifier can be matched by *some* deterministic classifier. In Section 4.2 we will discuss the implications of this result for randomization schemes commonly used in practice, such as finite mixtures and noise injection classifiers, for which we will be able to construct the deterministic classifier ensured by Theorem 4.1.3.

In Section 4.3 we will show that finding such deterministic classifier comes at a cost in terms of complexity of the hypothesis class. For example, when considering finite mixtures over a deterministic hypothesis class $\mathcal{H}$, the new deterministic hypothesis class given by Theorem 4.1.3 is that of weighted ensembles over $\mathcal{H}$, which is a much more expressive class. We will then focus on the case of very expressive hypothesis classes in which this determinization process stabilizes, and show that in such settings, the optimal adversarial risk over deterministic and randomized classifiers are equal. Afterwards, in Chapter 5 we will ask the same question when the number of classes is greater than 2, and discover a stark contrast with the binary case.

Notation. In this chapter, we are going to consider the case of binary classification with labels $\mathcal{Y} = \{0, 1\}$, in which case any distribution over $\mathcal{Y}$ is uniquely determined by a scalar $\nu \in [0, 1]$ that represents the probability of label 1. Recall that a randomized classifier h assigns to every x a distribution over $\mathcal{Y}$. Hence, any randomized binary classifier h can be described as a function $h : \mathcal{X} \rightarrow [0, 1]$. Accordingly, the zero-one loss of both deterministic and randomized classifiers can be simplified as follows:

$$\begin{aligned}\mathcal{L}^{0-1}(h(x), 0) &= h(x) \\ \mathcal{L}^{0-1}(h(x), 1) &= 1 - h(x)\end{aligned}$$

4.1 Any randomized classifier can be matched by a deterministic classifier

In this section we propose a determinization technique that will produce, for any randomized classifier h , a deterministic hypothesis class over which the optimal adversarial risk is at least as good as that of h . This determinization process is based on the well known layer-cake representation of measurable functions, which decomposes any measurable function as a sum of indicator functions. This decomposition will also prove to be useful for decomposing the adversarial risk of a randomized classifier into the adversarial risks of its *threshold classifiers*, which are the deterministic classifiers defined as its level sets.

4.1.1 Decomposing a randomized classifier using the layer-cake representation

Given a measurable function $h : \mathcal{X} \rightarrow [0, 1]$, the *layer-cake representation* [Rud87] of h allows us to write h as a sum of indicator functions. For a randomized classifier, it can be written as:

$$h(x) = \int_0^1 \mathbb{1}\{h(x) > t\} dt. \quad (4.1)$$

The layer-cake representation of a randomized classifier h provides a rewriting of h as an infinite uniform mixture of deterministic classifiers that we call *threshold classifiers*.

Definition 4.1.1 (α -threshold classifier).

For any randomized binary classifier $h : \mathcal{X} \rightarrow [0, 1]$ and threshold $\alpha \in [0, 1]$, we define the α -threshold classifier h^α as $h^\alpha(x) = \mathbb{1}\{h(x) > \alpha\}$.

The threshold hypothesis class of h is given by $\mathcal{H}_T(h) = \{h^\alpha\}_{\alpha \in [0,1]}$.

Given that h is a uniform mixture over $\mathcal{H}_T(h)$, one can immediately check that for any data distribution ρ , the standard risk of h can be rewritten as the average of the standard risks of its threshold classifiers:

$$\mathcal{R}(h, \rho) = \int_0^1 \mathcal{R}(h^\alpha, \rho) d\alpha. \quad (4.2)$$

This result follows directly from the layer-cake representation of h and Fubini's theorem. One natural question that arises is whether the adversarial risk of h can also be decomposed into the adversarial risks of its threshold classifiers. In the next section, we will prove that a similar relation holds, without equality being guaranteed in every case. More specifically, we will see that the adversarial risk of h is lower bounded by the average of the adversarial risks of its threshold classifiers.

4.1.2 Bounding the adversarial risk of a randomized classifier with its threshold classifiers

In order to obtain a similar decomposition to (4.2) for the adversarial risk, we need to account for the supremum operator that appears in the definition of the adversarial risk. The following lemma does exactly that, and it will be a key tool in the decomposition of the adversarial risk of h . Its proof is available in Appendix A.2.1.

Lemma 4.1.2.

Let $h : \mathcal{X} \rightarrow [0, 1]$ be any measurable function. Fix an arbitrary $x \in \mathcal{X}$ and $\alpha \in [0, 1]$. For any operator $\sup$ that can be either $>$ or $\geq$, $\epsilon > 0$ and p -norm with $p > 1$, the following inequality holds.

$$\mathbb{1} \left\{ \left(\sup_{x' \in B_p(x, \epsilon)} h(x') \right) > \alpha \right\} \geq \sup_{x' \in B_p(x, \epsilon)} \mathbb{1} \{h(x') > \alpha\} \quad (4.3)$$

Additionally, the inequality in (4.3) becomes an equality if h takes finitely many values, or if h is continuous.

We can use Lemma 4.1.2 to prove Theorem 4.1.3, which provides a decomposition of the adversarial risk of a randomized binary classifier h in terms of the adversarial risk of its threshold classifiers, and implies that the optimal adversarial risk over the deterministic hypothesis class $\mathcal{H}_T(h)$ is upper bounded by the adversarial risk of h . We provide a sketch of proof and include the complete proof in the supplementary material, Appendix A.2.2.

Theorem 4.1.3.

Let $h : \mathcal{X} \rightarrow [0, 1]$ be any randomized binary classifier. Let h^α be the α -threshold classifier. Fix an arbitrary $\epsilon > 0$ and p -norm, $p > 1$. Then, for any data distribution ρ , the following equation holds:

$$\mathcal{R}_{\epsilon,p}(h, \rho) \geq \int_0^1 \mathcal{R}_{\epsilon,p}(h^\alpha, \rho) d\alpha \geq \inf_{\alpha \in [0,1]} \mathcal{R}_{\epsilon,p}(h^\alpha, \rho). \quad (4.4)$$

Furthermore, if h takes finitely many values, or h is continuous then the first inequality in (4.4) becomes an equality.

Proof Sketch:

We start by rewriting the adversarial risk of h in terms of per-class risk:

$$\mathcal{R}_{\epsilon,p}(h, \rho) = \sum_{y \in \{0,1\}} \int_{\mathcal{X} \times \{y\}} \mathcal{L}_{\epsilon,p}^{0-1}(h(x), y) d\bar{\nu}_y \rho_Y(\{y\}) \quad (4.5)$$

where $\bar{\nu}_y$ is the completion of the conditional $\rho(x|y)$. The completion is necessary to avoid measurability issues [AFM21]. These issues are discussed in detail in the complete proof in Appendix A.2.2. We then use the layer-cake representation for the function $\mathcal{L}_{\epsilon,p}^{0-1}(h(x), y)$. In the case $y = 0$, we obtain that

$$\begin{aligned} \mathcal{L}_{\epsilon,p}^{0-1}(h(x), 0) &= \int_0^1 \mathbb{1}_{\{\mathcal{L}_{\epsilon,p}^{0-1}(h(x), 0) > \alpha\}} d\alpha \\ &= \int_0^1 \mathbb{1}_{\left\{ \sup_{x' \in B_p(x, \epsilon)} h(x') > \alpha \right\}} d\alpha \end{aligned} \quad (4.6)$$

By Lemma 4.1.2, we can interchange the supremum and the indicator function in (4.6), which makes the classifier h^α appear:

$$\begin{aligned} \mathcal{L}_{\epsilon,p}^{0-1}(h(x), 0) &\geq \int_0^1 \sup_{x' \in B_p(x, \epsilon)} \mathbb{1}_{\{h(x') > \alpha\}} d\alpha \\ &\geq \int_0^1 \mathcal{L}_{\epsilon,p}^{0-1}(h^\alpha(x), 0) d\alpha \end{aligned} \quad (4.7)$$

A similar result holds for $y = 1$. Putting (4.7) and the corresponding rewriting for $y = 1$ into (4.5) and using Fubini's theorem, we can rewrite the adversarial risk of h as

$$\mathcal{R}_{\epsilon,p}(h, \rho) \geq \int_0^1 \mathcal{R}_{\epsilon,p}(h^\alpha, \rho) d\alpha. \quad (4.8)$$

The conclusion of the theorem follows from the fact that the RHS of (4.8) is a convex combination, and thus

$$\mathcal{R}_{\epsilon,p}(h, \rho) \geq \int_0^1 \mathcal{R}_{\epsilon,p}(h^\alpha, \rho) d\alpha \geq \inf_{\alpha \in [0,1]} \mathcal{R}_{\epsilon,p}(h^\alpha, \rho).$$

A direct implication of this result, is that for any level of precision $\delta > 0$, there is a deterministic classifier $h^{\alpha_\delta} \in \mathcal{H}_T(h)$ such that $\mathcal{R}_{\epsilon,p}(h^{\alpha_\delta}, \rho) \leq \mathcal{R}_{\epsilon,p}(h, \rho) + \delta$. If the

infimum in the above equation is attained at α_* , then h^{α_*} is guaranteed to be at least as robust as h . In order to better understand the implications of this result, we will look at particular randomized hypothesis classes used in the literature and describe the deterministic counterparts introduced in Theorem 4.1.3.

4.2 Constructing the deterministic classifier that is at least as robust

In this section, we will consider two common randomization schemes: finite mixtures and noise injection classifiers. We will apply Theorem 4.1.3 to these hypothesis classes and discover that the deterministic classifiers ensured to be at least as robust are widely known.

4.2.1 Determinizing finite mixtures

Consider the finite mixture $h_\mu : \mathcal{X} \rightarrow [0, 1]$ built from the set of binary deterministic classifiers $\mathcal{H} = \{h_1, \dots, h_m\}$ with weights $\mu^{(i)} = \mu(\{h_i\})$, $i \in [m]$. The mixture h_μ can be written as the function $h_\mu(x) = \sum_{i=1}^m \mu^{(i)} h_i(x)$. Let us introduce the constant classifier $h_0(x) = 1$ for all x , and $\mu^{(0)} = -\alpha$. Then the α -threshold classifiers for h_μ can be written as

$$h^\alpha(x) = \mathbb{1} \left\{ \sum_{i=1}^m \mu^{(i)} h_i(x) > \alpha \right\} = \mathbb{1} \left\{ \sum_{i=0}^m \mu^{(i)} h_i(x) > 0 \right\}.$$

This shows that the threshold classifier h^α is actually a *weighted ensemble*, such as those that the boosting algorithm can learn [FSA99].

Furthermore, a finite mixture h_μ built with m base binary classifiers is a function that can output at most $b \leq 2^m$ distinct values between 0 and 1, corresponding to all possible partial sums of the weights $\mu^{(1)}, \dots, \mu^{(m)}$. Let $0 = r_1 \leq \dots \leq r_b = 1$ be those possible values. Then, for any $\alpha \in [r_i, r_{i+1})$, $h^\alpha = h^{r_i}$, which implies that the threshold hypothesis set $\mathcal{H}_T(h) = \{h^\alpha\}_{\alpha \in [0,1]}$ is composed of at most $b \leq 2^m$ distinct classifiers, each of which is in turn a weighted ensemble. Applying Theorem 4.1.3 yields,

$$\mathcal{R}_{\epsilon,p}(h_\mu, \rho) = \int_0^1 \mathcal{R}_{\epsilon,p}(h^\alpha, \rho) d\alpha = \sum_{i=1}^{b-1} \int_{r_i}^{r_{i+1}} \mathcal{R}_{\epsilon,p}(h^{r_i}, \rho) d\alpha = \sum_{i=1}^{b-1} (r_{i+1} - r_i) \mathcal{R}_{\epsilon,p}(h^{r_i}, \rho) \quad (4.9)$$

The above equation shows that the risk of the finite mixture h_μ is a convex combination of the risk of at most 2^m weighted ensembles. In case of uniform weights i.e. $\mu^{(m)} = \frac{1}{m}$, there are only $m+1$ distinct weighted ensembles in $\mathcal{H}_T(h_\mu)$ that corre-

spond to the thresholds $\{0, \frac{1}{m}, \dots, 1\}$. Given that $\mathcal{H}_T(h_\mu)$ is finite, the infimum in Theorem 4.1.3 is attained at some α_* , and the weighted ensemble h^{α_*} is guaranteed to be at least as robust as h_μ .

4.2.2 Determinizing noise injection classifiers

Fix a base deterministic binary classifier $h : \mathcal{X} \rightarrow \{0, 1\}$. For any $\eta \in \mathcal{X}$, define a new deterministic classifier by translating h , i.e. $h_\eta(x) = h(x + \eta)$. Let μ be a distribution over the input space $\mathcal{X}$ (typically a Gaussian). The *input noise injection* classifier h_μ is the randomized classifier defined as $h_\mu(x) = \mathbb{P}_{\eta \sim \mu}(h_\eta(x) = 1)$. In this case, the α -threshold classifier takes the following form

$$h^\alpha(x) = \mathbb{1} \left\{ \mathbb{P}_{\eta \sim \mu} (h(x + \eta) = 1) > \alpha \right\}. \quad (4.10)$$

The classifier in (4.10) is very similar to the smoothed classifier created by *randomized smoothing* presented in (2.26) in Section 2.3.2. The classifier in (4.10) generalizes the randomized smoothing classifier in the binary case by replacing the $1/2$ threshold with α . Theorem 4.1.3 suggests that *for any input noise injection classifier, there exists a randomized smoothing classifier with threshold α that is at least as robust*.

Similar to input noise injection classifiers, weight noise injection classifiers [HRF19] are built by injecting noise to the weights of a fixed parametric deterministic classifier. For example, one can consider the set of all neural network classifiers with a fixed architecture with weights in the set $\mathcal{W} \subseteq \mathbb{R}^p$. In such case, for any fixed weight vector $w_0 \in \mathcal{W}$ and any distribution μ over $\mathcal{W}$, we can define the randomized classifier h_μ as $h_\mu(x) = \mathbb{P}_{z \sim \mu}(h_{w_0+z}(x) = 1)$. For the deterministic counterpart, the α -threshold classifier h^α is then defined as

$$h^\alpha(x) = \mathbb{1} \left\{ \mathbb{P}_{z \sim \mu} (h_{w_0+z}(x) = 1) > \alpha \right\}, \quad (4.11)$$

which could be thought of as an infinite majority vote ensemble. Theorem 4.1.3 then suggests that there is an electoral threshold α such that the corresponding infinite majority vote ensemble is at least as robust as h_μ .

4.3 The expressivity growth required to determinize a randomized classifier

In Section 4.1, we saw that for any randomized binary classifier h , there exists a deterministic binary classifier $h^{\alpha*}$ that is at least as robust as h . This result becomes interesting when applied to randomized classifiers that are built from a deterministic hypothesis class $\mathcal{H}$, because it provides a new deterministic hypothesis class $\mathcal{H}'$ of interest. For example, we saw in Section 4.2 that for finite mixtures over $\mathcal{H}$, the hypothesis class $\mathcal{H}'$ is composed of weighted ensembles over $\mathcal{H}$. This already shows that the determinization process provided in Theorem 4.1.3 can lead to a more expressive hypothesis class.

In Appendix A.3 we develop an example with a data distribution ρ supported over four points in $\mathbb{R}^2$ in which, for the set $\mathcal{H}_0$ of all linear classifiers, the set $\mathcal{H}_1$ of finite mixtures over $\mathcal{H}_0$ contained a finite mixture with strictly lower adversarial risk. However, Theorem 4.1.3 applied to this example says that weighted ensembles of linear classifiers ($\mathcal{H}_2$) are at least as robust as finite mixtures. Consequently, the following relationship holds for this example:

$$\mathcal{R}_{\epsilon,p}(\mathcal{H}_0, \rho) > \mathcal{R}_{\epsilon,p}(\mathcal{H}_1, \rho) \geq \mathcal{R}_{\epsilon,p}(\mathcal{H}_2, \rho). \quad (4.12)$$

This naturally raises the question of whether finite mixtures over weighted ensembles could further improve robustness, but even if this was the case, Theorem 4.1.3 would direct us towards weighted ensembles of weighted ensembles! This would create a sequence of hypothesis classes $\mathcal{H}_i$ with non-increasing optimal adversarial risk

$$\mathcal{R}_{\epsilon,p}(\mathcal{H}_0, \rho) > \mathcal{R}_{\epsilon,p}(\mathcal{H}_1, \rho) \geq \mathcal{R}_{\epsilon,p}(\mathcal{H}_2, \rho) > \mathcal{R}_{\epsilon,p}(\mathcal{H}_3, \rho) \geq \dots \quad (4.13)$$

In this section we discuss a particular case in which this process stabilizes, which is when the base hypothesis class is very expressive, like the family of all measurable functions.

Let $\mathcal{X}$ be equipped with an arbitrary σ -algebra $\sigma(\mathcal{X})$ and $\mathcal{Y} = \{0, 1\}$. Consider $[0, 1]$ as a Borel space. In theoretical works like that of Pinot et al. [Pin+20], Awasthi et al. [AFM21], Pydi and Jog [PJ21] or Trillos et al. [TJK23], it is common practice to consider very general hypothesis classes like the set of all continuous or measurable functions. In the case of binary classification, any measurable deterministic classifier h can be identified with the measurable set $h^{-1}(1)$ of points predicted as label 1. Thus, we can identify the set of all measurable deterministic classifiers $\mathcal{H}_{\text{det}}$ with the σ -algebra $\sigma(\mathcal{X})$.

For a randomized classifier defined as a σ -measurable function $h : \mathcal{X} \rightarrow [0, 1]$, all of the h^α threshold classifiers are level sets of h , which are by definition measurable sets. Thus, all the h^α threshold classifiers used in Theorem 4.1.3 belong to $\mathcal{H}_{\text{det}}$. In other words, when considering a large hypothesis class like $\mathcal{H}_{\text{det}}$, randomization cannot provide any robustness improvement. We can formalize this in the following corollary of Theorem 4.1.3:

Corollary 4.3.1.

Let $\mathcal{X}$ be a measurable space with σ -algebra $\sigma(\mathcal{X})$ and $\mathcal{Y} = \{0, 1\}$. Let

$$\begin{aligned}\mathcal{H}_{\text{det}} &= \{x \mapsto \mathbb{1}\{x \in A\} \mid A \in \sigma(\mathcal{X})\}, \\ \mathcal{H}_{\text{rand}} &= \{h : \mathcal{X} \rightarrow [0, 1] \mid h \text{ is } \sigma\text{-measurable}\}\end{aligned}$$

be the family of deterministic and randomized classifiers respectively. Then, for any data distribution ρ , $\epsilon > 0$ and p -norm with $p > 1$, the following holds:

$$\inf_{h \in \mathcal{H}_{\text{rand}}} \mathcal{R}_{\epsilon, p}(h, \rho) = \inf_{h \in \mathcal{H}_{\text{det}}} \mathcal{R}_{\epsilon, p}(h, \rho). \quad (4.14)$$

Proof. Clearly, as $\mathcal{H}_{\text{det}} \subset \mathcal{H}_{\text{rand}}$, we have that

$$\inf_{h \in \mathcal{H}_{\text{rand}}} \mathcal{R}_{\epsilon, p}(h, \rho) \leq \inf_{h \in \mathcal{H}_{\text{det}}} \mathcal{R}_{\epsilon, p}(h, \rho).$$

On the other hand, for any $h \in \mathcal{H}_{\text{rand}}$, by Theorem 4.1.3 we have that

$$\mathcal{R}_{\epsilon, p}(h, \rho) \geq \inf_{\alpha \in [0, 1]} \mathcal{R}_{\epsilon, p}(h^\alpha, \rho) \geq \inf_{h' \in \mathcal{H}_{\text{det}}} \mathcal{R}_{\epsilon, p}(h', \rho),$$

where the last inequality holds because $h^\alpha \in \mathcal{H}_{\text{det}}$ for all $\alpha \in [0, 1]$. As this is valid for an arbitrary $h \in \mathcal{H}_{\text{rand}}$, we have that

$$\inf_{h \in \mathcal{H}_{\text{rand}}} \mathcal{R}_{\epsilon, p}(h, \rho) \geq \inf_{h \in \mathcal{H}_{\text{det}}} \mathcal{R}_{\epsilon, p}(h, \rho).$$

Combining both inequalities, we obtain the desired result.

The fact that randomization is not necessary with such complex hypothesis classes has been discussed in the case of discrete distributions with finite support by Bhagoji et al. [BCM19] and Dai et al. [Dai+24] using arguments from graph theory, invoking the famous König-Egevary theorem. Corollary 4.3.1 generalizes this result, showing that the same conclusion holds for continuous distributions. As a remark, Awasthi et al. [AFM21] and Pydi and Jog [PJ23] prove the existence of a deterministic minimizer of the adversarial risk over the family of Lebesgue measurable classifiers. In such settings, Corollary 4.3.1 implies that these classifiers are also minimizers of the adversarial risk over the set of randomized classifiers (all measurable functions).

One could think that the assumption of having all measurable deterministic classifiers as initial hypothesis class is too strong, and that therefore the fact that randomization cannot provide any improvement is not surprising. However, we will see in the next chapter that there is a marked contrast between the binary and multiclass case.

Lastly, one natural question that arises is whether there exist other hypothesis classes that are not as rich as all measurable classifiers, but for which a similar result to Corollary 4.3.1 holds. This could be an interesting follow up, and we conjecture that if one does not consider a family of deterministic classifiers $\mathcal{H}$ that is a σ -algebra of $\mathcal{X}$, then there must exist a data distribution for which the gap between the optimal adversarial risk over $\mathcal{H}$ and randomized classifiers built from $\mathcal{H}$ is positive.

Randomization in multi-class classification: A graph-theoretic approach

Contents

5.1	Computing the randomization gap using graph theoretical tools	74
5.1.1	Conflict hypergraph, conflict graph	75
5.1.2	Set packing problem and adversarial risk minimization . .	77
5.1.3	Examples of computing the randomization gap using the set packing problem	80
5.2	Link between structural properties of the conflict hypergraph and the randomization gap	82
5.2.1	Decomposition of the randomization gap using the clique hypergraph	82
5.2.2	Necessary structures in the support of the distribution for a positive randomization gap	84
5.3	The randomization gap can be arbitrarily close to $1/2$	86
5.4	Discussion	89

In the last chapter, we focused on binary classification and showed that even if mixtures of classifiers (and thus randomized classifiers) can sometimes provide robustness improvements over a base deterministic hypothesis class, there is always *some* deterministic classifier that is at least as robust as any randomized classifier. This negative result allowed us to conclude, in Corollary 4.3.1, that when the hypothesis class is very large (*i.e.* all measurable deterministic classifiers), randomized classifiers were of no use in improving robustness to adversarial attacks.

The immediate question that arises is whether this result extends to multi-class classification. In this chapter we will show that *this is not the case*: even when considering all measurable deterministic classifiers, there exist data distributions such that the optimal randomized classifier can provide a significant improvement in robustness to adversarial attacks w.r.t. the best deterministic classifier. We will also provide a partial characterization of the data distributions for which this improvement can be achieved, and provide explicit constructions of such data distributions.

One particular thing to note is that the tools used in this chapter are radically different from those used in the previous chapter. In particular, and guided by examples proposed by Trillos et al. [TJK23] and Dai et al. [Dai+24], we will focus on *data distributions with finite support*. We will leverage a recent reformulation of the adversarial risk minimization problem by Dai et al. [Dai+24] to analyze it from a graph theoretical perspective. This reformulation will allow us to establish a connection between the gap in performance between the optimal deterministic and randomized classifiers, and the integrality gap of a combinatorial optimization problem known as the *set packing* problem. This connection will be crucial, as it will allow us to leverage the vast literature on combinatorial optimization to provide a characterization of the data distributions for which this gap is large.

5.1 Computing the randomization gap using graph theoretical tools

In this chapter, we will focus on multi-class classification, *i.e.* $\mathcal{Y} = [K]$ with $K > 2$, and we will consider as hypothesis classes the set of all measurable deterministic and randomized classifiers $\mathcal{H}_{\text{det}}$ and $\mathcal{H}_{\text{rand}}$ respectively. We will further assume that the data distribution ρ is supported on a finite set of n points $s_\rho = \{(x_i, y_i)\}_{i \in [n]}$. We will denote by $\omega_\rho \in \Delta^n$ the probability vector of ρ , *i.e.* $\omega_\rho^{(i)} = \rho(\{(x_i, y_i)\})$ for all $i \in [n]$, and refer to such distributions as *discrete* in a abuse of language.

We are interested in the gap between the optimal error achievable by any deterministic classifier in $\mathcal{H}_{\text{det}}$ and any randomized classifier in $\mathcal{H}_{\text{rand}}$ for a given discrete data distribution ρ . Formally, for a given adversarial budget $\epsilon > 0$ and norm $p \in (1, \infty]$, we define the *randomization gap* of a data distribution ρ as:

$$\text{rg}(\rho) = \mathcal{R}_{\epsilon, p}(\mathcal{H}_{\text{det}}, \rho) - \mathcal{R}_{\epsilon, p}(\mathcal{H}_{\text{rand}}, \rho). \quad (5.1)$$

To compute the randomization gap of a discrete distribution ρ using graph theoretical tools, we will rely on the concept of *conflict hypergraph*, a representation that captures the intrinsic vulnerabilities of the optimal classifier on ρ , first introduced by Dai et al. [Dai+24]. Using this representation, we will map the adversarial risk minimization problem to well studied graph theoretical problems, known as *set packing* and *fractional set packing*. Before diving into the technical details, we will present some graph theoretical terminology.

Graph theoretical terminology. A simple undirected graph $G = (V, E)$ consists of a set of vertices V and a set of edges $E \subseteq \{e \in 2^V \text{ s.t. } |e| = 2\}$ denoting the connections between the vertices. We denote $|G|$ the number of vertices of G . A *hypergraph*

$H = (V, \mathcal{E})$ is a generalization of a graph in which the hyperedges are subsets of V and can contain any number of vertices. We call k -hyperedges all hyperedges that connect exactly k vertices. Note that a hypergraph that only contains 2-hyperedges is a simple graph.

5.1.1 Conflict hypergraph, conflict graph

The notions of *conflict hypergraph* and *conflict graph* are both useful to characterize intrinsic vulnerabilities of a discrete distribution ρ over $\mathcal{X} \times \mathcal{Y}$ to adversarial attacks. They rely on the concept of a *conflict*: two points (x_1, y_1) and (x_2, y_2) from the support of ρ are said to be in conflict if they are both within close range ($\|x_1 - x_2\|_p \leq 2\epsilon$) and belong to different classes ($y_1 \neq y_2$). Informally, configurations of two or more points in conflict are relevant to our problem because the error under attack has to be positive on at least one of these points, thus points involved in conflicts are vulnerable to adversarial attacks regardless of the classifiers that is used. To assess the vulnerability of a distribution ρ , we record all the conflicts as hyperedges in a hypergraph that has a vertex for each element of the support of ρ . However, hypergraphs are sometimes difficult to analyze, and thus it is also useful to consider the conflict graph of ρ , which only records pairwise conflicts between points. The conflict hypergraph, and the conflict graph, can be defined as:

Definition 5.1.1 (Adapted from [Dai+24]).

Let ρ be a discrete distribution over $\mathcal{X} \times \mathcal{Y}$ with support $s_\rho = \{(x_i, y_i)\}_{i \in [n]}$. For any p -norm with $p > 1$ and any $\epsilon > 0$, the conflict hypergraph of ρ at level ϵ is the hypergraph $H_{s_\rho}^\epsilon = (V, \mathcal{E})$ with $V = [n]$ and hyperedge set defined as follows. A set $e \in 2^V$ is a hyperedge of $H_{s_\rho}^\epsilon$ if and only if both the following assertions hold:

1. For any distinct $i, j \in e$ we have $y_i \neq y_j$,
2. $\bigcap_{i \in e} B_p(x_i, \epsilon) \neq \emptyset$ (i.e. the ϵ -balls overlap).

Similarly, the conflict graph is the graph $G_{s_\rho}^\epsilon = (V, E)$ whose edge set is the largest subset of 2^V such that for all $e \in E$, both the previous conditions hold and $|e| = 2$.

The topology of these graphs and hypergraphs depends only on the support of the distribution ρ . Hence, two distributions with the same support will have the same conflict graph and hypergraph. Clearly, the conflict graph¹ captures less information

¹In graph theory, the conflict graph corresponds to the 2-section of the conflict hypergraph (see e.g. [Ber84]), also known as the *skeleton graph*.

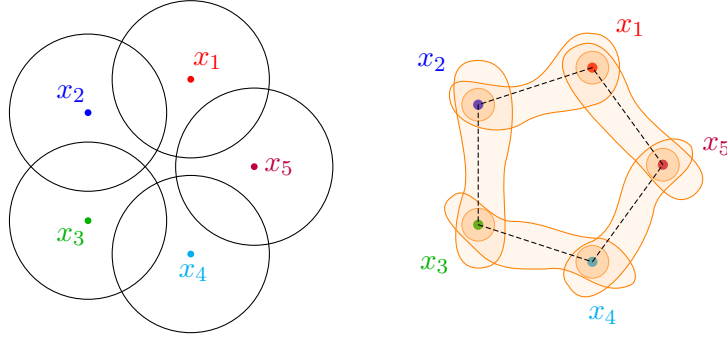


Fig. 5.1

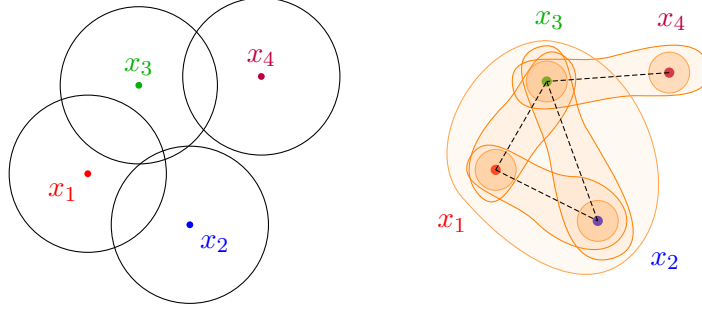


Fig. 5.2

Figs. 5.1, 5.2: On the LHS of each figure, a set S of points of different classes with their ϵ -balls in the ℓ_2 norm. On the RHS, the conflict hypergraph H_S^ϵ represented as orange regions, and the conflict graph G_S^ϵ represented as black dashed lines. For Figure 5.1, the set of hyperedges is $\mathcal{E} = \cup_{i \in [5]} \{\{i\}, \{i, i + 1_{\text{mod} 5}\}\}$, while for Figure 5.2 it is $\mathcal{E} = 2^{[3]} \cup \{\{3, 4\}, \{4\}\}$. Best viewed in color.

about the intrinsic adversarial vulnerability of ρ than the conflict hypergraph. Nevertheless, as we will demonstrate in the subsequent sections, it remains sufficient to identify many of the key structural properties needed for a positive randomization gap to arise.

We present in Figures 5.1 and 5.2 examples of discrete distributions and their corresponding conflict graphs and hypergraphs. In both cases, we assume a uniform distribution over the points of the support, all of which belong to different classes. On the LHS of each figure, we depict the support of the distributions and the ϵ -balls using the ℓ_2 norm. On the RHS of each figure, the dashed lines between the points represent the edges of the conflict graph $G_{s_\rho}^\epsilon$, and the orange regions represent the hyperedges of $H_{s_\rho}^\epsilon$.

Remark 5.1.2.

If no two points in the support of ρ are conflicting, then the conflict hypergraph will only contain singletons. In this case, one can easily show that $\mathcal{R}_{\epsilon,p}(\mathcal{H}_{\text{rand}}, \rho) = \mathcal{R}_{\epsilon,p}(\mathcal{H}_{\text{det}}, \rho) = 0$. See e.g., [Meu+21, Appendix E.4] for a proof when $K = 2$.

5.1.2 Set packing problem and adversarial risk minimization

Using the notion of conflict hypergraph for a given discrete distribution ρ , we now map the adversarial risk minimization problem to a well studied graph theoretical problem known as the *set packing problem* in both its fractional and integer versions.

Set packing problem. Let $H = (V, \mathcal{E})$ be a hypergraph and $\omega \in \mathbb{R}_+^{|V|}$ a weight vector for the vertices. The *set packing problem* over (H, ω) seeks a packing (i.e., a subset of vertices that intersects each hyperedge at most once) with maximum cumulative weight. We represent a packing $Q \subseteq V$ using its characteristic vector $q_Q \in \{0, 1\}^n$, defined component-wise as $q_Q^{(i)} = \mathbb{1}\{i \in Q\}$ for all $i \in [n]$. Then, the set of possible packings for H can be written as

$$\mathcal{Q}(H) = \left\{ q \in \{0, 1\}^n \mid \sum_{i \in e} q^{(i)} \leq 1, \forall e \in \mathcal{E} \right\}, \quad (5.2)$$

and the set packing problem can be formally put as finding a packing with maximal cumulative weight:²

$$\text{IP}(H, \omega) := \max_{q \in \mathcal{Q}(H)} \omega^T q. \quad (5.3)$$

A well known instance of the set packing problem is the *maximum independent set problem* in graphs, where the hypergraph H is a simple graph and the weight vector contains all ones.

Fractional set packing problem. The relaxed version of the set packing problem, called the *fractional set packing problem*, allows the characteristic vector of a packing Q to be fractional, i.e., $q_Q \in [0, 1]^n$. Specifically, given a hypergraph $H = (V, \mathcal{E})$ and weights $\omega \in \mathbb{R}_+^{|V|}$, the fractional set packing problem over (H, ω) consists in finding the *fractional packing* with maximum cumulative weight:

$$\text{FP}(H, \omega) := \max_{q \in \mathcal{Q}^{\text{frac}}(H)} \omega^T q, \quad (5.4)$$

where $\mathcal{Q}^{\text{frac}}(H) = \left\{ q \in [0, 1]^n \mid \sum_{i \in e} q^{(i)} \leq 1, \forall e \in \mathcal{E} \right\}$.

Equivalence between (fractional) set packing and adversarial risk minimization. It is possible to map the adversarial risk minimization problem for deterministic and randomized classifiers to the set packing and fractional set packing problems, respectively. The first step to do so is a construction, originally presented by Dai et al. [Dai+24], that links the solutions from one problem to the other. For the following

²The set packing problem is traditionally defined with a weight vector $\omega_\rho = \mathbf{1}_n$ (see e.g. [Sch79]). For simplicity, we use the name *set packing problem* even though we considered the weighted version in this thesis.

lemma in which the construction is presented, it will be easier to work with the adversarial accuracy instead of the adversarial risk. These two quantities are related by the following equation:

$$\mathcal{A}_{\epsilon,p}(h, \rho) = 1 - \mathcal{R}_{\epsilon,p}(h, \rho). \quad (5.5)$$

More importantly, the adversarial accuracy of a classifier in the case of a discrete distribution ρ can be rewritten as the following finite sum:

$$\mathcal{A}_{\epsilon,p}(h, \rho) = \sum_{i \in [n]} \omega_{\rho}^{(i)} \cdot \inf_{x' \in B_p(x_i, \epsilon)} h(x')^{(y_i)}, \quad (5.6)$$

where the points $\{(x_i, y_i)\}_{i \in [n]}$ are the support of the distribution ρ . The proof of Lemma 5.1.3 is available in Appendix B.1.1.

Lemma 5.1.3 (Extended from [Dai+24]).

Let us consider a p -norm with $p > 1$, and $\epsilon > 0$. Let ρ be an arbitrary discrete distribution over $\mathcal{X} \times \mathcal{Y}$ with support $s_{\rho} = \{(x_i, y_i)\}_{i \in [n]}$ and probability vector $\omega_{\rho} \in \Delta^n$. The following statements hold true:

- a) For any fractional packing $q \in \mathcal{Q}^{\text{frac}}(H_{s_{\rho}}^{\epsilon})$, there exists a classifier $h_q \in \mathcal{H}_{\text{rand}}$ such that $\mathcal{A}_{\epsilon,p}(h_q, \rho) \geq \omega_{\rho}^T q$.
- b) For any classifier $h \in \mathcal{H}_{\text{rand}}$, there exists a fractional packing $q_h \in \mathcal{Q}^{\text{frac}}(H_{s_{\rho}}^{\epsilon})$ such that $\mathcal{A}_{\epsilon,p}(h, \rho) = \omega_{\rho}^T q_h$.

Proof Sketch:.

The proof is based on the construction of a fractional packing q from a randomized classifier h and vice versa.

First, for any point $x \in \mathcal{X}$ and label $y \in \mathcal{Y}$, define the set $\mathcal{I}(x, y)$ that contains the indices of the points in the support of ρ that are ϵ -close to x and have label y . Formally, $\mathcal{I}(x, y)$ is defined as follows:

$$\mathcal{I}(x, y) := \{i \in [n] \mid x \in B_p(x_i, \epsilon) \text{ and } y_i = y\}.$$

Given a fractional packing q , the classifier h_q is defined as follows:

$$\forall x \in \mathcal{X}, \quad h_q(x)^{(y)} = \begin{cases} 1 - \sum_{k > 1} g_q(x)^{(k)} & \text{if } y = 1, \\ g_q(x)^{(y)}, & \text{if } y > 1, \end{cases}$$

where

$$\forall x \in \mathcal{X}, \quad g_q(x)^{(y)} = \begin{cases} \max_{i \in \mathcal{I}(x, y)} q^{(i)}, & \text{if } \mathcal{I}(x, y) \neq \emptyset. \\ 0, & \text{otherwise.} \end{cases}$$

On the other hand, for an arbitrary classifier $h \in \mathcal{H}_{\text{rand}}$, define the fractional packing q_h as follows:

$$\forall i \in [n], \quad q_h^{(i)} = \inf_{x' \in B_p(x_i, \epsilon)} h(x')^{(y_i)}. \quad (5.7)$$

These two constructions will be used to prove the two statements of the lemma. The most important thing to notice for the following is that if h is deterministic, then q_h is a packing, and if q is a packing, then h_q is a deterministic classifier.

Using the construction presented in the proof of Lemma 5.1.3, we can now show that the fractional set packing problem is equivalent to the adversarial risk minimization problem for randomized classifiers. This result is essentially a byproduct of [Dai+24, Corollary 1], and we restate it for completeness. The proof is available in Appendix B.1.2.

Corollary 5.1.4.

Let us consider a p -norm with $p > 1$, and $\epsilon > 0$. For any discrete distribution ρ over $\mathcal{X} \times \mathcal{Y}$ we have:

$$1 - \mathcal{R}_{\epsilon, p}(\mathcal{H}_{\text{rand}}, \rho) = \text{FP}(H_{s_\rho}^\epsilon, \omega_\rho).$$

Corollary 5.1.4 provides a way to compute the optimal adversarial risk for randomized classifiers using linear programming, which can be done efficiently. In Corollary 5.1.5 below, we map the optimal adversarial risk for deterministic classifiers on a discrete distribution ρ over $\mathcal{X} \times \mathcal{Y}$ to the value of the set packing problem on the conflict hypergraph of ρ . The proof is available in Appendix B.1.3.

Corollary 5.1.5.

Let us consider a p -norm with $p > 1$, and $\epsilon > 0$. For any discrete distribution ρ over $\mathcal{X} \times \mathcal{Y}$ we have:

$$1 - \mathcal{R}_{\epsilon, p}(\mathcal{H}_{\text{det}}, \rho) = \text{IP}(H_{s_\rho}^\epsilon, \omega_\rho).$$

Corollary 5.1.5 allows us to compute the optimal adversarial risk for deterministic classifiers using the set packing problem, which can be expressed as an integer linear program and for which powerful open-source solvers are available [Gur24]. In simple cases, such as the examples shown in Figures 5.1 and 5.2, the set packing problem can be manually solved, as we present in Section 5.1.3. For now, we can conclude with a corollary that uses both Corollaries 5.1.4 and 5.1.5 to rewrite the randomization gap of a discrete distribution when considering all measurable randomized and deterministic classifiers:

Corollary 5.1.6.

Let us consider a p -norm with $p > 1$, and $\epsilon > 0$. For any discrete distribution ρ over $\mathcal{X} \times \mathcal{Y}$ we have:

$$\text{rg}(\rho) = \text{FP}(H_{s_\rho}^\epsilon, \omega_\rho) - \text{IP}(H_{s_\rho}^\epsilon, \omega_\rho). \quad (5.8)$$

5.1.3 Examples of computing the randomization gap using the set packing problem

Example presented in Figure 5.1. The fractional set packing problem for this case can be written as the following linear program

$$\begin{aligned} \max_{q \in [0,1]^5} \quad & \frac{1}{5} \mathbf{1}_5^T q \\ \text{s.t.} \quad & Bq \leq \mathbf{1}_{10} \end{aligned} \quad (5.9)$$

where $B \in \{0,1\}^{10 \times 5}$ is the edge-incidence matrix of $H_{s_\rho}^\epsilon$. This is because the set of vectors that satisfy the constraints in (5.9) are exactly the fractional packings of $H_{s_\rho}^\epsilon$. Note that as the largest hyperedges of $H_{s_\rho}^\epsilon$ have size 2, every row in B has at most two components equal to 1. This implies that the fractional packing with characteristic vector $q = \frac{1}{2} \mathbf{1}_5$ is feasible, i.e., $q \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)$, and it has a cumulative weight of $1/2$. Therefore, $\text{FP}(H_{s_\rho}^\epsilon, \omega_\rho)$ is lower bounded by $1/2$. The dual problem of $\text{FP}(H_{s_\rho}^\epsilon, \omega_\rho)$ is the following linear program:

$$\begin{aligned} \min_{z \in [0,1]^{10}} \quad & \mathbf{1}_{10}^T z \\ \text{s.t.} \quad & B^T z \geq \frac{1}{5} \mathbf{1}_5 \end{aligned} \quad (5.10)$$

The vector z can be interpreted as assigning weights to the hyperedges of $H_{s_\rho}^\epsilon$, while the vector $B^T z$ represents the total cumulative weight assigned to the vertices of $H_{s_\rho}^\epsilon$. Consequently, each constraint in (5.10) can be understood as a requirement that each vertex be covered by hyperedges with a cumulative weight of at least $1/5$.

Consider the vector z^* , where the components corresponding to the hyperedges associated with singletons are set to 0, and those corresponding to hyperedges of size 2 are set to $1/10$. As a result, z^* consists of five components equal to 0 and five components equal to $1/10$.

To verify that z^* is feasible, observe that each vertex is contained in exactly two hyperedges of size 2. Since all these hyperedges have a weight of $1/10$ in z^* , we can confirm that every vertex has a total cumulative weight of exactly $1/5$. The associated value of z^* in (5.10) is $1/2$. Therefore, the dual problem is upper bounded by $1/2$, and by duality, we conclude that $\text{FP}(H_{s_\rho}^\epsilon, \omega_\rho) = 1/2$.

When considering the packings of $H_{s_\rho}^\epsilon$ in Figure 5.1 to solve the set packing problem, we can see that including the point x_1 in the packing automatically excludes two points: x_2 and x_5 . This same reasoning applies for all points, which leads to the conclusion that a packing of $H_{s_\rho}^\epsilon$ has size at most 2. We conclude that $\text{IP}(H_{s_\rho}^\epsilon, \frac{1}{5}\mathbf{1}_5) = 2/5$. Therefore, the optimal adversarial risk for deterministic classifiers for the distribution presented in Figure 5.1 is $3/5$.

Example presented in Figure 5.2. Similarly, the fractional set packing problem for this case can be written as the following linear program:

$$\begin{aligned} \max_{q \in [0,1]^4} \quad & \frac{1}{4}\mathbf{1}_4^T q \\ \text{s.t.} \quad & Bq \leq \mathbf{1}_9 \end{aligned}$$

where $B \in \{0,1\}^{9 \times 4}$ is the edge-incidence matrix of $H_{s_\rho}^\epsilon$. In this case, the fractional packing with characteristic vector $q = (1/2, 1/2, 0, 1)$ is feasible, and it has a cumulative weight of $1/2$. Therefore, $\text{FP}(H_{s_\rho}^\epsilon, \omega_\rho)$ is lower bounded by $1/2$.

The dual problem of $\text{FP}(H_{s_\rho}^\epsilon, \omega_\rho)$ is the following linear program:

$$\begin{aligned} \min_{z \in [0,1]^9} \quad & \mathbf{1}_9^T z \\ \text{s.t.} \quad & B^T z \geq \frac{1}{4}\mathbf{1}_4 \end{aligned} \tag{5.11}$$

Consider the vector z^* that is all zeros, except for the component corresponding to the 3-hyperedge $\{1, 2, 3\}$ and the hyperedge $\{4\}$, both of which have value $\frac{1}{4}$. To verify that z^* is feasible, observe that the 3-hyperedge $\{1, 2, 3\}$ covers these three vertices with a weight of $\frac{1}{4}$. Additionally, the vertex $\{4\}$ is covered by its own hyperedge, so all vertices are properly covered. The associated value of z^* in (5.11) is $\frac{1}{4} \cdot 2 = \frac{1}{2}$. Therefore, the dual problem is upper bounded by $\frac{1}{2}$, and by duality, we conclude that $\text{FP}(H_{s_\rho}^\epsilon, \omega_\rho) = \frac{1}{2}$.

Regarding the set packing problem, one can easily check that one maximal packing for the situation presented in Figure 5.2 is $\{x_2, x_4\}$. Thus, $\text{IP}(H_{s_\rho}^\epsilon, \frac{1}{4}\mathbf{1}_4) = 2/4$ and the optimal (deterministic) adversarial risk is $1/2$.

Randomization gap for the examples in Figures 5.1 and 5.2. Since we have already solved both the set packing and fractional set packing problems for the examples in Figures 5.1 and 5.2, we can apply Corollary 5.1.6 to compute the randomization gap for these two cases. For Figure 5.1, we find a positive gap of $1/2 - 2/5 = 1/10$, whereas for Figure 5.2 the randomization gap is 0. The fact that the randomization gap is positive in Figure 5.1 but zero in Figure 5.2 indicates a fundamental difference between the two data distributions. However, it remains unclear how to determine which

distributions may have a positive randomization gap. In the next section, we will characterize the structural properties of discrete distributions that exhibit a positive randomization gap. This is made possible by the reformulation in Corollary 5.1.6, which enables us to characterize these distributions by examining their conflict hypergraph representations.

5.2 Link between structural properties of the conflict hypergraph and the randomization gap

In this section, building upon Corollary 5.1.6, we establish necessary conditions on the conflict hypergraph $H_{s_\rho}^\epsilon$ for a distribution ρ to exhibit a positive randomization gap at level ϵ . We will be able to link a positive randomization gap with the presence of specific structures in the conflict hypergraph and the conflict graph of the distribution. Before diving in, let us introduce some graph theoretical terminology that we will use throughout the remainder of the section.

Different types of induced subgraphs. Let $G = (V, E)$ be an arbitrary graph. An *induced subgraph* of G is a graph $G' = (V', E')$ where

$$V' \subset V, E' = \{\{i, j\} \in E \mid i, j \in V'\}.$$

A *clique* of G is an induced subgraph of G in which every two distinct vertices are adjacent, and is said to be maximal if it cannot be extended by adding a vertex from V to V' . A *hole* of G is an induced subgraph of G consisting of a cycle of more than three vertices. Accordingly, we can define an *anti-hole* of G as an induced subgraph whose complement is a hole in the complement of G . Finally, a graph is considered *perfect* if it contains neither an odd hole nor an odd anti-hole [Chu+03].

5.2.1 Decomposition of the randomization gap using the clique hypergraph

We begin by rewriting the randomization gap as the sum of two non-zero terms. Each term in this decomposition will help us uncover key structural features in the conflict hypergraph. To achieve this, we first need to introduce the concept of *clique hypergraph* that plays a central role in the rewriting.

Definition 5.2.1.

Let us consider a p -norm with $p > 1$, and $\epsilon > 0$. For any discrete distribution ρ over $\mathcal{X} \times \mathcal{Y}$ we define the clique hypergraph of ρ at level ϵ as the hypergraph $C_{s_\rho}^\epsilon = (V, \mathcal{E})$, where $V = [n]$ and $\mathcal{E}$ is the set of all maximal cliques of the conflict graph $G_{s_\rho}^\epsilon$.

Coming back to the examples from Figures 5.1 and 5.2, we can illustrate the concept of a clique hypergraph. Specifically, in Figure 5.1, the clique hypergraph has hyperedge set $\{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 5\}, \{5, 1\}\}$, while in Figure 5.2, the clique hypergraph has hyperedge set $\{\{1, 2, 3\}, \{3, 4\}\}$.

With this notion of clique hypergraph at hand, we establish in Lemma 5.2.2 a relation between the conflict graph, the conflict hypergraph and the clique hypergraph: they are all different formulations of the same set packing problem, with their fractional counterparts providing different upper bounds of varying tightness. The tightest upper bound is given by the fractional set packing problem on the clique hypergraph, followed by the conflict hypergraph, and finally the conflict graph. The proof of Lemma 5.2.2 is available in Appendix B.2.1.

Lemma 5.2.2.

Let us consider an ℓ_p norm with $p \in (1, \infty]$, $\epsilon > 0$, and a discrete distribution ρ over $\mathcal{X} \times \mathcal{Y}$. The following holds:

$$\text{IP}(C_{s_\rho}^\epsilon, \omega_\rho) = \text{IP}(H_{s_\rho}^\epsilon, \omega_\rho) = \text{IP}(G_{s_\rho}^\epsilon, \omega_\rho). \quad (5.12)$$

$$\text{FP}(C_{s_\rho}^\epsilon, \omega_\rho) \leq \text{FP}(H_{s_\rho}^\epsilon, \omega_\rho) \leq \text{FP}(G_{s_\rho}^\epsilon, \omega_\rho). \quad (5.13)$$

Combining Corollary 5.1.6 and Lemma 5.2.2, we can rewrite the randomization gap using the packing problems on the clique hypergraph $C_{s_\rho}^\epsilon$ and the conflict hypergraph $H_{s_\rho}^\epsilon$ of ρ . Then by reintroducing the value of the fractional packing problem on the clique hypergraph, we obtain a particular decomposition of the randomization gap, as follows:

$$\text{rg}(\rho) = \text{FP}(H_{s_\rho}^\epsilon, \omega_\rho) - \text{FP}(C_{s_\rho}^\epsilon, \omega_\rho) \quad (5.14)$$

$$+ \text{FP}(C_{s_\rho}^\epsilon, \omega_\rho) - \text{IP}(C_{s_\rho}^\epsilon, \omega_\rho). \quad (5.15)$$

The above decomposition holds true as $\text{IP}(H_{s_\rho}^\epsilon, \omega_\rho)$ can be replaced by $\text{IP}(C_{s_\rho}^\epsilon, \omega_\rho)$, thanks to (5.12). Furthermore, both (5.14) and (5.15) are non-negative. Indeed, (5.14) is non-negative due to (5.13) and (5.15) is non-negative because the fractional set packing problem is a relaxation of the set packing problem, and thus its value is always an upper bound of the corresponding set packing problem. By studying the conditions under which each one of these terms is positive, we can

identify the necessary conditions on the conflict hypergraph $H_{s_\rho}^\epsilon$ for ρ to exhibit a positive randomization gap.

5.2.2 Necessary structures in the support of the distribution for a positive randomization gap

In order to identify the geometrical structures in the support of a discrete distribution that can create a positive randomization gap, we will make use of the *strong perfect graph theorem* proved by Chudnovsky et al. [Chu+03]. More specifically, we are going to establish a relation between this strong result in combinatorial optimization that has to do with the integrality gap of the set packing problem, and our problem regarding the randomization gap in the scenario of adversarial risk minimization for discrete distributions. To be more precise, Lemmas 5.2.3 and 5.2.4 are adapted from the seminal work of Chudnovsky et al. [Chu+03, Theorem 4.1] and are the fundamental blocks to prove Theorem 5.2.5. Each lemma translates into an equivalence between the positivity of (5.14) or (5.15), and the existence of a particular structure in the conflict hypergraph.

First of all, note that (5.14) becomes zero if the conflict hypergraph $H_{s_\rho}^\epsilon$ coincides with the clique hypergraph $C_{s_\rho}^\epsilon$. This occurs when every clique in the conflict graph $G_{s_\rho}^\epsilon$ is a hyperedge in $H_{s_\rho}^\epsilon$; in this case $H_{s_\rho}^\epsilon$ is said to be *conformal* [Ber84]. Hence, for (5.14) to be positive, one needs $H_{s_\rho}^\epsilon$ to be non-conformal. This is formalized in Lemma 5.2.3 (proof available in Appendix B.2.2):

Lemma 5.2.3 (Conformal hypergraphs).

Let us consider a p -norm with $p > 1$, and $\epsilon > 0$. Let also $S = \{(x_i, y_i)\}_{i \in [n]}$ be an arbitrary set of points from $\mathcal{X} \times \mathcal{Y}$. The assertions a) and b) below are equivalent.

- a) $\forall \omega \in \mathbb{R}_+^n, \quad \text{FP}(H_S^\epsilon, \omega) = \text{FP}(C_S^\epsilon, \omega)$
- b) Every clique in G_S^ϵ is a hyperedge in H_S^ϵ .

On the other hand, determining when (5.15) is non-zero is less trivial and involves the characterization of perfect graphs [Chu+03; Con+14]. Specifically, (5.15) is zero for any weight vector if the conflict graph $G_{s_\rho}^\epsilon$ is perfect (proof available in Appendix B.2.3):

Lemma 5.2.4 (Perfect graphs).

Let us consider a p -norm with $p > 1$, and $\epsilon > 0$. Let also $S = \{(x_i, y_i)\}_{i \in [n]}$ be an arbitrary set of points from $\mathcal{X} \times \mathcal{Y}$. The assertions a) and b) below are equivalent.

$$a) \quad \forall \omega \in \mathbb{R}_+^n, \quad \text{FP}(C_S^\epsilon, \omega) = \text{IP}(C_S^\epsilon, \omega).$$

b) G_S^ϵ is perfect.

Taking into account that the properties of being conformal and perfect can be stated in terms of forbidden structures in either the conflict graph or the conflict hypergraph, we can express conditions for the *existence* of a discrete distribution with positive randomization gap as follows (full proof in Appendix B.2.4):

Theorem 5.2.5.

Let us consider a p -norm with $p > 1$, and $\epsilon > 0$. Let also $S = \{(x_i, y_i)\}_{i \in [n]}$ be an arbitrary set of points from $\mathcal{X} \times \mathcal{Y}$. There exists a distribution ρ over $\mathcal{X} \times \mathcal{Y}$ with support $s_\rho = S$ such that $\text{rg}(\rho) > 0$ if and only if at least one of the following assertions holds true:

a) H_S^ϵ is not conformal (i.e., there exists a clique in G_S^ϵ that is not a hyperedge in H_S^ϵ).

b) G_S^ϵ is not perfect (i.e., it contains at least one odd hole or one odd anti-hole).

Theorem 5.2.5 is a strong result providing both necessary and sufficient conditions for the *existence* of a discrete distribution ρ over $\mathcal{X} \times \mathcal{Y}$ with positive randomization gap. However, it does not provide a testable condition for a *given* discrete distribution ρ to determine whether the randomization gap is positive. The following corollary provides (testable) necessary conditions on the support of ρ for a positive randomization gap (proof in).

Corollary 5.2.6.

Let us consider a p -norm with $p > 1$, and $\epsilon > 0$. Let us also consider a discrete distribution ρ over $\mathcal{X} \times \mathcal{Y}$. If $\text{rg}(\rho) > 0$, then $G_{s_\rho}^\epsilon$ satisfies either assertion a) or assertion b) from Theorem 5.2.5.

Hardness of verifying sufficient conditions. Given a distribution ρ , Corollary 5.2.6 identifies a necessary condition for the randomization gap to be positive. Furthermore, this condition is checkable in polynomial time [CLV03; Bor+23]. However, we conjecture that any condition that is both *necessary and sufficient* cannot be checked in polynomial time, unless "P=NP"³. This conjecture is supported by the fact that, for

³It is widely believed that problems solvable in polynomial time (P) are not the same as problems whose solutions are verifiable in polynomial time (NP).

a given discrete distribution ρ and $\alpha \in [0, 1]$, checking if $\text{rg}(\rho) \geq \alpha$ is co-NP-complete. The proof of this result is available in Appendix B.4

Special case of $K = 2$. In binary classification, the conflict graph $G_{s_\rho}^\epsilon$ can be shown to be bipartite. Since any bipartite graph is perfect, we have that (5.15) is always zero. Additionally, the conflict hypergraph $H_{s_\rho}^\epsilon$ is always conformal, as a clique in $G_{s_\rho}^\epsilon$ is simply an edge and thus a hyperedge of $H_{s_\rho}^\epsilon$. This implies that (5.14) is also zero. Combining both arguments, the contrapositive of the implication stated in Corollary 5.2.6 tells us that the randomization gap is always zero in the binary case, which validates our results in Theorem 4.1.3 that were also discussed by Bhagoji et al. [BCM19].

The special case of the ℓ_∞ norm Interestingly, when considering the ℓ_∞ norm, we can prove that the only structures that matter in this case are those related to the perfect graph condition. This is because we can show that the conflict hypergraph $H_{s_\rho}^\epsilon$ is always conformal, as stated in the following corollary (proof in Appendix B.2.5):

Corollary 5.2.7.

Consider the ℓ_∞ norm, $\epsilon > 0$ and a discrete distribution ρ over $\mathcal{X} \times \mathcal{Y}$ with clique hypergraph $C_{s_\rho}^\epsilon = (V, \mathcal{E})$. Let $G' = (V', E')$ be an induced subgraph of $G_{s_\rho}^\epsilon$. If G' is a clique, then $V' \in \mathcal{E}$. Thus, for any $\omega \in \mathbb{R}_+^{|V|}$:

$$\text{FP}(H_{s_\rho}^\epsilon, \omega) = \text{FP}(C_{s_\rho}^\epsilon, \omega).$$

Corollary 5.2.7 shows that the first source of randomization gap in (5.14) is always zero when using the ℓ_∞ norm. This implies that the only way for a distribution to exhibit a positive randomization gap is for the conflict graph $G_{s_\rho}^\epsilon$ to be non-perfect. In contrast, when using the ℓ_2 norm, such uncovered cliques can exist (see Figure 5.4).

5.3 The randomization gap can be arbitrarily close to $1/2$

We now identify a variety of discrete distributions for which we can establish that the randomization gap is arbitrarily close to $1/2$. Specifically, even though the structural conditions introduced in the last section are not sufficient for the randomization gap to be positive, they provide a systematic way to construct distributions with a significantly large randomization gap, thus generalizing the initial results presented in Trillos et al. [TJK23] and Dai et al. [Dai+24].

Large randomization gap based on the conformal condition. We start by designing a simple data distribution for which the randomization gap is close to $1/2$. Let us consider the discrete distribution ρ that is uniformly distributed over the canonical basis of $\mathbb{R}^K$ and in which each vector is assigned to a distinct class. Specifically, let $(b_1, \dots, b_K)$ denote the canonical basis of $\mathbb{R}^K$ and $\mathbf{1}_n$ denote the vector in $\mathbb{R}^n$ with all components to one. We set $s_\rho = \{(b_1, 1), \dots, (b_K, K)\} \subset \mathbb{R}^K \times [K]$ and $\omega_P = \frac{1}{K} \mathbf{1}_K$. Then for $\epsilon = 1/\sqrt{2}$ and considering the ℓ_2 norm, one can verify (see Appendix B.2.6) that the conflict hypergraph $H_{s_\rho}^\epsilon$ of ρ contains all 2-hyperedges between every pair of points, with no hyperedges of larger size. In other words, the conflict hypergraph $H_{s_\rho}^\epsilon$ is a complete graph over K vertices, including self-loops. Furthermore, every set of vertices forms a clique in the conflict graph $G_{s_\rho}^\epsilon$, but none of the cliques of size greater than 2 are hyperedges in $H_{s_\rho}^\epsilon$. Therefore, $H_{s_\rho}^\epsilon$ is not conformal (See Figure 5.4).

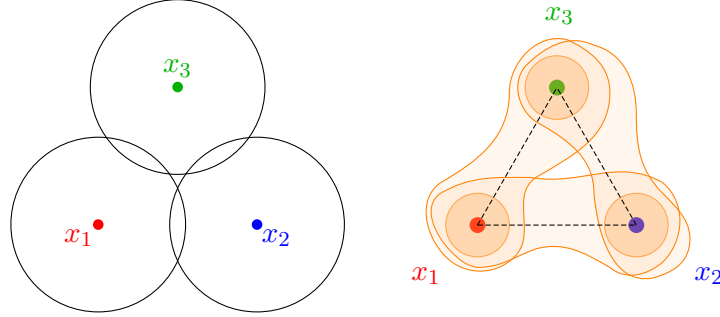


Fig. 5.4.: Example of a set of points with a non-conformal conflict hypergraph. The support is the canonical basis of $\mathbb{R}^3$, $\epsilon = 1.1/\sqrt{2}$, and $p = 2$. The LHS illustrates the points in the space together with their ϵ -balls, and the RHS shows $H_{s_\rho}^\epsilon$, which is not conformal because it does not contain the hyperedge $\{1, 2, 3\}$.

The vector $q = \frac{1}{2} \mathbf{1}_K$ is feasible for the fractional packing problem over $(H_{s_\rho}^\epsilon, \omega_\rho)$, i.e., $q \in \mathcal{Q}(H_{s_\rho}^\epsilon)$, since the conflict hypergraph $H_{s_\rho}^\epsilon$ does not contain any hyperedges of size larger than 2. This implies that $\text{FP}(H_{s_\rho}^\epsilon, \omega_\rho) \geq \omega_\rho^T q = \frac{1}{2}$. Furthermore, $H_{s_\rho}^\epsilon$ contains every possible 2-hyperedge, hence the only possible packings for $H_{s_\rho}^\epsilon$ are the singletons. This implies that $\text{IP}(H_{s_\rho}^\epsilon, \omega_\rho) = \frac{1}{K}$. Combining these two arguments and using Corollary 5.1.6, the randomization gap of ρ is greater than $\frac{1}{2} - \frac{1}{K}$. Therefore, we can make it arbitrarily close to $1/2$ by choosing K large enough.

The construction presented in this subsection generalizes the initial example from Trillos et al. [TJK23] and Dai et al. [Dai+24] by scaling it to an arbitrary number of classes K . Nevertheless, it remains restrictive because the distribution ρ we design is constrained to having one point per class. Additionally, such a construction is not feasible when considering the ℓ_∞ norm, as indicated by Corollary 5.2.7. For these reasons, we seek a procedure that enables the construction of less trivial distributions with a large randomization gap, valid for any ℓ_p -norm.

Large randomization gap based on the perfect graph condition. To design more general distributions, we focus on the perfect graph condition and leverage existing graph constructions from the graph theory literature. Doing so, we obtain the following statement.

Theorem 5.3.1.

Fix any $\epsilon > 0$ and p -norm with $p > 1$. For any $\delta > 0$, there exist $d, K \in \mathbb{N}$ and a discrete distribution ρ over $\mathbb{R}^d \times [K]$ such that

$$\text{rg}(\rho) \geq 1/2 - \delta, \quad (5.16)$$

the conflict graph $G_{s_\rho}^\epsilon$ is not perfect and the conflict hypergraph $H_{s_\rho}^\epsilon$ is conformal. Furthermore, the number of classes satisfies $K \in \mathcal{O}\left(\sqrt{n/\log n}\right)$ with $n = |s_\rho|$.

Theorem 5.3.1 show that there exist non-trivial distributions (with $K \ll n$) for which the randomization gap is arbitrarily close to $1/2$. While Theorem 5.3.1 itself does not provide an example of such distribution, our proof is constructive and thus provide one. We outline below the main idea of this construction, and leave the full proof of Theorem 5.3.1 to Appendix B.3.

Proof technique. The main idea of the proof consists in leveraging existing results in graph theory on the construction of large non-perfect, triangle-free graphs. Specifically, we aim to use the iterative procedure presented by Chung et al. [CCD93] that starts from a triangle-free graph G_0 and creates a sequence of triangle-free graphs $\{G_t\}_{t \in \mathbb{N}}$ such that, for all $t \in \mathbb{N}$,

$$\text{FP}(G_t, \omega_t) - \text{IP}(G_t, \omega_t) \geq \frac{1}{2} - \left(\frac{2}{3}\right)^t |G_0|, \quad (5.17)$$

where $\omega_t = \frac{1}{|G_t|} \mathbf{1}_{|G_t|}$ and $|G_0|$ is the size of G_0 . Note that the RHS of (5.17) is an increasing sequence, and when it becomes positive, the graphs G_t are guaranteed to be non-perfect, i.e. $\text{FP}(G_t, \omega_t) > \text{IP}(G_t, \omega_t)$. This result is arguably close to what we would like to demonstrate. Intuitively, we would like to design a sequence of distributions $\{\rho_t\}_{t \in \mathbb{N}}$ such that, for any $t \in \mathbb{N}$, the LHS of (5.17) can be represented as the randomization gap of ρ_t . To design such a sequence, we first show in Lemma B.3.5 that for any graph G_t , there exists a distribution ρ_t whose conflict graph $G_{s_{\rho_t}}^\epsilon$ is isomorphic to G_t . Second, we show that when G_t is triangle-free we can characterize the randomization gap of ρ_t as

$$\text{rg}(\rho_t) = \text{FP}(G_t, \omega_t) - \text{IP}(G_t, \omega_t), \quad (5.18)$$

which comes from the fact that the conflict graph $G_{s_{\rho_t}}^\epsilon$ is the loopless version of the conflict hypergraph $H_{s_{\rho_t}}^\epsilon$.

With all this components in place, for any $\delta > 0$, one can choose a sufficiently large $t^* \in \mathbb{N}$ such that $(2/3)^{t^*} |G_0| < \delta$. Then, by combining (5.17) and (5.18), the randomization gap of ρ_{t^*} satisfies (5.16). Note that the dependency of K on the size of the support $n = |s_{\rho_{t^*}}|$ can be made explicit using Lemma B.3.5.

5.4 Discussion

In this chapter, we considered the problem of multi-class classification with discrete data distributions. Furthermore, we focused on two particular hypothesis classes: $\mathcal{H}_{\text{det}}$ the set of all measurable deterministic classifiers and $\mathcal{H}_{\text{rand}}$ the set of all measurable randomized classifiers. We introduced the notion of randomization gap, which quantifies the difference in risk between the optimal deterministic and randomized classifiers. Contrary to the binary case, in which the randomization gap is always zero when considering such hypothesis classes, we showed that in the multi-class setting the randomization gap can be positive and arbitrarily close to $1/2$. We provided necessary and sufficient conditions for *existence* of a discrete distribution with a positive randomization gap. For a given data distribution ρ with its conflict graph $G_{s_\rho}^\epsilon$ and hypergraph $H_{s_\rho}^\epsilon$, we can check in polynomial time if the randomization gap is null by testing for conformality of $H_{s_\rho}^\epsilon$ and perfection of $G_{s_\rho}^\epsilon$. However, we proved that checking if $\text{rg}(\rho) > \alpha$ is co-NP-complete.

The tools used to prove these results are mainly from combinatorial optimization and graph theory, and include a reformulation of the adversarial risk minimization problem as a fractional set packing problem, as well as the use of the strong perfect graph theorem. Our constructions of distributions with large randomization gap are based on existing constructions in graph theory of large triangle-free graphs with low independence number (which makes them non-perfect).

While the link between adversarial multi-class classification and graph theory is interesting, it is not clear how to extend these results to continuous data distributions. This is an interesting direction for future work. Another direction for future work is considering less complex hypothesis classes. In this case, however, the reformulation of the adversarial risk minimization problem that is at the root of our analysis does not hold, meaning that new tools and techniques would be required.

Conclusion

This thesis has explored the use of randomization as a theoretically grounded approach to designing robust machine learning models resistant to adversarial attacks. Inspired by game theory, randomization offers a compelling alternative to empirical methods like adversarial training. However, the unique challenges of adversarial machine learning have made the application of randomization both theoretically and practically complex. Through this work, we have made significant contributions to the field, addressing key questions and opening new avenues for future research.

Summary of the contributions:

- **Attacking mixtures of classifiers**

In Chapter 3 we analyzed the problem of attacking randomized mixtures, a straightforward yet underdeveloped application of randomization in adversarial machine learning. Despite the impracticality of deriving an optimal attack algorithm, we identified critical properties that enabled us to improve upon existing methods. Our proposed attack algorithm, LCA (Lattice Climber Attack), reformulates the problem in terms of an adversarial lattice and provides a stronger benchmark for evaluating defense mechanisms based on randomized mixtures. We also explored the limitations of existing attacks and proposed ways to enhance LCA, such as incorporating multiple climbs. These advancements deepen the field's understanding of the robustness of randomized mixtures.

- **The benefits of randomization and expressivity of hypothesis classes**

In Chapter 4 we demonstrated that while it is always possible to match randomized classifiers with deterministic classifiers in binary classification, this comes at a cost in terms of expressivity. Specifically, the deterministic alternative may belong to a much more expressive hypothesis class, which may not be desirable in practice. In Chapter 5 we saw that this result did not generalize to the multiclass setting, because there exist data distributions for which no matter how expressive the hypothesis class, randomization is necessary to achieve optimal adversarial risk. These results highlight the trade-offs involved in using randomization and underscore the need for careful design when applying these techniques in practice.

- **Partial characterization of the benefits of randomization in adversarial multiclass classification**

In Chapter 5, we focused on the benefits of randomization in adversarial multiclass classification with very expressive hypothesis classes. Recent counter examples had shown that the gap between the optimal adversarial risk with randomized and deterministic classifiers could be positive even in the case of very expressive hypothesis classes, without providing any further explanation or intuition. We provided a theoretical explanation for this phenomenon in the case of discrete distributions with finite support by making a connection with well-known graph theoretical problems and tools. This allowed us to partially characterize the discrete data distributions for which randomization is necessary in terms of forbidden structures in the support of the data distribution. Our findings allowed us to exploit these characterization to construct data distributions for which the benefits of randomization are large.

These contributions have significantly strengthened the case for randomization as a viable solution to adversarial machine learning. However, this thesis also reveals numerous open questions that must be addressed to realize the full potential of randomization in practice.

Future directions:

- **Training robust randomized classifiers**

Our experiments in Chapter 3 revealed that existing diversity-inducing techniques fail to increase the robustness of finite mixtures of classifiers. Instead, robustness gains primarily stem from the robustness of individual classifiers. This raises several critical questions:

- *Can we develop methods to effectively increase classifier diversity? Such methods would ensure that individual classifiers are not vulnerable to the same adversarial examples, enabling finite mixtures to provide robustness gains over individual classifiers. This also establishes a link with the literature on ensemble diversity.*
- *Why does adversarial training fail to improve the robustness of finite mixtures? Understanding this phenomenon could provide insights into designing more robust mixtures. For instance, why does adversarial training with adaptive attacks like ARC or LCA fail to improve robustness, while weaker attacks like EOL-PGD succeed?*

- **Theoretical understanding of the benefits of randomization**

Our theoretical work in Chapters 4 and 5 opens several avenues for future research regarding the benefits of randomization in adversarial machine learning.:

- *What is the explicit relationship between hypothesis class complexity and the benefits of randomization? In binary classification, randomization can be an alternative to using a more expressive hypothesis class. However, it is*

not yet clear what is the relationship between the complexity of the starting hypothesis class and the benefits of randomization. Our intuition is that the less expressive the starting hypothesis class, the more likely randomization is to be beneficial. In the multiclass settings, randomization can provide significant performance improvements with highly expressive classes, making a strong point for the use of randomized classifiers. However, it remains unclear how and at what extent do these benefits extend to less expressive classes.

- What can we say when the hypothesis class is composed of deep neural networks with modern architectures? Deep neural networks are one of the most popular choices for modern machine learning tasks. We know that these models are highly expressive, in which case our work shows that the benefits of randomization should be negligible in the binary case, but possibly large in the multiclass case. At the same time, evaluating their robustness in practice is very challenging, so taking out conclusions in practice is not straightforward. Due to their predominance in practice, this particular hypothesis class might deserve special attention.
- Can we generalize our multiclass results to continuous distributions? Our current results in the multi class setting focus on discrete distributions with finite support. Extending these insights to continuous distributions could reveal new theoretical and practical implications. Just as there is a stark difference between the binary and multiclass settings, the continuous case could be surprisingly different from the discrete case. Moreover, given the tools we have used in our work, it is not clear how to generalize to the continuous case. This might require new tools and techniques like optimal transport [TJK23; TJK24].

- **Broader questions regarding randomized classifiers**

Apart from the specific questions that stemmed from our work, there are broader questions that need to be addressed to advance the field of robust randomized classifiers:

- How can we design a benchmark of attacks against randomized classifiers? A comprehensive benchmark, similar to Robustbench [Cro+21], is essential for evaluating the robustness of randomized classifiers and advancing the field. The Robustbench benchmark is based on AutoAttack, an attack suite that was designed to be as comprehensive as possible for deterministic classifiers. Developing a similar benchmark for randomized classifiers would require new attack algorithms that can effectively evaluate the robustness of different types of randomized classifiers.

- Can we effectively train robust randomized classifiers? *While finite mixtures are simple to manipulate, there are many other randomization techniques that have received plenty of attention, such as noise injection or stochastic pruning. However, they remain under-explored compared to deterministic alternatives like adversarial training (with its many variants), diverse ensembles or Lipschitz neural networks. More research is needed to understand how to effectively train robust randomized classifiers that are truly robust against adaptive attacks. Evidently, this will also require the development of stronger attacks, as has been suggested previously.*

Closing remarks This thesis represents a significant step forward in understanding the role of randomization in adversarial machine learning. By addressing key theoretical and practical challenges, we have demonstrated the potential of randomization as a robust defense mechanism. However, much work remains to be done to translate these insights into practical solutions. We hope that this work inspires further research into the theoretical foundations of randomization and its applications, ultimately contributing to the development of more secure and reliable machine learning systems.

Supplementary material for Chapters 3 and 4

Contents

A.1	Proofs of Chapter 3	95
A.1.1	Proof of Theorem 3.1.4	95
A.1.2	Proof of Lemma 3.2.1	96
A.1.3	Proof of Theorem 3.2.2	96
A.2	Proofs of Chapter 4	97
A.2.1	Proof of Lemma 4.1.2	97
A.2.2	Proof of Theorem 4.1.3	98
A.3	Example of mixtures with strictly better adversarial risk	103

A.1 Proofs of Chapter 3

A.1.1 Proof of Theorem 3.1.4

Theorem 3.1.4 (Hardness of attacking linear classifiers).

Consider a binary classification setting, a p -norm with $p > 1$ and $\epsilon > 0$ to be defined. Given a labeled point (x, y) , a set of m linear classifiers $x \mapsto \mathbb{1}\{\theta_i^\top x + b_i \geq 0\}$ where $(\theta_i, b_i) \in \mathbb{R}^{d+1}$, a uniform mixture h composed of these linear classifiers and a value $\beta > 0$, there exists a noise budget $\epsilon > 0$ such that the problem of checking if there exists $\delta \in B_p(0, \epsilon)$ such that $\mathcal{L}^{0-1}(h(x + \delta), y) \geq \beta$ is NP-hard.

Proof. To show NP-hardness of our problem, we will use a reduction involving the NP-hard MaxFLS [AK95] problem defined as follows:

- **Input of MaxFLS:** A set of m pairs $\{(\theta_i, b_i)\}_{i=1}^m$ where $\theta_i \in \mathbb{R}^d$ and $b_i \in \mathbb{R}$, and a value $\alpha > 0$
- **Output of MaxFLS:** Yes if there exists a vector $\delta \in \mathbb{R}^d$ such that at least a fraction α of inequalities among the set $\{\delta^\top \theta_i + b_i \geq 0 : i \in [m]\}$ are satisfied

Let us consider an instance $(\{(\theta_i, b_i)\}_{i=1}^m, \alpha)$ of MaxFLS. The set $\{(\theta_i, b_i)\}_{i=1}^m$ defines an arrangement of hyper-planes in $\mathbb{R}^d$ which can be thought as linear classifiers, and this arrangement partitions the space $\mathbb{R}^d$ into regions. Let $x = 0$. Clearly, we can

always find a value ϵ such that each region has a non-empty intersection with the ball $B_p(x, \epsilon)$. Let $y = 0$ and $\mu = (\frac{1}{m} \dots \frac{1}{m})$, and define h_μ as the mixture of the following binary classifiers

$$h_i(x) = \mathbb{1} \{ \theta_i^\top x + b_i \geq 0 \}, \text{ for } i \in [m],$$

with weights μ . Then, $\ell^{0-1}(h_\mu, x + \delta, y) = \frac{1}{m} \sum_{i \in [m]} \mathbb{1} \{ \delta^\top \theta_i + b_i \geq 0 \}$. Clearly, the value $\ell^{0-1}(h_\mu, x + \delta, y)$ is equal to the fraction of satisfied inequalities among the set $\{ \delta^\top \theta_i + b_i \geq 0 : i \in [m] \}$. Thus, $\ell^{0-1}(h_\mu, x + \delta, y) \geq \alpha$ if and only if at least an α proportion of inequalities in MaxFLS are satisfied. Thus, if a polynomial time algorithm was able to solve the problem of finding an optimal adversarial attack for a mixture, we could also use it to solve the MaxFLS problem in polynomial time, which is a contradiction, as MaxFLS is NP-hard. Thus, our problem is also NP-hard.

A.1.2 Proof of Lemma 3.2.1

Lemma 3.2.1.

Consider a set of m binary linear classifiers $\mathcal{H}$. Fix $\epsilon > 0$ the attack budget and a point (x, y) . If there exists a point $x' \in B_p(x, \epsilon)$ such that $\text{SRH}(\mathcal{H}, x', y) = 0$, then there exist parameters T and η such that minimizing $\text{SRH}(\mathcal{H}, x, y)$ w.r.t. x with PGD(T, η) will return x'' such that $\text{SRH}(\mathcal{H}, x'', y) = 0$.

Proof. (See [PS19, Appendix F] and [Bub+15])

Note that the hypothesis that there exists x' adversarial to all $h \in \mathcal{H}$ means that $\text{SRH}(\mathcal{H}, x', y) = 0$. This is a global minimum, as SRH is a non-negative function. Also note that SRH as a function of x can only take a finite number of values, the smallest positive one being $\frac{1}{m}$. By [PS19, Theorem 3], [Bub+15, Theorem 3.2] with $\delta < \frac{1}{m}$, running PGD for $T > \epsilon^2 \cdot m^2$ steps with step size $\eta = \frac{\epsilon}{\sqrt{T}}$ returns a solution x'' such that

$$\text{SRH}(\mathcal{H}, x'', y) - \text{SRH}(\mathcal{H}, x', y) = \text{SRH}(\mathcal{H}, x'', y) < \frac{1}{m}.$$

This implies that $\text{SRH}(\mathcal{H}, x'', y) = 0$.

Note that in the actual implementation of LCA (Algorithm 3), we will perform PGD multiple times with a subset of models that can be of size at most m . That means that we can take $T > \epsilon^2 \cdot m^2$ and $\eta < \frac{\epsilon}{\sqrt{T}}$, and these parameters would work for all the PGD runs during the whole execution of the algorithm.

A.1.3 Proof of Theorem 3.2.2

Theorem 3.2.2 (LCA is maximal in the binary linear setting).

Let $\mathcal{H}$ be a finite set of binary linear classifiers and h_μ a mixture over $\mathcal{H}$. Fix $\epsilon > 0$ the attack budget and a p -norm with $p > 1$. For any $(x, y) \in \mathbb{R}^d \times \{-1, 1\}$, there exist

parameters T and η for the inner PGD such that Algorithm 3 returns an adversarial example $x + \delta \in \text{CV}(\mathcal{H}')$, where $\mathcal{H}'$ is a maximal element of the adversarial lattice of h_μ at (x, y) .

Proof. Choose parameters T and η such that Lemma 3.2.1 holds.

Denote $\mathcal{H}_i$ the pool of fooled classifiers at each step i of the m steps in the outer loop of Algorithm 3, and x_i the adversarial attack at each step i . By construction, $\mathcal{H}_i \subseteq \mathcal{H}_{i+1}$, $\mathcal{H}_0 = \emptyset$ and $\mathcal{H}_m$ is the final pool of classifiers fooled by x_m , the output of Algorithm 3.

Suppose by contradiction that Algorithm 3 returns x_m that attacks a subset of classifiers $\mathcal{H}_m$ that is not maximal in the adversarial lattice of h_μ at (x, y) .

By definition of maximality, this means there exists some classifier $h' \in \mathcal{H} \setminus \mathcal{H}_m$ such that $\text{CV}(\mathcal{H}_m \cup \{h'\}) \neq \emptyset$.

Let j be the step at which h' was considered. At step $j - 1$ of the algorithm, the pool $\mathcal{H}_{j-1}$ consists of classifiers that are all vulnerable. Given that $\mathcal{H}_{j-1} \cup \{h'\} \subset \mathcal{H}_m \cup \{h'\}$, we have that $\text{CV}(\mathcal{H}_{j-1} \cup \{h'\}) \supset \text{CV}(\mathcal{H}_m \cup \{h'\}) \neq \emptyset$.

As $\text{CV}(\mathcal{H}_{j-1} \cup \{h'\}) \neq \emptyset$, at step j , Lemma 3.2.1 ensures that the PGD step for attacking the pool of classifiers $\mathcal{H}_{j-1} \cup \{h'\}$ returns x_j such that

$$\text{SRH}(\mathcal{H}_{j-1} \cup \{h'\}, x_j, y) = 0,$$

meaning that h' would be added to the pool, i.e $\mathcal{H}_j = \mathcal{H}_{j-1} \cup \{h'\}$. This implies that $h' \in \mathcal{H}_m$, which is a contradiction.

This contradiction arises from supposing that $\text{CV}(\mathcal{H}_m)$ was not maximal, which concludes the proof.

A.2 Proofs of Chapter 4

A.2.1 Proof of Lemma 4.1.2

Lemma 4.1.2.

Let $h : \mathcal{X} \rightarrow [0, 1]$ be any measurable function. Fix an arbitrary $x \in \mathcal{X}$ and $\alpha \in [0, 1]$. For any operator $>$ that can be either $>$ or $\geq$, $\epsilon > 0$ and p -norm with $p > 1$, the following inequality holds.

$$\mathbb{1} \left\{ \left(\sup_{x' \in B_p(x, \epsilon)} h(x') \right) > \alpha \right\} \geq \sup_{x' \in B_p(x, \epsilon)} \mathbb{1} \{ h(x') > \alpha \} \quad (4.3)$$

Additionally, the inequality in (4.3) becomes an equality if h takes finitely many values, or if h is continuous.

Proof. As both functions only take the values 0 and 1, it suffices to show that if the RHS is equal to 1, then so is the LHS.

Suppose $\sup_{x' \in B_p(x, \epsilon)} \mathbb{1} \{h(x') > \alpha\} = 1$. As the function $x \mapsto \mathbb{1} \{h(x) > \alpha\}$ takes a finite number of values, then the fact that the supremum over $B_p(x, \epsilon)$ equals 1 implies that

$$\exists x^* \in B_p(x, \epsilon) \text{ such that } \mathbb{1} \{h(x^*) > \alpha\} = 1,$$

which is equivalent to $h(x^*) > \alpha$. Taking supremum over $B_p(x, \epsilon)$, we get that $(\sup_{x' \in B_p(x, \epsilon)} h(x')) \geq h(x^*) > \alpha$. Thus, $\mathbb{1} \{(\sup_{x' \in B_p(x, \epsilon)} h(x')) > \alpha\} = 1$.

Equality. If h takes finitely many values, then clearly the supremum of h over $B_p(x, \epsilon)$ is attained. This is also the case if we add the assumption that h is continuous because $B_p(x, \epsilon)$ is a compact set.

In both cases, let $x^* \in B_p(x, \epsilon)$ be the point attaining the supremum of h over $B_p(x, \epsilon)$, i.e. $h(x^*) = \sup_{x' \in B_p(x, \epsilon)} h(x')$. If the LHS of (4.3) equals 1, then

$$\left(\sup_{x' \in B_p(x, \epsilon)} h(x') \right) > \alpha,$$

which implies that $h(x^*) > \alpha$. Thus, $\mathbb{1} \{h(x^*) > \alpha\} = 1$, and therefore

$$\sup_{x' \in B_p(x, \epsilon)} \mathbb{1} \{h(x') > \alpha\} \geq \mathbb{1} \{h(x^*) > \alpha\} = 1.$$

This completes the proof.

A.2.2 Proof of Theorem 4.1.3

Before proving Theorem 4.1.3, we need to establish a few definitions and lemmas:

Definition A.2.1 (Completion of a measure [Rud87, Theorem 1.36]).

Let ν be a measure over the set $\mathcal{Z}$ with σ -algebra $\mathcal{S}$. Define the set $\mathcal{L}(\mathcal{Z})_\nu$ as the collection of sets $E \subset \mathcal{Z}$ for which there exists $A, B \in \mathcal{S}$ with $A \subset E \subset B$ and $\nu(B \setminus A) = 0$, and define $\bar{\nu}(E) = \nu(A)$ in this situation. Then the set $\mathcal{L}(\mathcal{Z})_\nu$ is a σ -algebra and $\bar{\nu}$ is a measure on $\mathcal{L}(\mathcal{Z})_\nu$. The measure space $(\mathcal{Z}, \mathcal{L}(\mathcal{Z})_\nu, \bar{\nu})$ is called the completion of the measure space $(\mathcal{Z}, \mathcal{S}, \nu)$.

In particular, note that the following holds:

- For any $A \in \mathcal{S}$, $A \in \mathcal{L}(\mathcal{Z})_\nu$ and $\bar{\nu}(A) = \nu(A)$.
- Every $\mathcal{S}$ -measurable function f is also $\mathcal{L}(\mathcal{Z})_\nu$ -measurable and

$$\int_{\mathcal{Z}} f \, d\nu = \int_{\mathcal{Z}} f \, d\bar{\nu}. \quad (\text{A.1})$$

Regarding the completion of a measure ν , Equation (A.1) tells us that for integration purposes with $\mathcal{S}$ -measurable functions, we can work with the completion of the measure without loss of generality. We can ask what happens if we consider a $\mathcal{L}(\mathcal{Z})_\nu$ -measurable function to begin with. The following lemma provides an answer to this question:

Lemma A.2.2 (From [Rud87, Lemma 1, Theorem 8.12]).

Let ν be a positive measure over on a σ -algebra $\mathcal{S}$, $(\mathcal{L}(\mathcal{Z})_\nu, \bar{\nu})$ be the completion of $\mathcal{S}$ relative to ν and f be a $\mathcal{L}(\mathcal{Z})_\nu$ -measurable function. Then there exists a $\mathcal{S}$ -measurable function g such that $f = g$ $\bar{\nu}$ -almost everywhere.

Equation (A.1) and Lemma A.2.2 tell us that for integration purposes, we can work with the completion of a measurable space, because the integrals w.r.t. the base measure and its completion match. This will be useful in the proof of Theorem 4.1.3 when working with the adversarial risk.

We now define a particular σ -algebra that will be needed to ensure that there are no measurability issues when dealing with the zero-one loss under attack. We follow Awasthi et al. [AFM21] and Pydi and Jog [PJ23]:

Definition A.2.3 (Universal σ -algebra).

Let $\mathcal{B}(\mathcal{Z})$ denote the Borel σ -algebra over some set $\mathcal{Z}$. The universal σ -algebra is defined as the intersection of all completions of $(\mathcal{Z}, \mathcal{B}(\mathcal{Z}), \nu)$ for all finite measures ν over $\mathcal{B}(\mathcal{Z})$. Formally, let us denote $\mathcal{M}(\mathcal{Z})$ the set of all finite positive Borel measures over $\mathcal{B}(\mathcal{Z})$, and define the universal σ -algebra as

$$\mathcal{U}(\mathcal{Z}) = \bigcap_{\nu \in \mathcal{M}(\mathcal{Z})} \mathcal{L}(\mathcal{Z})_\nu.$$

The universal σ -algebra is larger than the Borel σ -algebra, and for any set $A \in \mathcal{U}(\mathcal{Z})$, there exists a Borel set $B \in \mathcal{B}(\mathcal{Z})$ such that $B \subseteq A$ and $\bar{\nu}(A) = \nu(B)$.

The universal σ -algebra is important in our case because of the following result:

Lemma A.2.4 (From [AFM21]).

Consider a Borel measurable classifier $h : \mathcal{X} \rightarrow [0, 1]$. Then, for any $\epsilon > 0$, $p > 1$ and $y \in \{0, 1\}$, the zero-one loss under attack $\mathcal{L}_{\epsilon, p}^{0-1}(h(\cdot), y) : \mathcal{X} \rightarrow [0, 1]$ is universally measurable.

As highlighted by [AFM21, Section 5], in order to properly define the adversarial risk, we need to start from the measure space $(\mathcal{X} \times \mathcal{Y}, \mathcal{B}(\mathcal{X} \times \mathcal{Y}), \rho)$, then consider

the completion $(\mathcal{X} \times \mathcal{Y}, \mathcal{L}(\mathcal{X} \times \mathcal{Y})_{\rho}, \bar{\rho})$, and finally consider the restriction of $\bar{\rho}$ to $\mathcal{U}(\mathcal{X} \times \mathcal{Y}) \subset \mathcal{L}(\mathcal{X} \times \mathcal{Y})_{\rho}$, that is, consider the measure space $(\mathcal{X} \times \mathcal{Y}, \mathcal{U}(\mathcal{X} \times \mathcal{Y}), \bar{\rho})$.

Now we focus on our case of interest: We assume that $\mathcal{X}$ is a compact subset of $\mathbb{R}^d$ and that $\mathcal{Y} = \{0, 1\}$, in which case the topology of $\mathcal{X} \times \mathcal{Y}$ is the product topology generated by the sets $A \times B$, where A is an open set of $\mathcal{X}$ with the usual topology and B is any subset of $\mathcal{Y}$. We denote $\mathcal{B}(\mathcal{X})$ the Borel σ -algebra over $\mathcal{X}$ (generated by open sets) and $\mathcal{B}(\mathcal{X} \times \mathcal{Y})$ the Borel σ -algebra over $\mathcal{X} \times \mathcal{Y}$.

Theorem 4.1.3.

Let $h : \mathcal{X} \rightarrow [0, 1]$ be any randomized binary classifier. Let h^α be the α -threshold classifier. Fix an arbitrary $\epsilon > 0$ and p -norm, $p > 1$. Then, for any data distribution ρ , the following equation holds:

$$\mathcal{R}_{\epsilon,p}(h, \rho) \geq \int_0^1 \mathcal{R}_{\epsilon,p}(h^\alpha, \rho) d\alpha \geq \inf_{\alpha \in [0,1]} \mathcal{R}_{\epsilon,p}(h^\alpha, \rho). \quad (4.4)$$

Furthermore, if h takes finitely many values, or h is continuous then the first inequality in (4.4) becomes an equality.

Proof. Fix an arbitrary Borel data distribution ρ over $\mathcal{X} \times \mathcal{Y}$ and any Borel measurable classifier $h : \mathcal{X} \rightarrow [0, 1]$.

Measurability considerations. We know by Lemma A.2.4 that in order to ensure the well-definedness of the adversarial risk using the zero-one loss, we need to work with the completion of the measure ρ and the universal σ -algebra $\mathcal{U}(\mathcal{X} \times \mathcal{Y})$. For simplicity during the following, we denote $f(x, y) = \mathcal{L}_{\epsilon,p}^{0-1}(h(x), y)$. We want to bound the adversarial risk which is equal to

$$\mathcal{R}_{\epsilon,p}(h, \rho) = \int_{\mathcal{X} \times \mathcal{Y}} f(x, y) d\bar{\rho}. \quad (A.2)$$

Disintegration of ρ .

We will use the disintegration theorem to decompose the measure ρ into a family of conditional measures that allow us to rewrite the adversarial risk of a classifier in terms of the per-class risk. Let $\text{proj}_2 : \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{Y}$ denote the projection into the second component, i.e. $\text{proj}_2(x, y) = y$. This function is continuous and Borel measurable over $\mathcal{X} \times \mathcal{Y}$, and defines the following push-forward measure over $\mathcal{Y}$:

$$\rho_{\mathcal{Y}}(\{y\}) = \rho(\text{proj}_2^{-1}(\{y\})) = \rho(\mathcal{X} \times \{y\}).$$

By the disintegration theorem, there exists a family of Borel measures $\{\nu_0, \nu_1\}$ over $\mathcal{X} \times \mathcal{Y}$ that satisfy the following properties:

- ν_y is supported on $\mathcal{X} \times \{y\}$ for $y \in \{0, 1\}$, i.e.

$$\forall y \in \{0, 1\}, \nu_y(\mathcal{X} \times \mathcal{Y} \setminus \text{proj}_2^{-1}(\{y\})) = 0,$$

- $\forall g: \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}_+$ Borel measurable,

$$\int_{\mathcal{X} \times \mathcal{Y}} g(x, y) d\rho = \int_{\mathcal{Y}} \int_{\mathcal{X} \times \{y\}} g(x, y) d\nu_y(x, y) d\rho_{\mathcal{Y}}(y). \quad (\text{A.3})$$

Given that $\mathcal{Y}$ is a finite space, we can further simplify (A.3) as follows:

$$\begin{aligned} \int_{\mathcal{X} \times \mathcal{Y}} g(x, y) d\rho &= \int_{\mathcal{Y}} \int_{\mathcal{X} \times \{y\}} g(x, y) d\nu_y(x, y) d\rho_{\mathcal{Y}}(y) \\ &= \sum_{y \in \{0, 1\}} \int_{\mathcal{X} \times \{y\}} g(x, y) d\nu_y(x, y) \cdot \rho_{\mathcal{Y}}(\{y\}) \\ &= \sum_{y \in \{0, 1\}} \int_{\mathcal{X} \times \{y\}} g(x, y) d\bar{\nu}_y(x, y) \cdot \rho_{\mathcal{Y}}(\{y\}). \end{aligned} \quad (\text{A.4})$$

The measures ν_0 and ν_1 can be seen as the conditional distributions of x given $y = 0$ and $y = 1$ respectively. We will denote $\bar{\rho}$, $\bar{\nu}_0$ and $\bar{\nu}_1$ their completions when restricted to $\mathcal{U}(\mathcal{X} \times \mathcal{Y})$, and note that the last equality in (A.4) holds because of (A.1).

Bounding the zero-one loss under attack. Coming back to (A.2), we can use (A.4) to obtain:

$$\begin{aligned} \mathcal{R}_{\epsilon, p}(h, \rho) &= \int_{\mathcal{X} \times \mathcal{Y}} f(x, y) d\bar{\rho} \\ &= \sum_{y \in \{0, 1\}} \int_{\mathcal{X} \times \{y\}} f(x, y) d\bar{\nu}_y(x, y) \cdot \rho_{\mathcal{Y}}(\{y\}) \end{aligned}$$

Note that for any fixed (x, y) and for any operator $>$ that can be either $>$ or $\geq$, we can rewrite the scalar $f(x, y) \in [0, 1]$ as $\int_0^1 \mathbb{1}\{f(x, y) > \alpha\} d\alpha$. This allows us to rewrite the previous equation as follows:

$$\begin{aligned} \mathcal{R}_{\epsilon, p}(h, \rho) &= \sum_{y \in \{0, 1\}} \int_{\mathcal{X} \times \{y\}} f(x, y) d\bar{\nu}_y(x, y) \cdot \rho_{\mathcal{Y}}(\{y\}) \\ &= \sum_{y \in \{0, 1\}} \int_{\mathcal{X} \times \{y\}} \int_0^1 \mathbb{1}\{f(x, y) > \alpha\} d\alpha d\bar{\nu}_y(x, y) \cdot \rho_{\mathcal{Y}}(\{y\}) \\ &= \sum_{y \in \{0, 1\}} \int_0^1 \int_{\mathcal{X} \times \{y\}} \mathbb{1}\{f(x, y) > \alpha\} d\bar{\nu}_y(x, y) d\alpha \cdot \rho_{\mathcal{Y}}(\{y\}). \end{aligned} \quad (\text{A.5})$$

In the last equation, we were able to interchange the two integrals using Tonelli's theorem. Indeed, the non-negative bounded function $(x, \alpha) \mapsto \mathbb{1}\{f(x, y) > \alpha\}$ inherits the measurability of $f(\cdot, y)$.

The next step is to use Lemma 4.1.2 to interchange the supremum operator and the indicator function. This will make the adversarial risk of h^α appear.

$$\begin{aligned}
\mathcal{R}_{\epsilon,p}(h, \rho) &= \sum_{y \in \{0,1\}} \int_0^1 \int_{\mathcal{X} \times \{y\}} \mathbb{1} \{f(x, y) > \alpha\} d\bar{\nu}_y(x, y) d\alpha \cdot \rho_Y(\{y\}) \\
&= \sum_{y \in \{0,1\}} \int_0^1 \int_{\mathcal{X} \times \{y\}} \mathbb{1} \left\{ \left(\sup_{x' \in B_p(x, \epsilon)} \mathcal{L}^{0-1}(h(x'), y) \right) > \alpha \right\} d\bar{\nu}_y(x, y) d\alpha \cdot \rho_Y(\{y\}) \\
&\geq \sum_{y \in \{0,1\}} \int_0^1 \int_{\mathcal{X} \times \{y\}} \sup_{x' \in B_p(x, \epsilon)} \mathbb{1} \{ \mathcal{L}^{0-1}(h(x'), y) > \alpha \} d\bar{\nu}_y(x, y) d\alpha \cdot \rho_Y(\{y\}). \quad (\text{A.6})
\end{aligned}$$

For the next step, we need to rewrite the zero-one loss of h and h^α as follows

$$\begin{aligned}
\mathcal{L}^{0-1}(h(x), 0) &= h(x) \\
\mathcal{L}^{0-1}(h(x), 1) &= 1 - h(x) \\
\mathcal{L}^{0-1}(h^\alpha(x), 0) &= h^\alpha(x) = \mathbb{1} \{h(x) > \alpha\}, \\
\mathcal{L}^{0-1}(h^\alpha(x), 1) &= 1 - h^\alpha(x) = 1 - \mathbb{1} \{h(x) > \alpha\}.
\end{aligned}$$

When $y = 0$ and setting $>$ to $>$, the double integral in (A.6) becomes

$$\begin{aligned}
&\int_0^1 \int_{\mathcal{X} \times \{0\}} \sup_{x' \in B_p(x, \epsilon)} \mathbb{1} \{ \mathcal{L}^{0-1}(h(x'), 0) > \alpha \} d\bar{\nu}_0(x, 0) d\alpha \\
&= \int_0^1 \int_{\mathcal{X} \times \{0\}} \sup_{x' \in B_p(x, \epsilon)} \mathbb{1} \{ h(x') > \alpha \} d\bar{\nu}_0(x, 0) d\alpha \\
&= \int_0^1 \int_{\mathcal{X} \times \{0\}} \sup_{x' \in B_p(x, \epsilon)} \mathcal{L}^{0-1}(h^\alpha(x'), 0) d\bar{\nu}_0(x, 0) d\alpha \quad (\text{A.7})
\end{aligned}$$

When $y = 1$ and setting $>$ to $\geq$, the double integral in (A.6) becomes

$$\begin{aligned}
&\int_0^1 \int_{\mathcal{X} \times \{1\}} \sup_{x' \in B_p(x, \epsilon)} \mathbb{1} \{ \mathcal{L}^{0-1}(h(x'), 1) \geq \alpha \} d\bar{\nu}_1(x, 1) d\alpha \\
&= \int_0^1 \int_{\mathcal{X} \times \{1\}} \sup_{x' \in B_p(x, \epsilon)} \mathbb{1} \{ 1 - h(x') \geq \alpha \} d\bar{\nu}_1(x, 1) d\alpha \\
&= \int_0^1 \int_{\mathcal{X} \times \{1\}} \sup_{x' \in B_p(x, \epsilon)} \mathbb{1} \{ 1 - \alpha \geq h(x') \} d\bar{\nu}_1(x, 1) d\alpha \\
&= \int_0^1 \int_{\mathcal{X} \times \{1\}} \sup_{x' \in B_p(x, \epsilon)} 1 - \mathbb{1} \{ h(x') > 1 - \alpha \} d\bar{\nu}_1(x, 1) d\alpha \\
&= \int_0^1 \int_{\mathcal{X} \times \{1\}} \sup_{x' \in B_p(x, \epsilon)} \mathcal{L}^{0-1}(h^{1-\alpha}(x'), 1) d\bar{\nu}_1(x, 1) d\alpha \\
&= \int_0^1 \int_{\mathcal{X} \times \{1\}} \sup_{x' \in B_p(x, \epsilon)} \mathcal{L}^{0-1}(h^u(x'), 1) d\bar{\nu}_1(x, 1) du \quad (\text{A.8})
\end{aligned}$$

In the last equation, the change of variable $u = 1 - \alpha$ was used. Putting (A.7) and (A.8) together into (A.6), we obtain the following result:

$$\begin{aligned}
\mathcal{R}_{\epsilon,p}(h, \rho) &\geq \sum_{y \in \{0,1\}} \int_0^1 \int_{\mathcal{X} \times \{y\}} \sup_{x' \in B_p(x, \epsilon)} \mathbb{1} \{ \mathcal{L}^{0-1}(h(x'), y) > \alpha \} d\bar{\nu}_y(x, y) d\alpha \cdot \rho_Y(\{y\}) \quad (\text{A.9}) \\
&= \sum_{y \in \{0,1\}} \int_0^1 \int_{\mathcal{X} \times \{y\}} \sup_{x' \in B_p(x, \epsilon)} \mathcal{L}^{0-1}(h^\alpha(x'), y) d\bar{\nu}_y(x, y) d\alpha \cdot \rho_Y(\{y\}) \\
&= \int_0^1 \sum_{y \in \{0,1\}} \int_{\mathcal{X} \times \{y\}} \sup_{x' \in B_p(x, \epsilon)} \mathcal{L}^{0-1}(h^\alpha(x'), y) d\bar{\nu}_y(x, y) \cdot \rho_Y(\{y\}) d\alpha \\
&= \int_0^1 \int_{\mathcal{X} \times \mathcal{Y}} \sup_{x' \in B_p(x, \epsilon)} \mathcal{L}^{0-1}(h^\alpha(x'), y) d\bar{\rho}(x, y) d\alpha \\
&= \int_0^1 \mathcal{R}_{\epsilon,p}(h^\alpha, \rho) d\alpha. \quad (\text{A.10})
\end{aligned}$$

This completes the first part of the proof. If h takes finitely many values or is continuous, then by Lemma 4.1.2 the inequality (A.6) becomes an equality, and thus (A.9) becomes an equality as well. Thus, we obtain that $\mathcal{R}_{\epsilon,p}(h, \rho) = \int_0^1 \mathcal{R}_{\epsilon,p}(h^\alpha, \rho) d\alpha$.

Finally, to complete the proof, note that $\eta := \inf_{\alpha \in [0,1]} \mathcal{R}_{\epsilon,p}(h^\alpha, \rho)$ trivially satisfies $\eta \leq \mathcal{R}_{\epsilon,p}(h^\alpha, \rho)$ for all $\alpha \in [0,1]$. Taking integral over α on both sides, we obtain $\eta \leq \int_0^1 \mathcal{R}_{\epsilon,p}(h^\alpha, \rho) d\alpha$. We thus obtain that

$$\inf_{\alpha \in [0,1]} \mathcal{R}_{\epsilon,p}(h^\alpha, \rho) \leq \mathcal{R}_{\epsilon,p}(h, \rho). \quad (\text{A.11})$$

Clearly, if the infimum is attained at some α^* , then we have that $\mathcal{R}_{\epsilon,p}(h^{\alpha^*}, \rho) \leq \mathcal{R}_{\epsilon,p}(h, \rho)$.

A.3 Example of mixtures with strictly better adversarial risk

In this section, we are going to present a motivating example that shows how even finite mixtures of two classifiers can achieve a strictly better adversarial risk than their deterministic components, effectively demonstrating that this kind of randomization can produce a *real robustness gain*. We will then apply Theorem 4.1.3 to find the corresponding deterministic classifiers that will outperform the best mixture.

In this example, we assume the real data distribution is supported on four points $x_1, \dots, x_4 \in \mathbb{R}^2$. We will use colors to simplify the distinction between points, which have the following coordinates: $x_1 = (0, 1)$, $x_2 = (-2.7, 1.1)$, $x_3 = (2.7, 1.1)$ and $x_4 = (0, -2)$. The mass of x_1 is equal to $\frac{1}{2}$, while the mass of all the other points equals $\frac{1}{6}$. For this extended analysis, it will be easier to work with classes in $\{-1, 1\}$, so we have labels $y_1 = 1$ and $y_2 = y_3 = y_4 = -1$. We set $\epsilon = 1$ and consider an ℓ_2 adversary.

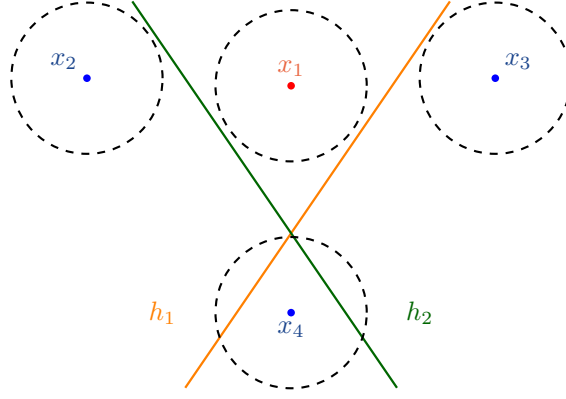


Fig. A.3.1.: Example with a data distribution over 4 points and 2 linear classifiers with optimal adversarial risk over the family of linear classifiers.

We will consider $\mathcal{H}$ to be the family of all linear classifiers in $\mathbb{R}^2$. Let $f = (w, b)$ be a generic linear function in $\mathbb{R}^2$, where $w = (w_1, w_2)$ is the normal vector of the hyperplane and b is the bias term. The classification is done using the rule $h(x) = \text{sign}(f(x)) = \text{sign}(w^T x + b)$, as in standard binary classification with labels $\{-1, 1\}$.

In this setting, a point $(x, y) \in \mathbb{R}^2 \times \{-1, 1\}$ is correctly classified if $yf(x) > 0$. Moreover, for linear classifiers, the optimal adversarial perturbation is known and robustness at a point can be easily studied. Note that given a perturbation δ , we can check if it induces a misclassification by simply computing $yf(x + \delta)$. We can say h is robust at (x, y) if, for every perturbation δ of norm at most ϵ , we have that $yf(x + \delta) > 0$. This can be developed to get

$$\begin{aligned}
 yf(x + \delta) > 0 &\iff y(w^T(x + \delta) + b) > 0 \\
 &\iff y(w^T x + b) + y(w^T \delta) > 0 \\
 &\iff y(w^T x + b) > -y(w^T \delta)
 \end{aligned} \tag{A.12}$$

For h to be robust at (x, y) , (A.12) must hold for all δ , in particular for the quantity maximizing the RHS. This happens for $\delta^* = -y\epsilon \frac{w}{\|w\|}$. Inserting this worst case perturbation, we obtain a simple condition for robustness in the linear case:

$$y(w^T x + b) > \epsilon \|w\| \tag{A.13}$$

Without loss of generality, we can assume w has norm 1 so the last equation further simplifies. We can use these inequalities to show that no linear classifier can be robust at x_1 , x_2 and x_4 at the same time. Indeed, being robust at each point gives us the following inequalities:

$$\begin{aligned}
(x_1) : w_2 + b &> 1 & \implies b > 1 - w_2 \\
(x_2) : 2.7w_1 - 1.1w_2 - b &> 1 & \implies b < 2.7w_1 - 1.1w_2 - 1 \\
(x_4) : 2w_2 - b &> 1 & \implies b < 2w_2 - 1
\end{aligned}$$

Using the inequalities for x_1 and x_4 , we get the condition $w_2 > \frac{2}{3}$. Using the inequalities for x_1 and x_2 gives us that $2.7w_1 - 0.1w_2 > 2$. Together with the bound on w_2 , we get the bound $w_1 > \frac{62}{81}$. These two bounds make it impossible for w to have norm 1, contradicting our hypothesis. An analogous reasoning shows that no linear classifier can be robust at x_1 , x_3 and x_4 at the same time.

Knowing that a linear classifier can't be robust at x_1 , x_2 and x_4 simultaneously, we can further ask what is the best adversarial risk a linear classifier can have. One can easily check that it is possible to be robust on any of the pairs (x_1, x_2) , (x_1, x_3) , or (x_2, x_4) . At each time, robustness on the two selected points comes at the expense of non-robustness on at least a third one. Summarizing, there are three families of classifiers of interest, each with adversarial risk of $\frac{1}{3}$: robust at (x_1, x_2) , robust at (x_1, x_3) or robust at (x_1, x_4) . Any other type of classifier will have at least this adversarial risk.

Now let us consider the family of finite mixtures over $\mathcal{H}$ with two classifiers. We will show that we can get a strictly better adversarial risk by using such classifiers. To do so, we propose two linear classifiers h_1 and h_2 (See Figure A.3.1), with weights $\mu(\{h_1\}) = \mu(\{h_2\}) = \frac{1}{2}$:

- $h_1 = \text{sign}(w_1^T x + b_1)$ with $w_1 = (0.825, 0.565132728)$ and $b_1 = 0.536876091$.
- $h_2 = \text{sign}(w_2^T x + b_2)$ with $w_2 = (-0.825, 0.565132728)$ and $b_2 = 0.536876091$.

Using (A.13) one can check that these classifiers match the situation that is illustrated in Figure A.3.1, i.e. they are both robust at x_1 , they are robust at x_3 and x_2 respectively, they are non-robust at x_4 but they can't be attacked on the same region, as their intersections with the ϵ -ball around x_4 are disjoint.

Adversarial risk calculation for the mixture of h_1 and h_2 . Let us start by getting an explicit description of h_μ over all our inputs. A vector $(\alpha, 1 - \alpha)$ means that class -1 is predicted with probability α and class 1 with probability $1 - \alpha$. One can verify that h_μ makes the following predictions:

$$\begin{aligned}
h_\mu(x_1) &= (0, 1) & h_\mu(x_2) &= (\tfrac{1}{2}, \tfrac{1}{2}) \\
h_\mu(x_3) &= (\tfrac{1}{2}, \tfrac{1}{2}) & h_\mu(x_4) &= (1, 0)
\end{aligned}$$

Note that for the points x_1 , x_2 and x_3 , these predictions remain constant inside the whole ϵ -balls around them. This is not the case for x_4 , where the prediction becomes $h_\mu(\cdot) = (\frac{1}{2}, \frac{1}{2})$ in the two circular sectors created by the two linear classifiers.

Now, let us compute the adversarial 0-1 loss over each point. We have the following:

$$\begin{aligned}\mathcal{L}_{\epsilon,p}^{0-1}(h_\mu, x_1, 1) &= 0 & \mathcal{L}_{\epsilon,p}^{0-1}(h_\mu, x_2, -1) &= 0.5 \\ \mathcal{L}_{\epsilon,p}^{0-1}(h_\mu, x_3, -1) &= 0.5 & \mathcal{L}_{\epsilon,p}^{0-1}(h_\mu, x_4, -1) &= 0.5,\end{aligned}$$

where the loss over x_4 is $\frac{1}{2}$ because h_1 nor h_2 are in a matching pennies configuration.

Therefore, the adversarial risk of h_μ is exactly

$$\mathcal{R}_{\epsilon,p}(h_\mu, \rho) = \frac{1}{2} \cdot 0 + \frac{1}{6} \cdot 0.5 + \frac{1}{6} \cdot 0.5 + \frac{1}{6} \cdot 0.5 = \frac{1}{4}, \quad (\text{A.14})$$

which is strictly better than, $\frac{1}{3}$, the best adversarial risk that can be achieved by any linear classifier. One can also prove that this is the best risk possible with this kind of randomized classifiers.

Determinizing the optimal mixture. The example presented in Figure A.3.1 showed a situation in which a finite mixture h_μ of two linear classifiers h_1 and h_2 could outperform any linear classifier, including h_1 and h_2 themselves. Theorem 4.1.3 applied to this case guarantees that there exists a *weighted ensemble* h^α that is at least as robust as h_μ . We are going to explicitly compute the threshold hypothesis set $\mathcal{H}_T(h_\mu)$ for this example and find the deterministic classifier guaranteed by Theorem 4.1.3.

Note that $h_\mu(x) = \frac{1}{2}h_1(x) + \frac{1}{2}h_2(x)$ can take only three possible values: 0, $1/2$ or 1 depending on the values of $h_1(x)$ and $h_2(x)$. This fact can be summarized in Table A.3.1.

$h_1(x)$	$h_2(x)$	$h_\mu(x)$
0	0	0
0	1	0.5
1	0	0.5
1	1	1

Tab. A.3.1.: Definition of the finite mixture h_μ in terms of the outputs of the base classifiers h_1 and h_2 .

From the definition of h_μ , we can define the α -threshold classifier h_μ^α in a similar manner depending on the value of the threshold α . This is done in Tables A.3.2 and A.3.3, which resemble the truth tables of the or and and logical operators respectively. Additionally, if $\alpha = 1$, then h_μ^α is constant and predicts only the class 0. Let us denote h_μ^{or} the classifier defined in Table A.3.2, h_μ^{and} the classifier defined in Table A.3.3, and h_μ^0 the constant classifier that predicts class 0 for all input x . Then, the threshold hypothesis set $\mathcal{H}_T(h_\mu)$ is given by $\mathcal{H}_T(h_\mu) = \{h_\mu^0, h_\mu^{\text{and}}, h_\mu^{\text{or}}\}$. The classifiers h_μ^{and} and h_μ^{or} are represented in Figure A.3.4.

$h_1(x)$	$h_2(x)$	$h_\mu^\alpha(x)$
0	0	0
0	1	1
1	0	1
1	1	1

Tab. A.3.2.: h_μ^α in the case $\alpha \in [0, 1/2)$.

$h_1(x)$	$h_2(x)$	$h_\mu^\alpha(x)$
0	0	0
0	1	0
1	0	0
1	1	1

Tab. A.3.3.: h_μ^α in the case $\alpha \in [1/2, 1)$.

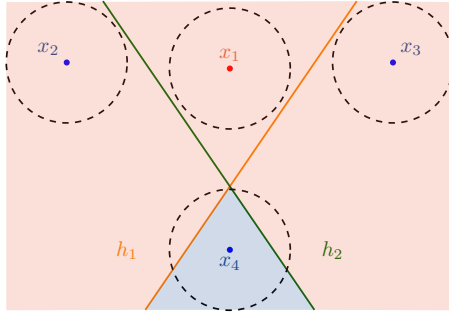


Fig. A.3.2.: Decision regions of h_μ^{or} .

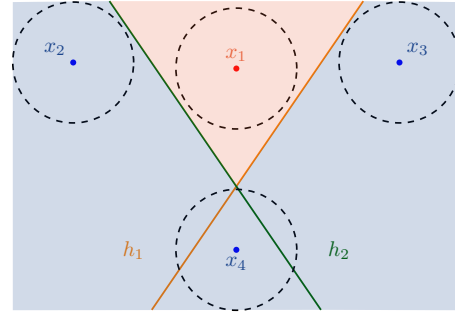


Fig. A.3.3.: Decision regions of h_μ^{and} .

Fig. A.3.4: Visual representation of the threshold classifiers h_μ^α for the two regimes of α . On the LHS is the and classifier, and on the RHS the or classifier. The orange and blue regions represent the regions of the space on which the classifiers predict 1 and 0, respectively.

Table A.3.4 details the adversarial risk of each of the classifiers in $\mathcal{H}_T(h_\mu)$, as well as the adversarial risk of the mixture h_μ and the individual classifiers h_1 and h_2 . Recall that the probabilities associated to the points are $\rho(\{(x_1, 1)\}) = 1/2$, $\rho(\{(x_2, 0)\}) = \rho(\{(x_3, 0)\}) = \rho(\{(x_4, 0)\}) = 1/6$ and 0 otherwise. From Table A.3.4 we conclude that the classifier h_μ^{and} is the one with the lowest adversarial risk, and that it is at least as robust as the mixture h_μ . In this particular case, it is strictly better and it is even a perfect classifier under attack. We can also check that (4.9) holds, that is, that the adversarial risk of the mixture h_μ can be written in terms of the risks of the classifiers in $\mathcal{H}_T(h_\mu)$.

	Error at each point				
Classifier	$(x_1, 1)$	$(x_2, 0)$	$(x_3, 0)$	$(x_4, 0)$	$\mathcal{R}_{\epsilon, 2}(\cdot, \rho)$
h_1	0	1	0	1	$1/3$
h_2	0	0	1	1	$1/3$
h_μ	0	0.5	0.5	0.5	$1/4$
h_μ^0	1	0	0	0	$1/2$
h_μ^{and}	0	0	0	0	0
h_μ^{or}	0	1	1	1	$1/2$

Tab. A.3.4.: Adversarial risk of each of the classifiers in the threshold hypothesis set $\mathcal{H}_T(h_\mu)$, the mixture h_μ and the optimal linear classifiers h_1, h_2 for the example presented in Figure A.3.1. We also represent in the intermediate columns the error under attack at each point (x_i, y_i) .

Supplementary material for Chapter 5

Contents

B.1	Reformulation of the adversarial risk minimization problem for discrete distributions with finite support as a set packing problem	109
B.1.1	Proof of Lemma 5.1.3	109
B.1.2	Proof of Corollary 5.1.4	112
B.1.3	Proof of Corollary 5.1.5	112
B.2	Necessary structures for a positive randomization gap	113
B.2.1	Proof of Lemma 5.2.2	113
B.2.2	Proof of Lemma 5.2.3	115
B.2.3	Proof of Lemma 5.2.4	117
B.2.4	Proof of Theorem 5.2.5	117
B.2.5	Proof of Corollary 5.2.7	118
B.2.6	Constructions of specific structures for different ℓ_p norms	119
B.3	Constructing discrete distributions with large randomization gap	121
B.3.1	Every graph is the conflict graph of a distribution	122
B.3.2	Constructing non-perfect graphs with large randomization gap using the fibration construction	126
B.3.3	Proof of Theorem 5.3.1	129
B.4	On the hardness of computing the randomization gap	130
B.4.1	Reminder on NP-hardness and co-NP-hardness	131
B.4.2	Hardness of the RG decision problem	132

B.1 Reformulation of the adversarial risk minimization problem for discrete distributions with finite support as a set packing problem

B.1.1 Proof of Lemma 5.1.3

Lemma 5.1.3 (Extended from [Dai+24]).

Let us consider a p -norm with $p > 1$, and $\epsilon > 0$. Let ρ be an arbitrary discrete

distribution over $\mathcal{X} \times \mathcal{Y}$ with support $s_\rho = \{(x_i, y_i)\}_{i \in [n]}$ and probability vector $\omega_\rho \in \Delta^n$. The following statements hold true:

- a) For any fractional packing $q \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)$, there exists a classifier $h_q \in \mathcal{H}_{\text{rand}}$ such that $\mathcal{A}_{\epsilon, p}(h_q, \rho) \geq \omega_\rho^T q$.
- b) For any classifier $h \in \mathcal{H}_{\text{rand}}$, there exists a fractional packing $q_h \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)$ such that $\mathcal{A}_{\epsilon, p}(h, \rho) = \omega_\rho^T q_h$.

Proof. Throughout the proof, we denote $\mathcal{E}$ the set of hyperedges of the conflict hypergraph $H_{s_\rho}^\epsilon$.

Proof of a). Let $q \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)$ be a fractional packing for (5.4) over $(H_{s_\rho}^\epsilon, \omega_\rho)$. The first part of the proof will be to define an auxiliary function g_q that will allow us, on a second stage, to define a classifier h_q with the properties that we seek.

Definition of the auxiliary function. Given $(x, y) \in \mathcal{X} \times \mathcal{Y}$, define

$$\mathcal{I}(x, y) = \{i \in [n] \mid x \in B_p(x_i, \epsilon) \text{ and } y_i = y\},$$

the set of indices of neighbors of x on the set s_μ that are of class y . We denote $\mathcal{Y}_x = \{y \in \mathcal{Y} \mid \mathcal{I}(x, y) \neq \emptyset\}$ the set of classes for which x has a neighbor in s_μ that belongs to that class. Denote $g_q : \mathcal{X} \rightarrow \mathbb{R}_+^K$ the function defined as follows:

$$\forall x \in \mathcal{X}, \quad g_q(x)^{(y)} = \begin{cases} \max_{i \in \mathcal{I}(x, y)} q^{(i)}, & \text{if } \mathcal{I}(x, y) \neq \emptyset. \\ 0, & \text{otherwise.} \end{cases}$$

Given that q is a fractional packing of $H_{s_\rho}^\epsilon$, we have that for any hyperedge $e \in \mathcal{E}$,

$$\sum_{i \in e} q^{(i)} \leq 1. \tag{B.1}$$

Recall that for every $i \in [n]$, $\{i\} \in \mathcal{E}$, which by (B.1) implies that

$$\forall i \in [n], \quad q^{(i)} \leq 1.$$

Thus, $0 \leq g_q(x)^{(y)} \leq 1$ for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$. Let us now consider a given $x \in \mathcal{X}$. If x is such that $\mathcal{Y}_x \neq \emptyset$, we can define for every $y \in \mathcal{Y}_x$ the index i_y as

$$i_y = \operatorname{argmax}_{j \in \mathcal{I}(x, y)} q^{(j)}.$$

Then, for any x such that $\mathcal{Y}_x \neq \emptyset$, we can write $g_q(x)^{(y)}$ as

$$g_q(x)^{(y)} = \begin{cases} q^{(i_y)}, & \text{if } \mathcal{I}(x, y) \neq \emptyset. \\ 0, & \text{otherwise.} \end{cases}$$

Furthermore, let $e_x = \{i_y \in [n] \mid y \in \mathcal{Y}_x\}$. Note that $e_x \in \mathcal{E}$, because $x \in \bigcap_{j \in e} B_p(x_j, \epsilon)$. Given that $q \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)$, the following inequality holds

$$\sum_{y \in \mathcal{Y}} g_q(x)^{(y)} = \sum_{y \in \mathcal{Y}_x} g_q(x)^{(y)} = \sum_{i \in e_x} q^{(i)} \leq 1.$$

Definition of the classifier. We now define a classifier h_q as follows:

$$\forall x \in \mathcal{X}, \quad h_q(x)^{(y)} = \begin{cases} 1 - \sum_{k > 1} g_q(x)^{(k)} & \text{if } y = 1, \\ g_q(x)^{(y)}, & \text{if } y > 1. \end{cases}$$

By construction, $\sum_{y \in \mathcal{Y}} h_q(x)^{(y)} = 1$, and also $h_q(x)^{(y)} \geq g_q(x)^{(y)}$ for any $(x, y) \in \mathcal{X} \times \mathcal{Y}$ with $\mathcal{Y}_x \neq \emptyset$. The adversarial accuracy of h_q satisfies the following inequality:

$$\begin{aligned} \mathcal{A}_{\epsilon, p}(h_q, \rho) &= \sum_{i \in [n]} \omega_\rho^{(i)} \cdot \inf_{x' \in B_p(x_i, \epsilon)} h_q(x')^{(y_i)} \\ &\geq \sum_{i \in [n]} \omega_\rho^{(i)} \cdot \inf_{x' \in B_p(x_i, \epsilon)} \max_{j \in \mathcal{I}(x', y_i)} q^{(j)} \\ &\geq \sum_{i \in [n]} \omega_\rho^{(i)} \cdot \inf_{x' \in B_p(x_i, \epsilon)} q^{(i)} \quad (\text{as } i \in \mathcal{I}(x', y_i)) \\ &\geq \sum_{i \in [n]} \omega_\rho^{(i)} \cdot q^{(i)} \end{aligned}$$

This completes the first part of the proof.

Proof of b). Let us consider an arbitrary classifier $h \in \mathcal{H}_{\text{rand}}$, and denote q_h the vector defined as

$$\forall i \in [n], \quad q_h^{(i)} = \inf_{x' \in B_p(x_i, \epsilon)} h(x')^{(y_i)}.$$

Clearly, $q_h^{(i)} \geq 0$ for all $i \in [n]$. To see that q_h is feasible for (5.4) over $(H_{s_\rho}^\epsilon, \omega_\rho)$, consider any hyperedge $e \in \mathcal{E}$ and take an arbitrary $x_e \in \bigcap_{i \in e} B_p(x_i, \epsilon)$. We have that

$$\begin{aligned} \sum_{i \in e} q_h^{(i)} &= \sum_{i \in e} \inf_{x' \in B_p(x_i, \epsilon)} h(x')^{(y_i)} \\ &\leq \sum_{i \in e} h(x_e)^{(y_i)} \quad (x_e \in B_p(x_i, \epsilon) \text{ for all } i \in e) \\ &\leq \sum_{y \in \mathcal{Y}} h(x_e)^{(y)} \quad (y_i \neq y_j \text{ for any } i, j \in e) \\ &= 1 \quad (h \text{ is a valid classifier}) \end{aligned}$$

This holds for any hyperedge in $\mathcal{E}$, hence $q_h \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)$. Finally, we can compute the accuracy of h :

$$\mathcal{A}_{\epsilon, p}(h, \rho) = \sum_{i \in [n]} \omega_\rho^{(i)} \cdot \inf_{x' \in B_p(x_i, \epsilon)} h(x')^{(y_i)} = \sum_{i \in [n]} \omega_\rho^{(i)} \cdot q_h^{(i)} = \omega_\rho^T q_h,$$

which concludes the proof.

^aNote that the isolated points, i.e. $\mathcal{Y}_x = \emptyset$, are not involved in the computation of the accuracy. Therefore, the classifier h_q can be defined in any way on those points. For this construction in particular, h_q will always predict class 1 on points x such that $\mathcal{Y}_x = \emptyset$.

B.1.2 Proof of Corollary 5.1.4

Corollary 5.1.4.

Let us consider a p -norm with $p > 1$, and $\epsilon > 0$. For any discrete distribution ρ over $\mathcal{X} \times \mathcal{Y}$ we have:

$$1 - \mathcal{R}_{\epsilon,p}(\mathcal{H}_{\text{rand}}, \rho) = \text{FP}(H_{s_\rho}^\epsilon, \omega_\rho).$$

Proof. By the first item of Lemma 5.1.3, we have that for any vector $q \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)$, there is a classifier $h_q \in \mathcal{H}_{\text{rand}}$ for which

$$\mathcal{A}_{\epsilon,p}(\mathcal{H}_{\text{rand}}, \rho) \geq \mathcal{A}_{\epsilon,p}(h_q, \rho) \geq \omega_\rho^T q.$$

As this is true for any $q \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)$, we have that

$$\mathcal{A}_{\epsilon,p}(\mathcal{H}_{\text{rand}}, \rho) \geq \max_{q \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)} \omega_\rho^T q = \text{FP}(H_{s_\rho}^\epsilon, \omega_\rho). \quad (\text{B.2})$$

By the second item of Lemma 5.1.3, we have that for any classifier $h \in \mathcal{H}_{\text{rand}}$, there is a vector $q_h \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)$ for which

$$\mathcal{A}_{\epsilon,p}(h, \rho) = \omega_\rho^T q_h \leq \max_{q \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)} \omega_\rho^T q = \text{FP}(H_{s_\rho}^\epsilon, \omega_\rho).$$

As this is true for any classifier $h \in \mathcal{H}_{\text{rand}}$, then by taking max over all classifiers we have that

$$\mathcal{A}_{\epsilon,p}(\mathcal{H}_{\text{rand}}, \rho) \leq \text{FP}(H_{s_\rho}^\epsilon, \omega_\rho). \quad (\text{B.3})$$

Combining (B.2) and (B.3) concludes the proof.

B.1.3 Proof of Corollary 5.1.5

Corollary 5.1.5.

Let us consider a p -norm with $p > 1$, and $\epsilon > 0$. For any discrete distribution ρ over $\mathcal{X} \times \mathcal{Y}$ we have:

$$1 - \mathcal{R}_{\epsilon,p}(\mathcal{H}_{\text{det}}, \rho) = \text{IP}(H_{s_\rho}^\epsilon, \omega_\rho).$$

Proof. The reasoning is analogous to the proof of Corollary 5.1.4 but restricting to deterministic classifiers $\mathcal{H}_{\text{det}}$. Let $q \in \mathcal{Q}(H_{s_\rho}^\epsilon)$ be an arbitrary packing of $H_{s_\rho}^\epsilon$. Note

that $q \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)$, so by Lemma 5.1.3, there exists a classifier $h_q \in \mathcal{H}_{\text{rand}}$ such that $\mathcal{A}_{\epsilon,p}(h_q, \rho) \geq \omega_\rho^T q$. By the construction of h_q in the proof of Lemma 5.1.3, we know that h_q is deterministic, i.e. $h_q \in \mathcal{H}_{\text{det}}$ and thus $\mathcal{A}_{\epsilon,p}(\mathcal{H}_{\text{det}}, \rho) \geq \mathcal{A}_{\epsilon,p}(h_q, \rho)$. As this is true for any $q \in \mathcal{Q}(H_{s_\rho}^\epsilon)$, we have that

$$\mathcal{A}_{\epsilon,p}(\mathcal{H}_{\text{det}}, \rho) \geq \max_{q \in \mathcal{Q}(H_{s_\rho}^\epsilon)} \omega_\rho^T q = \text{IP}(H_{s_\rho}^\epsilon, \omega_\rho). \quad (\text{B.4})$$

On the other hand, any deterministic classifier $h \in \mathcal{H}_{\text{det}}$ is also a randomized classifier, thus by Lemma 5.1.3, there exists a vector $q_h \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)$ for which $\mathcal{A}_{\epsilon,p}(h, \rho) = \omega_\rho^T q_h$. Moreover, given the construction of q_h in the proof of Lemma 5.1.3, we know that $q_h \in \{0, 1\}^n$, which means that q_h is a packing, i.e., $q_h \in \mathcal{Q}(H_{s_\rho}^\epsilon)$, and therefore $\mathcal{A}_{\epsilon,p}(h, \rho) \leq \text{IP}(H_{s_\rho}^\epsilon, \omega_\rho)$. As this is true for any classifier $h \in \mathcal{H}_{\text{det}}$, we have that

$$\mathcal{A}_{\epsilon,p}(\mathcal{H}_{\text{det}}, \rho) \leq \text{IP}(H_{s_\rho}^\epsilon, \omega_\rho). \quad (\text{B.5})$$

Combining (B.4) and (B.5) concludes the proof.

B.2 Necessary structures for a positive randomization gap

B.2.1 Proof of Lemma 5.2.2

Before proving Lemma 5.2.2, we need the following technical lemma:

Lemma B.2.1.

Let us consider an ℓ_p norm with $p \in (1, \infty]$, and $\epsilon > 0$. Let also $S = \{(x_i, y_i)\}_{i \in [n]}$ be an arbitrary set of points from $\mathcal{X} \times \mathcal{Y}$. The following assertion hold true

- a) For any edge $e \in E(G_S^\epsilon)$, there exists some hyperedge $e' \in \mathcal{E}(H_S^\epsilon)$ such that $e \subseteq e'$.
- b) For any hyperedge $e \in \mathcal{E}(H_S^\epsilon)$, there exists some hyperedge $e' \in \mathcal{E}(C_S^\epsilon)$ such that $e \subseteq e'$.

Proof. Proof of a). This is true by definition of conflict hypergraph and conflict graph. In particular, by the fact that G_S^ϵ is the 2-section of H_S^ϵ .

Proof of b). Now consider an arbitrary hyperedge $e \in \mathcal{E}(H_S^\epsilon)$. By definition of conflict graph, this implies that

$$\forall i, j \in e \text{ with } i \neq j, \quad \{i, j\} \in E(G_S^\epsilon).$$

The fact that all the pairs $i, j \in e$ are edges of G_S^ϵ means that e constitutes a clique of G_S^ϵ . Then, by definition of the clique hypergraph as the one whose hyperedges are the maximal cliques of G_S^ϵ , we have that

$$\exists e' \in \mathcal{E}(C_S^\epsilon) \text{ such that } e \subseteq e'.$$

The hierarchy over the three hypergraphs, as established in Lemma B.2.1, corresponds to an ordering of the values in their respective fractional set packing problems. We now prove Lemma 5.2.2, which states that the three hypergraphs $G_{s_\rho}^\epsilon$, $H_{s_\rho}^\epsilon$ and $C_{s_\rho}^\epsilon$ represent different formulations of the same set packing problem, with their fractional versions providing upper bounds of varying tightness.

Lemma 5.2.2.

Let us consider an ℓ_p norm with $p \in (1, \infty]$, $\epsilon > 0$, and a discrete distribution ρ over $\mathcal{X} \times \mathcal{Y}$. The following holds:

$$\text{IP}(C_{s_\rho}^\epsilon, \omega_\rho) = \text{IP}(H_{s_\rho}^\epsilon, \omega_\rho) = \text{IP}(G_{s_\rho}^\epsilon, \omega_\rho). \quad (5.12)$$

$$\text{FP}(C_{s_\rho}^\epsilon, \omega_\rho) \leq \text{FP}(H_{s_\rho}^\epsilon, \omega_\rho) \leq \text{FP}(G_{s_\rho}^\epsilon, \omega_\rho). \quad (5.13)$$

Proof. Proof of (5.13). We will begin by using Lemma B.2.1 to show that the set of fractional packings of the hypergraphs $C_{s_\rho}^\epsilon$, $H_{s_\rho}^\epsilon$ and $G_{s_\rho}^\epsilon$ satisfy the following relation:

$$\mathcal{Q}^{\text{frac}}(C_{s_\rho}^\epsilon) \subseteq \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon) \subseteq \mathcal{Q}^{\text{frac}}(G_{s_\rho}^\epsilon).$$

This will prove that (5.13) holds, as the fractional packing problem is a maximization problem.

($\mathcal{Q}^{\text{frac}}(C_{s_\rho}^\epsilon) \subseteq \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)$). Take an arbitrary $q \in \mathcal{Q}^{\text{frac}}(C_{s_\rho}^\epsilon)$ and an arbitrary hyperedge $e \in \mathcal{E}(H_{s_\rho}^\epsilon)$. By Lemma B.2.1, there exists a hyperedge $e' \in \mathcal{E}(C_{s_\rho}^\epsilon)$ such that $e \subseteq e'$. We then have that

$$\sum_{i \in e} q^{(i)} \leq \sum_{i \in e'} q^{(i)} \leq 1,$$

where the last inequality holds because $q \in \mathcal{Q}^{\text{frac}}(C_{s_\rho}^\epsilon)$ and $e' \in \mathcal{E}(C_{s_\rho}^\epsilon)$. As this holds for every hyperedge $e \in \mathcal{E}(H_{s_\rho}^\epsilon)$, we conclude that q is a fractional packing of $H_{s_\rho}^\epsilon$, i.e. $q \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)$.

($\mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon) \subseteq \mathcal{Q}^{\text{frac}}(G_{s_\rho}^\epsilon)$). Similarly, take an arbitrary $q \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)$ and an arbitrary edge $e \in \mathcal{E}(G_{s_\rho}^\epsilon)$. By Lemma B.2.1, there exists a hyperedge $e' \in \mathcal{E}(H_{s_\rho}^\epsilon)$ such that $e \subseteq e'$. We then have that

$$\sum_{i \in e} q^{(i)} \leq \sum_{i \in e'} q^{(i)} \leq 1,$$

where the last inequality holds because $q \in \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon)$ and $e' \in \mathcal{E}(H_{s_\rho}^\epsilon)$. As this holds for every hyperedge $e \in \mathcal{E}(G_{s_\rho}^\epsilon)$, we conclude that q is a fractional packing of $G_{s_\rho}^\epsilon$, i.e. $q \in \mathcal{Q}^{\text{frac}}(G_{s_\rho}^\epsilon)$. We thus have that $\mathcal{Q}^{\text{frac}}(C_{s_\rho}^\epsilon) \subseteq \mathcal{Q}^{\text{frac}}(H_{s_\rho}^\epsilon) \subseteq \mathcal{Q}^{\text{frac}}(G_{s_\rho}^\epsilon)$, which proves that (5.13) holds.

Proof of (5.12). We will prove that

$$\mathcal{Q}(C_{s_\rho}^\epsilon) = \mathcal{Q}(H_{s_\rho}^\epsilon) = \mathcal{Q}(G_{s_\rho}^\epsilon).$$

Note that the reasoning used to deal with (5.13) can be directly applied to prove that

$$\mathcal{Q}(C_{s_\rho}^\epsilon) \subseteq \mathcal{Q}(H_{s_\rho}^\epsilon) \subseteq \mathcal{Q}(G_{s_\rho}^\epsilon).$$

Therefore, it suffices to show that $\mathcal{Q}(G_{s_\rho}^\epsilon) \subseteq \mathcal{Q}(C_{s_\rho}^\epsilon)$. Suppose by contradiction that this is not the case. Then there must exist some binary vector $q \in \{0, 1\}^n$ such that $q \in \mathcal{Q}(G_{s_\rho}^\epsilon)$ but $q \notin \mathcal{Q}(C_{s_\rho}^\epsilon)$. In particular, the condition $q \notin \mathcal{Q}(C_{s_\rho}^\epsilon)$ implies that

$$\exists e \in \mathcal{E}(C_{s_\rho}^\epsilon) \text{ such that } \sum_{i \in e} q^{(i)} > 1.$$

Given that q is a binary vector, this implies that

$$\exists i, j \in e, i \neq j \text{ such that } q^{(i)} = q^{(j)} = 1.$$

These two indices i, j form an edge in $G_{s_\rho}^\epsilon$, which in turn implies that q cannot be a packing of $G_{s_\rho}^\epsilon$.

This contradiction allows us to conclude that $\mathcal{Q}(G_{s_\rho}^\epsilon) \subseteq \mathcal{Q}(C_{s_\rho}^\epsilon)$, and therefore

$$\mathcal{Q}(C_{s_\rho}^\epsilon) = \mathcal{Q}(H_{s_\rho}^\epsilon) = \mathcal{Q}(G_{s_\rho}^\epsilon).$$

This concludes the proof.

B.2.2 Proof of Lemma 5.2.3

Lemma 5.2.3 (Conformal hypergraphs).

Let us consider a p -norm with $p > 1$, and $\epsilon > 0$. Let also $S = \{(x_i, y_i)\}_{i \in [n]}$ be an arbitrary set of points from $\mathcal{X} \times \mathcal{Y}$. The assertions a) and b) below are equivalent.

$$\text{a) } \forall \omega \in \mathbb{R}_+^n, \quad \text{FP}(H_S^\epsilon, \omega) = \text{FP}(C_S^\epsilon, \omega)$$

$$\text{b) } \text{Every clique in } G_S^\epsilon \text{ is a hyperedge in } H_S^\epsilon.$$

Proof. b) $\implies$ a). Let $\omega \in \mathbb{R}_+^n$. From Lemma 5.2.2, we have that $\text{FP}(H_S^\epsilon, \omega) \geq \text{FP}(C_S^\epsilon, \omega)$. Furthermore, for any fractional packing of H_S^ϵ with characteristic vector

$q \in \mathcal{Q}^{\text{frac}}(H_S^\epsilon)$, assertion **b**) implies that $q \in \mathcal{Q}^{\text{frac}}(C_S^\epsilon)$. Thus, $\mathcal{Q}^{\text{frac}}(H_S^\epsilon) \subseteq \mathcal{Q}^{\text{frac}}(C_S^\epsilon)$ and then $\text{FP}(H_S^\epsilon, \omega) \leq \text{FP}(C_S^\epsilon, \omega)$ which concludes this first implication.

a) $\implies$ b). By contradiction, suppose there exists a clique c' in G_S^ϵ such that c' is not a hyperedge in H_S^ϵ (i.e. $c' \notin \mathcal{E}(H_S^\epsilon)$). Let c be the maximal clique in G_S^ϵ containing c' (i.e. $c \in \mathcal{E}(C_S^\epsilon)$). Note that, as H_S^ϵ is downward closed, we have that $c \notin \mathcal{E}(H_S^\epsilon)$. Consider the probability vector $\omega \in \Delta^n$ where $\omega^{(i)} = \frac{1}{|c|} \mathbb{1}_{\{i \in c\}}$ for all $i \in [n]$. Note that ω is an optimal solution for the fractional set packing problem over (C_S^ϵ, ω) as $\omega^T \omega = \frac{1}{|c|}$ and for any $q \in \mathcal{Q}^{\text{frac}}(C_S^\epsilon)$ we have

$$\omega^T q = \frac{1}{|c|} \sum_{i \in c} q^{(i)} \leq \frac{1}{|c|} \quad (\text{as } c \in \mathcal{E}(C_S^\epsilon)).$$

We now construct a feasible solution q for the fractional set packing problem over (H_S^ϵ, ω) such that $\omega^T q > \omega^T \omega$, which would imply that $\text{FP}(H_S^\epsilon, \omega) \geq \omega^T q > \omega^T \omega = \text{FP}(C_S^\epsilon, \omega)$, hence contradicting **a**). Let $\mathcal{E}_c := \{e \in \mathcal{E}(H_S^\epsilon) : e \subseteq c\}$ and consider $e^* \in \text{argmax}\{|e| : e \in \mathcal{E}_c\}$. Note that $e^* \subset c$ as $c \notin \mathcal{E}_c$, which implies

$$\sum_{i \in e^*} \omega^{(i)} < \sum_{i \in c} \omega^{(i)} = 1. \quad (\text{B.6})$$

Let $i^* \in e^*$ and define $q \in [0, 1]^n$ such that $q^{(i)} = \omega^{(i)}$ for $i \neq i^*$ and $q_{i^*} = 1 - \sum_{i \in e^* \setminus \{i^*\}} \omega^{(i)}$. Notice that

$$\begin{aligned} \omega^T q &= \frac{1}{|c|} \sum_{i \in c} q^{(i)} = \frac{1}{|c|} \left(\sum_{i \in c \setminus e^*} q^{(i)} + \sum_{i \in e^*} q^{(i)} \right) \\ &= \frac{1}{|c|} \left(\sum_{i \in c \setminus e^*} \omega^{(i)} + 1 \right) \\ &> \frac{1}{|c|} \left(\sum_{i \in c \setminus e^*} \omega^{(i)} + \sum_{i \in e^*} \omega^{(i)} \right) \quad (\text{by (B.6)}) \\ &= \omega^T \omega. \end{aligned}$$

In order to conclude the proof, we need to check that q is feasible, i.e. $q \in \mathcal{Q}^{\text{frac}}(H_S^\epsilon)$. Let us consider an arbitrary $e \in \mathcal{E}(H_S^\epsilon)$. Note that the following is true given the definition of q :

$$\sum_{i \in e} q^{(i)} = \sum_{i \in e \cap c} q^{(i)}.$$

Now let us consider two cases, depending on whether i^* belongs to e or not. If $i^* \in e \cap c$, then

$$\begin{aligned} \sum_{i \in e \cap c} q^{(i)} &= 1 - \sum_{i \in e^* \setminus \{i^*\}} q^{(i)} + \sum_{\substack{i \in e \cap c \\ i \neq i^*}} q^{(i)} \\ &= 1 - \frac{|e^*| - 1}{|c|} + \frac{|e \cap c| - 1}{|c|} \end{aligned}$$

$$\begin{aligned}
&= 1 - \frac{|e^*| - |e \cap c|}{|c|} \\
&\leq 1. \quad \text{(By definition of } e^*)
\end{aligned}$$

Otherwise, if $i^* \notin e \cap c$, then

$$\sum_{i \in e \cap c} q^{(i)} = \frac{|e \cap c|}{|c|} \leq 1.$$

This concludes the proof.

B.2.3 Proof of Lemma 5.2.4

Lemma 5.2.4 (Perfect graphs).

Let us consider a p -norm with $p > 1$, and $\epsilon > 0$. Let also $S = \{(x_i, y_i)\}_{i \in [n]}$ be an arbitrary set of points from $\mathcal{X} \times \mathcal{Y}$. The assertions a) and b) below are equivalent.

$$a) \quad \forall \omega \in \mathbb{R}_+^n, \quad \text{FP}(C_S^\epsilon, \omega) = \text{IP}(C_S^\epsilon, \omega).$$

$$b) \quad G_S^\epsilon \text{ is perfect.}$$

Proof. Let A be the $m \times n$ edge-incidence matrix of the hypergraph C_S^ϵ . Recall the linear program formulation of the fractional set packing problem

$$\begin{aligned}
\text{FP}(C_S^\epsilon, \omega) = \max_{q \in [0,1]^n} & \quad \omega^T q \\
\text{s.t.} & \quad Aq \leq \mathbf{1}_m
\end{aligned} \tag{B.7}$$

By [Chu+03, Theorem 4.1], the linear program in (B.7) has an integral optimum solution for every objective function $\omega \in \mathbb{R}_+^n$ if and only if the matrix A is the edge-incidence matrix of the clique hypergraph of a perfect graph.

As we already assumed that A is the incidence matrix of C_S^ϵ , which is the clique hypergraph of G_S^ϵ , then the only condition to ensure the existence of an integral optimum solution for every objective function $\omega \in \mathbb{R}_+^n$ for $\text{FP}(C_S^\epsilon, \omega)$ is that G_S^ϵ is perfect.

B.2.4 Proof of Theorem 5.2.5

Theorem 5.2.5.

Let us consider a p -norm with $p > 1$, and $\epsilon > 0$. Let also $S = \{(x_i, y_i)\}_{i \in [n]}$ be an arbitrary set of points from $\mathcal{X} \times \mathcal{Y}$. There exists a distribution ρ over $\mathcal{X} \times \mathcal{Y}$ with support

$s_\rho = S$ such that $\text{rg}(\rho) > 0$ if and only if at least one of the following assertions holds true:

a) H_S^ϵ is not conformal (i.e., there exists a clique in G_S^ϵ that is not a hyperedge in H_S^ϵ).

b) G_S^ϵ is not perfect (i.e., it contains at least one odd hole or one odd anti-hole).

Proof. First implication. Let $\mu \in \mathcal{P}(\mathcal{X} \times \mathcal{Y})$ be the distribution with support s_ρ and probability vector ω_ρ such that $\text{rg}(\mu, \epsilon) > 0$. By the decomposition of the randomization gap, one of either (5.14) or (5.15) has to be positive.

If (5.14) is positive, i.e. $\text{FP}(H_{s_\rho}^\epsilon, \omega_\rho) - \text{FP}(C_{s_\rho}^\epsilon, \omega_\rho) > 0$, then by Lemma 5.2.3, there exists a clique in $G_{s_\rho}^\epsilon$ that is not a hyperedge of $H_{s_\rho}^\epsilon$. This means that $H_{s_\rho}^\epsilon$ is not conformal.

If (5.15) is positive, i.e. $\text{FP}(C_{s_\rho}^\epsilon, \omega_\rho) - \text{IP}(C_{s_\rho}^\epsilon, \omega_\rho) > 0$, then by Lemma 5.2.4, the graph $G_{s_\rho}^\epsilon$ is not perfect, which means that it has either an odd hole or an odd anti-hole.

Second implication. If a) holds and there is a clique of G_S^ϵ that is not a hyperedge of H_S^ϵ , then by Lemma 5.2.3 there must exist some vector ω such that $\text{FP}(H_S^\epsilon, \omega) \neq \text{FP}(C_S^\epsilon, \omega)$. By Lemma 5.2.2, it is always true that $\text{FP}(H_S^\epsilon, \omega) \geq \text{FP}(C_S^\epsilon, \omega)$, which implies that $\text{FP}(H_S^\epsilon, \omega) - \text{FP}(C_S^\epsilon, \omega) > 0$.

If b) holds and G_S^ϵ is not perfect, by Lemma 5.2.4, there must exist some vector ω such that $\text{FP}(C_S^\epsilon, \omega) \neq \text{IP}(C_S^\epsilon, \omega)$. Given that $\text{FP}(C_S^\epsilon, \omega) \geq \text{IP}(C_S^\epsilon, \omega)$, we have that $\text{FP}(C_S^\epsilon, \omega) - \text{IP}(C_S^\epsilon, \omega) > 0$.

Define μ as the distribution with support on S and probability vector ω such that either $\text{FP}(H_S^\epsilon, \omega) - \text{FP}(C_S^\epsilon, \omega) > 0$ or $\text{FP}(C_S^\epsilon, \omega) - \text{IP}(C_S^\epsilon, \omega) > 0$. Then, we have that:

$$\text{rg}(\mu, \epsilon) = \text{FP}(C_S^\epsilon, \omega) - \text{IP}(C_S^\epsilon, \omega) + \text{FP}(H_S^\epsilon, \omega) - \text{FP}(C_S^\epsilon, \omega) > 0.$$

B.2.5 Proof of Corollary 5.2.7

Corollary 5.2.7.

Consider the ℓ_∞ norm, $\epsilon > 0$ and a discrete distribution ρ over $\mathcal{X} \times \mathcal{Y}$ with clique hypergraph $C_{s_\rho}^\epsilon = (V, \mathcal{E})$. Let $G' = (V', E')$ be an induced subgraph of $G_{s_\rho}^\epsilon$. If G' is a clique, then $V' \in \mathcal{E}$. Thus, for any $\omega \in \mathbb{R}_+^{|V|}$:

$$\text{FP}(H_{s_\rho}^\epsilon, \omega) = \text{FP}(C_{s_\rho}^\epsilon, \omega).$$

Proof. Let $\{(x_i, y_i)\}_{i \in [n]} \subset \mathcal{X} \times \mathcal{Y}$ be the support of μ such that $1, \dots, n'$ are the indices in V' with $n' = |V'|$. Suppose that G' is a clique. Recall that $\mathcal{X} \subseteq \mathbb{R}^d$. For any $j \in [d]$, consider a permutation $\sigma_j : [n'] \rightarrow [n']$ that satisfies $x_{\sigma_j(1)}^{(j)} \leq \dots \leq x_{\sigma_j(n')}^{(j)}$. Then, given that $\{\sigma_j(1), \sigma_j(n')\} \in E'$, we have that $B_\infty(x_{\sigma_j(1)}, \epsilon) \cap B_\infty(x_{\sigma_j(n')}, \epsilon) \neq \emptyset$, which implies

$$I_j := [x_{\sigma_j(n')}^{(j)} - \epsilon, x_{\sigma_j(1)}^{(j)} + \epsilon] \neq \emptyset, \quad \text{for any } j \in [d].$$

Let $z \in \mathbb{R}^d$ be such that $z^{(j)} \in I_j$ for every $j \in [d]$. Given that for any $j \in [d]$, $z^{(j)}$ satisfies the inequality $x_{\sigma_j(n')}^{(j)} - \epsilon \leq z^{(j)} \leq x_{\sigma_j(1)}^{(j)} + \epsilon$ and that $x_{\sigma_j(1)}^{(j)} \leq \dots \leq x_{\sigma_j(n')}^{(j)}$, we can conclude that:

$$\forall i \in V', \quad \forall j \in [d], \quad x_i^{(j)} - \epsilon \leq z^{(j)} \leq x_i^{(j)} + \epsilon.$$

In other words, we have that $z \in B_\infty(x_i, \epsilon)$ for all $i \in V'$. This implies that the intersection of the ϵ -balls is non-empty, i.e., $\bigcap_{i \in V'} B_\infty(x_i, \epsilon) \neq \emptyset$. Since G' is a clique of the conflict graph, for any $i, j \in V'$ we have that $\{i, j\} \in E'$ and thus $y_i \neq y_j$. Therefore, V' is a hyperedge in $H_{s_\rho}^\epsilon$.

B.2.6 Constructions of specific structures for different ℓ_p norms

Example B.2.2 (Non-conformal conflict hypergraph in the ℓ_2 norm.).

Let us consider the ℓ_2 norm. Consider ρ the uniform distribution over the canonical basis of $\mathbb{R}^K$, i.e., $s_\rho = \{(b_1, 1), \dots, (b_K, K)\} \subset \mathbb{R}^K \times [K]$ and $\omega_\rho = \frac{1}{K} \mathbf{1}_K$.

Intersection of ϵ -balls for subsets of points. For any subset $S \subseteq [K]$ such that $|S| = m$, we denote by b_S the average of the canonical vectors indexed by S , i.e.,

$$b_S := \frac{1}{m} \sum_{i \in S} b_i.$$

Furthermore, we can prove the following equivalence (see below):

$$\bigcap_{i \in S} B_2(b_i, \epsilon) \neq \emptyset \iff b_S \in \bigcap_{i \in S} B_2(b_i, \epsilon). \quad (\text{B.8})$$

Simple calculations tell us that, for any S of size m and $i \in [n]$, the Euclidean distance between b_S and b_i is exactly $\sqrt{\frac{m-1}{m}}$. Hence, using (B.8), we have

$$\bigcap_{i \in S} B_2(b_i, \epsilon) \neq \emptyset \iff \epsilon \geq \sqrt{\frac{m-1}{m}}. \quad (\text{B.9})$$

The conflict hypergraph is not conformal. Let us set $\epsilon = \frac{1}{\sqrt{2}}$. We have that

$$\forall i, j \in [K], \quad \|b_i - b_j\|_2 = 2\epsilon,$$

and therefore $\{i, j\} \in \mathcal{E}(H_{s_\rho}^\epsilon)$ for all $i, j \in [K]$. This means that the subset of vertices $[K]$ induces a clique in $G_{s_\rho}^\epsilon$.

On the other hand, for any subset $S \subset [K]$ of size m greater than 2, we have that $\sqrt{\frac{m-1}{m}} > \epsilon$, so by (B.9) we have that

$$\forall S \subset [K], \quad |S| > 2 \implies \bigcap_{i \in S} B_2(b_i, \epsilon) = \emptyset,$$

which implies that no subset of size greater than 2 can be a hyperedge of $H_{s_\rho}^\epsilon$. In particular, the subset $[K]$, which constitutes a clique in $G_{s_\rho}^\epsilon$, is not a hyperedge of $H_{s_\rho}^\epsilon$. Thus, $H_{s_\rho}^\epsilon$ is not conformal.

Proof of (B.8). Now we proceed to prove the result stated in (B.8). We will do so by contradicting the fact that the mean minimizes the sum of squared distances. One direction is obvious, so we are going to prove that if the intersection $\cap_{i \in S} B_2(b_i, \epsilon)$ is non-empty, then the average b_S must belong to it.

Suppose by contradiction that $b_S \notin \cap_{i \in S} B_2(b_i, \epsilon)$ but that $\cap_{i \in S} B_2(b_i, \epsilon) \neq \emptyset$. Then, there exists an index $i^* \in S$ such that $b_S \notin B_2(b_{i^*}, \epsilon)$. This implies that

$$\|b_S - b_{i^*}\|_2 > \epsilon. \quad (\text{B.10})$$

Note that the distance from b_S to any b_i , $i \in S$ is exactly $\sqrt{\frac{m-1}{m}}$, so (B.10) implies the following:

$$\forall i \in S, \quad \|b_S - b_i\|_2 > \epsilon. \quad (\text{B.11})$$

By (B.11), we have that

$$\sum_{i \in S} \|b_S - b_i\|_2^2 > m\epsilon^2. \quad (\text{B.12})$$

Now, from the assumption that the intersection is non-empty, take any $\bar{x} \in \cap_{i \in S} B_2(b_i, \epsilon) \neq \emptyset$. Note that this implies that

$$\forall i \in S, \quad \|\bar{x} - b_i\|_2^2 \leq \epsilon^2. \quad (\text{B.13})$$

Recall that the average b_S has the property of minimizing the sum of squared norms, i.e.

$$b_S \in \operatorname{argmin}_{x \in \mathbb{R}^K} \sum_{i \in S} \|x - b_i\|_2^2. \quad (\text{B.14})$$

However, by (B.12) and (B.13) we get that

$$\sum_{i \in S} \|\bar{x} - b_i\|_2^2 \leq m\epsilon^2 < \sum_{i \in S} \|b_S - b_i\|_2^2.$$

This contradicts (B.14).

Example B.2.3 (Odd anti-holes with the ℓ_∞ norm).

The anti-hole of size 5 is isomorphic to the hole of size 5, which is always possible to build. Let us see that for sizes greater than 5, it remains possible to build odd anti-holes.

Consider the following set of points in $\mathbb{R}^7$:

$$\begin{aligned} x_1 &= (0, 0.2, 0.3, 0.4, 0.5, 0.6, 1) \\ x_2 &= (1, 0, 0.3, 0.4, 0.5, 0.6, 0.7) \\ x_3 &= (0.1, 1, 0, 0.4, 0.5, 0.6, 0.7) \\ x_4 &= (0.1, 0.2, 1, 0, 0.5, 0.6, 0.7) \\ x_5 &= (0.1, 0.2, 0.3, 1, 0, 0.6, 0.7) \\ x_6 &= (0.1, 0.2, 0.3, 0.4, 1, 0, 0.7) \\ x_7 &= (0.1, 0.2, 0.3, 0.4, 0.5, 1, 0) \end{aligned}$$

Let $\epsilon = 0.5 - 0.01$. Then it can be seen that the following properties hold:

- $\forall i \in [7], B_\infty(x_i, \epsilon) \cap B_\infty(x_{(i-1) \bmod 7}, \epsilon) = B_p(x_i, \epsilon) \cap B_p(x_{(i+1) \bmod 7}, \epsilon) = \emptyset$
- $\forall i, j \in [7]$ such that $i < j$ and $j - i \neq 1 \bmod 7$, $B_p(x_i, \epsilon) \cap B_p(x_j, \epsilon) \neq \emptyset$

In other words, these points form an anti-hole of size 7. We conclude that anti-holes of any size exist for a sufficiently large dimension.

B.3 Constructing discrete distributions with large randomization gap

The proof of Theorem 5.3.1, which states that there exist discrete distributions with randomization gap arbitrarily close to $1/2$, relies on two key elements: First, the construction of a non-perfect graph for which the value of the set packing problem and its fractional counterpart differ significantly, and second, the representation of

this difference as the randomization gap of a distribution. For the first component, in Appendix B.3.2, we establish Corollary B.3.10, which provides a triangle-free construction based on the iterative procedure presented in [CCD93]. This will be crucial for the next part. For the second component, we demonstrate in Appendix B.3.1 through Lemma B.3.5 that any loopless graph is isomorphic to the conflict graph of a distribution. Furthermore, if the graph is triangle-free, the representation as a randomization gap of a certain data distribution is amenable. Combining both results, we prove Theorem 5.3.1 in Appendix B.3.3.

B.3.1 Every graph is the conflict graph of a distribution

In Lemmas B.3.2 and B.3.3 we prove that for the ℓ_∞ and ℓ_p norms (with $p \in (1, \infty)$) respectively, any graph G can be constructed by considering the overlap of some ϵ -balls. This corresponds to the second condition in the definition of the conflict graph (Definition 5.1.1). Building on this, we utilize the chromatic number (Definition B.3.4) to restrict the overlap of ϵ -balls to points from different classes (satisfying the first condition in Definition 5.1.1). Consequently, we prove Lemma B.3.5, which states that G is isomorphic to the conflict graph $G_{s_p}^\epsilon$ of a distribution μ . Furthermore, if G is triangle-free, then it is isomorphic to the loopless version of the conflict hypergraph $H_{s_p}^\epsilon$. This implies that the values of their respective (fractional) set packing problems coincide.

Lemma B.3.1 establishes a straightforward equivalence that will be used in the proofs throughout this subsection.

Lemma B.3.1 (Restatement of intersecting neighborhoods).

Let us consider an ℓ_p norm with $p \in (1, \infty]$ and $\epsilon > 0$. The following assertion holds true:

$$\forall x, x' \in \mathcal{X}, \quad B_p(x, \epsilon) \cap B_p(x', \epsilon) \neq \emptyset \iff \|x - x'\|_p \leq 2\epsilon$$

Proof. First, suppose that $B_p(x, \epsilon) \cap B_p(x', \epsilon) \neq \emptyset$ and take any $z \in B_p(x, \epsilon) \cap B_p(x', \epsilon)$. Then, by the triangle inequality

$$\|x - x'\|_p = \|x - z + z - x'\|_p \leq \|x - z\|_p + \|z - x'\|_p \leq \epsilon + \epsilon = 2\epsilon$$

For the other direction, suppose that $\|x - x'\|_p \leq 2\epsilon$ and consider the point $z = \frac{x+x'}{2}$. Then,

$$\|x - z\|_p = \left\| x - \frac{x+x'}{2} \right\|_p = \left\| \frac{x-x'}{2} \right\|_p = \frac{1}{2} \|x - x'\|_p \leq \frac{1}{2} 2\epsilon = \epsilon$$

Thus, $z \in B_p(x, \epsilon)$. An analogous argument exchanging x by x' yields that $z \in B_p(x', \epsilon)$.

As $z \in B_p(x, \epsilon) \cap B_p(x', \epsilon)$, we conclude that $B_p(x, \epsilon) \cap B_p(x', \epsilon) \neq \emptyset$.

Now we restate and prove lemmas related to the cubicity and sphericity of graphs. Basically, these results show for any p -norm, $\epsilon > 0$ and graph G , we can think of G as the intersection graph of some set of $|G|$ points. This will be the first step in the construction of discrete distributions, as it provides the support of the distribution in such a way that the conflicts between points are exactly those represented in G .

Lemma B.3.2 (Cubicity, restated from [Rob69]).

Let $G = (V, E)$ be any graph with $n \in \mathbb{N}^*$ vertices. Let us consider the ℓ_∞ norm and $\epsilon > 0$. There exist $d \in \mathbb{N}^*$ and a set of points $x_1, \dots, x_n$ from $\mathbb{R}^d$ such that the following holds:

$$\forall i, j \in V, \quad \{i, j\} \in E \iff B_\infty(x_i, \epsilon) \cap B_\infty(x_j, \epsilon) \neq \emptyset$$

Proof. Following the remark given in [Rob69, Section 2], we build coordinate functions to embed the n vertices of G into $\mathbb{R}^n$. For any $i \in V$, consider the following vector $x_i = (x_i^{(1)}, x_i^{(2)}, \dots, x_i^{(n)}) \in \mathbb{R}^n$, where:

$$\forall j \in V, \quad x_i^{(j)} = \begin{cases} 0, & \text{if } j = i \\ 0.9\epsilon, & \text{if } i \neq j \text{ and } \{i, j\} \in E \\ 1.1\epsilon, & \text{if } i \neq j \text{ and } \{i, j\} \notin E \end{cases}$$

Now, we distinguish between two key cases.

Case 1: If $\{i, j\} \in E$, the distance $\|x_i - x_j\|_\infty$ is strictly less than ϵ . We need to verify that for all $k \in [n]$, $|x_i^{(k)} - x_j^{(k)}| < \epsilon$. Let us consider each coordinate k :

- If $k = i$, then:

$$|x_i^{(k)} - x_j^{(k)}| = |0 - x_j^{(i)}| = x_j^{(i)} = 0.9\epsilon,$$

- Similarly, if $k = j$, then:

$$|x_i^{(k)} - x_j^{(k)}| = |0 - x_i^{(j)}| = x_i^{(j)} = 0.9\epsilon,$$

- If $k \neq i$ and $k \neq j$, then both $x_i^{(k)}$ and $x_j^{(k)}$ are either 0.9ϵ or 1.1ϵ , depending on whether $\{i, k\} \in E$ and $\{j, k\} \in E$. In any case, we have:

$$|x_i^{(k)} - x_j^{(k)}| \leq |1.1\epsilon - 0.9\epsilon| = 0.2\epsilon < \epsilon.$$

Thus, for all coordinates $k \in [n]$, we have $|x_i^{(k)} - x_j^{(k)}| < \epsilon$. Therefore, $\|x_i - x_j\|_\infty < \epsilon$, as desired.

Case 2: If $\{i, j\} \notin E$, the distance $\|x_i - x_j\|_\infty$ is strictly greater than ϵ .

- If $k = i$ or $k = j$, then:

$$|x_i^{(k)} - x_j^{(k)}| = 1.1\epsilon > \epsilon$$

Thus, if $\{i, j\} \notin E$ then $\|x_i - x_j\|_\infty > \epsilon$.

Conclusion. We have shown that $\|x_i - x_j\|_\infty \leq \epsilon$ if and only if $\{i, j\} \in E$.

By replacing ϵ by 2ϵ and using Lemma B.3.1, we obtain that:

$$\{i, j\} \in E \iff B_\infty(x_i, \epsilon) \cap B_\infty(x_j, \epsilon) \neq \emptyset.$$

This concludes the proof.

We now prove a result related to the concept of sphericity [Fis83] of a graph, but generalized to any p -norm with $p \in (1, \infty)$.

Lemma B.3.3 (Sphericity for any p -norm).

Let $G = (V, E)$ be any graph with $n \in \mathbb{N}^*$ vertices. Let us consider an ℓ_p norm with $p \in (1, \infty)$ and $\epsilon > 0$. There exist $d \in \mathbb{N}^*$ and a set of points $x_1, \dots, x_n$ from $\mathbb{R}^d$ such that for any $i, j \in V$ such that $i \neq j$, the following holds:

$$\{i, j\} \in E \iff B_p(x_i, \epsilon) \cap B_p(x_j, \epsilon) \neq \emptyset. \quad (\text{B.15})$$

Proof. Let $G = (V, E)$ be any graph with $n \in \mathbb{N}^*$ vertices and $m \in \left[\frac{(n-1)n}{2}\right]$ edges. For any $i \in V$, we denote by $\deg_i$ the degree of i , i.e., the number of neighbors of i in G . Set $d = n + m$, and define $x_1, \dots, x_n \in \mathbb{R}^{n+m}$ as follows.

Definition of the vectors. For any $i \in [n]$ and $k \in [d]$, define $x_i^{(k)}$ as

$$x_i^{(k)} = \begin{cases} 1, & \text{if } k \in [m] \text{ and } i \in e_k \\ (n - \deg_i)^{\frac{1}{p}}, & \text{if } k = m + i \\ 0, & \text{otherwise.} \end{cases}$$

Let us now consider $i, j \in V$ such that $i \neq j$, we can distinguish two cases.

Case 1. If $\{i, j\} \notin E$ then $\nexists k \in [m]$ such that $x_i^{(k)} = x_j^{(k)} = 1$. Furthermore, as $i \neq j$, we have $m + i \neq m + j$. Hence, we have $\|x_i - x_j\|_p = (\deg_i + \deg_j + (n - \deg_i) + (n - \deg_j))^{\frac{1}{p}} = (2n)^{\frac{1}{p}}$.

Case 2. If $\{i, j\} \in E$, then $\exists! k \in [m]$ such that $x_i^{(k)} = x_j^{(k)} = 1$. Furthermore, $m + i \neq m + j$. Hence, we have $\|x_i - x_j\|_p = (\deg_i + \deg_j - 1 + (n - \deg_i) + (n - \deg_j))^{\frac{1}{p}} = (2n - 1)^{\frac{1}{p}}$.

Finally, multiplying each x_i by a constant $2\epsilon \cdot (2n-1)^{-\frac{1}{p}}$ we obtain that

$$\|x_i - x_j\|_p = \begin{cases} 2\epsilon \cdot \left(\frac{2n}{2n-1}\right)^{\frac{1}{p}}, & \text{if } \{i, j\} \notin E \\ 2\epsilon, & \text{if } \{i, j\} \in E \end{cases}$$

As $\left(\frac{2n}{2n-1}\right)^{\frac{1}{p}} \geq 1$ for any $n \in \mathbb{N}^*$ and $p \in (1, \infty)$, we finally get that

$$\|x_i - x_j\|_p \leq 2\epsilon \iff \{i, j\} \in E.$$

By Lemma B.3.1, we conclude that (B.15) holds.

Now that we know that for any graph G we can find a set of points S such that $G \simeq G_S^\epsilon$, we only need to include the labels of the points to be able to create a discrete distribution for a classification task. Interestingly, the condition that all points in conflict must be from different classes has a direct counterpart in graph theory: *vertex colorings*. We define below the chromatic number before proving Lemma B.3.5.

Definition B.3.4 (Chromatic number $\chi(G)$).

Given a graph $G = (V, E)$, a coloring of the vertices of G is a function $c : V \rightarrow \mathbb{N}$ such that

$$\forall \{i, j\} \in E, \quad c(i) \neq c(j).$$

The chromatic number of G , denoted $\chi(G)$, is the smallest number of colors needed to color the vertices of G . That is, $\chi(G)$ is the smallest M such that there exists a coloring $c : V \rightarrow [M]$.

Lemma B.3.5.

Let $G = (V, E)$ be any loopless graph with $n \in \mathbb{N}^*$ vertices. Let us consider an ℓ_p norm with $p \in (1, \infty]$ and $\epsilon > 0$. There exist $d, K \in \mathbb{N}$ and $S = \{(x_i, y_i)\}_{i \in [n]} \subset \mathbb{R}^d \times [K]$, such that $G_S^\epsilon \simeq G$, where the relation $\simeq$ denotes graph isomorphism. Moreover, if G is triangle-free, then G is isomorphic to the loopless version of H_S^ϵ and the number of classes satisfies $K \in \mathcal{O}\left(\sqrt{n/\log n}\right)$, where $n = |S|$.

Proof. Using either Lemma B.3.2 or Lemma B.3.3 for $p = \infty$ or $p \in (1, \infty)$ respectively, there exist $d \in \mathbb{N}^*$ and a set of points $\{x_i\}_{i \in [n]}$ in $\mathbb{R}^d$ such that for any $i, j \in V$ with $i \neq j$, the following holds:

$$\{i, j\} \in E \iff B_p(x_i, \epsilon) \cap B_p(x_j, \epsilon) \neq \emptyset. \quad (\text{B.16})$$

For the number of classes, take $K = \chi(G)$, the chromatic number of G . By definition of the chromatic number, there exists a K -coloring of G , i.e. a function $c: V \mapsto [K]$, such that for any $i, j \in V$

$$\{i, j\} \in E \implies c(i) \neq c(j). \quad (\text{B.17})$$

We define $S = \{(x_i, c(i))\}_{i \in [n]}$ and show that G_S^ϵ is isomorphic to G .

Proving isomorphism. As G and G_S^ϵ share the same set of vertices, we have to show that

$$\{i, j\} \in E \iff \{i, j\} \in E(G_S^\epsilon).$$

If $\{i, j\} \in E$, as G is loopless we have $i \neq j$, and therefore by (B.16) and (B.17) we know that $B_p(x_i, \epsilon) \cap B_p(x_j, \epsilon) \neq \emptyset$ and $c(i) \neq c(j)$. These two conditions imply that $\{i, j\} \in E(G_S^\epsilon)$ by definition of the conflict graph.

On the other hand, if $\{i, j\} \in E(G_S^\epsilon)$, we have that in particular, the ϵ -balls intersect, i.e. $B_p(x_i, \epsilon) \cap B_p(x_j, \epsilon) \neq \emptyset$. As G_S^ϵ is loopless, we have $i \neq j$, and this implies that $\{i, j\} \in E$ by (B.16).

Adding the triangle-free assumption. Note that we do not know the full structure of the conflict hypergraph H_S^ϵ , apart from the fact that the corresponding conflict graph G_S^ϵ is isomorphic to the graph G . However, if we add the assumption that G is triangle-free, we can show that H_S^ϵ is the loopless version of G . Suppose by contradiction that H_S^ϵ has a hyperedge $e \in \mathcal{E}(H_S^\epsilon)$ of size greater than 2. Then there would be a triplet of points x_i, x_j, x_k such that

$$B_p(x_i, \epsilon) \cap B_p(x_j, \epsilon) \cap B_p(x_k, \epsilon) \neq \emptyset.$$

This would imply the existence of the three edges $\{i, j\}$, $\{i, k\}$ and $\{j, k\}$ in G_S^ϵ , and therefore in G , which form a triangle. As G is triangle-free, we can conclude that H_S^ϵ cannot contain any hyperedge of size greater than 2. Thus, by definition of conflict graph G_S^ϵ , we conclude that G is isomorphic to the loopless version of H_S^ϵ .

Regarding the number of classes when G is triangle-free, we know that $K = \chi(G) \in \mathcal{O}\left(\sqrt{\frac{n}{\log n}}\right)$ (See [EH85; JT11; DI22]).

B.3.2 Constructing non-perfect graphs with large randomization gap using the fibration construction

Lemma B.3.5 ensures that any graph G can be materialized as the conflict graph of some discrete distribution μ . Moreover, if the G is triangle-free, then G is essentially the conflict hypergraph, which implies that we can compute the randomization gap of μ using only the (fractional) set packing problem over G . To prove Theorem 5.3.1 we thus only need to build a graph G for which we can control the gap between the value of the fractional set packing problem and the set packing problem. With this

objective in mind, we go over the construction proposed in [CCD93] of a graph that is triangle-free and without large independent sets. We restate and adapt their main results to our notation for completeness. The construction in [CCD93] is iterative. Starting from a graph G , the authors propose to build a new, larger graph H called its *fibration*, using 6 copies of G and connecting vertices between copies in a particular manner. This construction will preserve the property of being triangle-free, will increase the number of nodes by a factor of 6, but more importantly, it will at most increase the size of the largest independent set by a factor of 4. Let us formalize this.

Definition B.3.6 (Independence number $\alpha(G)$).

Given a graph G , the independence number of G , denoted $\alpha(G)$, is the size of a maximum independent set of G .

Definition B.3.7 (Fibration [CCD93]).

For any graph $G = (V(G), E(G))$, the fibration of G is the graph $H = (V(H), E(H))$ defined as follows:

1. $V(H) = V(G) \times [6]$.
2. $\forall i \in [6], \quad \{u, v\} \in E(G) \implies \{(u, i), (v, i)\} \in E(H)$.
3. $\forall i, j \in [6] \text{ with } j \equiv (i + 1) \pmod{6}, \quad \{u, v\} \in E(G) \implies \{(u, i), (v, j)\} \in E(H) \text{ and } \{(u, j), (v, i)\} \in E(H)$.
4. $\forall i, j \in [6] \text{ with } j \equiv (i + 3) \pmod{6}, \quad u \in V(G) \implies \{(u, i), (u, j)\} \in E(H)$.

There are two properties of the fibration that are important:

Lemma B.3.8 (Triangle-free property preservation, Lemma 1 [CCD93]).

If G is triangle-free and H is the fibration of G , then H is triangle-free.

Lemma B.3.9 (Bound on the maximum independent set, Lemma 2 in [CCD93]).

If H is the fibration of G , then $\alpha(H) \leq 4 \cdot \alpha(G)$.

To construct our graph of interest, we will iteratively apply the fibration operation, starting from an initial graph. Let G_0 be a fixed triangle-free graph, and let G_t

denote the graph obtained after applying the fibration operation t times to G_0 . Then, by Lemma B.3.8, we have the following properties:

$$\begin{aligned} G_t &\text{ is triangle-free} \\ |G_t| &= 6^t \cdot |G_0| \end{aligned} \tag{B.18}$$

$$\alpha(G_t) \leq 4^t \cdot \alpha(G_0) \tag{B.19}$$

For any graph G , the set packing problem with the weight vector $\mathbf{1}_{|G|}$ has a natural interpretation: it is exactly the size of the maximum independent set. In other words, for any graph G ,

$$\text{IP}(G, \mathbf{1}_{|G|}) = \alpha(G). \tag{B.20}$$

Coming back to the graph G_t , by (B.19) and (B.20), we can conclude that

$$\text{IP}(G_t, \mathbf{1}_{|G_t|}) \leq 4^t \alpha(G_0). \tag{B.21}$$

On the other hand, for any graph G , the vector $\frac{1}{2}\mathbf{1}_{|G|}$ is feasible for the fractional set packing problem because there are no hyperedges of size larger than 2, so any constraint in the definition of fractional packing involves at most 2 vertices. In other words, for any graph G ,

$$\text{FP}(G, \mathbf{1}_{|G|}) \geq \frac{1}{2} \mathbf{1}_{|G|}^T \mathbf{1}_{|G|} \geq \frac{|G|}{2}. \tag{B.22}$$

Now let us consider the normalized vector $\frac{1}{|G_t|}\mathbf{1}_{|G_t|}$. Clearly, we have that

$$\text{IP}(G_t, \frac{1}{|G_t|}\mathbf{1}_{|G_t|}) = \frac{1}{|G_t|} \text{IP}(G_t, \mathbf{1}_{|G_t|}) \tag{B.23}$$

$$\text{FP}(G_t, \frac{1}{|G_t|}\mathbf{1}_{|G_t|}) = \frac{1}{|G_t|} \text{FP}(G_t, \mathbf{1}_{|G_t|}) \tag{B.24}$$

By (B.22) and (B.24), we can lower bound the value of the fractional set packing problem over G_t as follows:

$$\text{FP}(G_t, \frac{1}{|G_t|}\mathbf{1}_{|G_t|}) \geq \frac{1}{2}. \tag{B.25}$$

Now we upper bound the value of the set packing problem for the t -fibration graph:

$$\text{IP}(G_t, \frac{1}{|G_t|}\mathbf{1}_{|G_t|}) = \frac{1}{|G_t|} \text{IP}(G_t, \mathbf{1}_{|G_t|}) \quad (\text{By (B.23)}) \tag{B.26}$$

$$\leq \frac{4^t}{|G_t|} \alpha(G_0) \quad (\text{By (B.21)}) \quad (\text{B.27})$$

$$\leq \frac{4^t}{6^t} \alpha(G_0) \quad (\text{By (B.18)}) \quad (\text{B.28})$$

From (B.25) and (B.28) we can immediately deduce the following:

Corollary B.3.10.

For any graph G_0 , let G_t be the graph obtained by applying the fibration operation t times, starting from G_0 . Then, for any $\delta > 0$, the following holds:

$$t > \frac{\log(\delta/\alpha(G_0))}{\log(2/3)} \implies \text{FP}(G_t, \frac{1}{|G_t|} \mathbf{1}_{|G_t|}) - \text{IP}(G_t, \frac{1}{|G_t|} \mathbf{1}_{|G_t|}) \geq \frac{1}{2} - \delta \quad (\text{B.29})$$

Proof. If $t > \frac{\log(\frac{\delta}{\alpha(G_0)})}{\log(\frac{2}{3})}$ then we have

$$\begin{aligned} t \log \frac{2}{3} &\leq \log \frac{\delta}{\alpha(G_0)} \\ \left(\frac{2}{3}\right)^t &\leq \frac{\delta}{\alpha(G_0)} \\ \left(\frac{2}{3}\right)^t \alpha(G_0) &\leq \delta \end{aligned}$$

Thus, by (B.28), $t > \frac{\log(\frac{\delta}{\alpha(G_0)})}{\log(\frac{2}{3})}$ implies $\text{IP}(G_t, \frac{1}{|G_t|} \mathbf{1}_{|G_t|}) \leq \delta$. Then, injecting (B.25), we obtain the desired result.

B.3.3 Proof of Theorem 5.3.1

We are ready to provide the full proof of Theorem 5.3.1, that will use the fibration operation with Lemma B.3.5 to produce discrete distributions with randomization gap arbitrarily close to $1/2$.

Theorem 5.3.1.

Fix any $\epsilon > 0$ and p -norm with $p > 1$. For any $\delta > 0$, there exist $d, K \in \mathbb{N}$ and a discrete distribution ρ over $\mathbb{R}^d \times [K]$ such that

$$\text{rg}(\rho) \geq 1/2 - \delta, \quad (5.16)$$

the conflict graph $G_{s_\rho}^\epsilon$ is not perfect and the conflict hypergraph $H_{s_\rho}^\epsilon$ is conformal. Furthermore, the number of classes satisfies $K \in \mathcal{O}\left(\sqrt{n/\log n}\right)$ with $n = |s_\rho|$.

Proof. For the fixed ϵ , p -norm and δ , take any triangle-free loopless graph G (for example, the 5-cycle C_5). Take any t that satisfies the condition in (B.29) from Corollary B.3.10 and consider G_t the t -fibration of G .

By Lemma B.3.8, G_t is triangle-free, and by Corollary B.3.10, we have that

$$\text{FP}(G_t, \frac{1}{|G_t|} \mathbf{1}_{|G_t|}) - \text{IP}(G_t, \frac{1}{|G_t|} \mathbf{1}_{|G_t|}) \geq \frac{1}{2} - \delta \quad (\text{B.30})$$

By Lemma B.3.5, as G_t is loopless, there exist $d, K \in \mathbb{N}$ and $S = \{(x_i, y_i)\}_{i \in [|G_t|]} \subset \mathbb{R}^d \times [K]$ such that $G_S^\epsilon \simeq G_t$. Moreover, as G_t is triangle-free, we have that H_S^ϵ without considering loops is isomorphic^a to G_t . This implies^a that

$$\forall \omega \in \mathbb{R}_+^n, \quad \text{FP}(H_S^\epsilon, \omega) = \text{FP}(G_S^\epsilon, \omega).$$

Define μ as the discrete distribution with support $s_\rho = S$ and uniform probability vector $\omega_\rho = \frac{1}{|G_t|} \mathbf{1}_{|G_t|}$. Given that $\text{IP}(H_{s_\rho}^\epsilon, \omega_\rho) = \text{IP}(G_{s_\rho}^\epsilon, \omega_\rho)$ by Lemma 5.2.2, we can conclude thanks to (B.30) that

$$\text{rg}(\mu, \epsilon) = \text{FP}(H_{s_\rho}^\epsilon, \omega_\rho) - \text{IP}(H_{s_\rho}^\epsilon, \omega_\rho) = \text{FP}(G_{s_\rho}^\epsilon, \omega_\rho) - \text{IP}(G_{s_\rho}^\epsilon, \omega_\rho) \geq \frac{1}{2} - \delta.$$

To check that $H_{s_\rho}^\epsilon$ is conformal, notice that as $G_{s_\rho}^\epsilon$ is triangle-free, the largest cliques of $G_{s_\rho}^\epsilon$ are the 2-edges, which are also hyperedges in $H_{s_\rho}^\epsilon$. Thus, all cliques from $G_{s_\rho}^\epsilon$, including the maximal ones, are hyperedges of $H_{s_\rho}^\epsilon$. Lastly, by Theorem 5.2.5, that $G_{s_\rho}^\epsilon$ is not perfect.

The fact that $K \in \mathcal{O}(\sqrt{\frac{n}{\log n}})$ where $n = |G_t|$ is also given by Lemma B.3.5 because G_t is triangle-free.

^aThe constraints associated to loops are not considered in the definition of a fractional packing as $\mathcal{Q}^{\text{frac}}(H_S^\epsilon) \subseteq [0, 1]^n$.

B.4 On the hardness of computing the randomization gap.

Here, we formally redefine the optimization and decision problems discussed in the paper. Our focus is to express these problems in the context of hypergraph-based set packing, providing clear formulations for both their optimization and decision versions. We then prove the co-NP-completeness of the Randomization Gap decision problem.

Definition of the Optimization Problems

Optimization Problem: Weighted Set Packing $\text{IP}(H, \omega)$

Input: A hypergraph $H = (V, \mathcal{E})$ and a weight vector $\omega \in \mathbb{Q}_+^{|V|}$

Output: $\sup_{q \in \mathcal{Q}(H)} \omega^T q$

Optimization Problem: *Fractional Weighted Set Packing* $\text{FP}(H, \omega)$

Input: A hypergraph $H = (V, \mathcal{E})$ and a weight vector $\omega \in \mathbb{Q}_+^{|V|}$

Output: $\sup_{q \in \text{ConvexHull}(\mathcal{Q}(H))} \omega^T q$

Decision Problems Associated to these Optimization Problems

Decision Problem: *Weighted Set Packing* $\text{IP}(H, \omega, \alpha)$

Input: A hypergraph $H = (V, \mathcal{E})$ and a weight vector $\omega \in \mathbb{Q}_+^{|V|}$, a rational number $\alpha \in \mathbb{Q}$

Output: YES if there exists a packing $q \in \mathcal{Q}(H)$ such that $\omega^T q \geq \alpha$, otherwise NO

Decision Problem: *Randomization Gap Problem* $\text{RG}(H, \omega, \alpha)$

Input: A hypergraph $H = (V, \mathcal{E})$ and a weight vector $\omega \in \mathbb{Q}_+^{|V|}$, a rational number $\alpha \in \mathbb{Q}$

Output: YES if and only if $\text{FP}(H, \omega) - \text{IP}(H, \omega) \geq \alpha$, otherwise NO

B.4.1 Reminder on NP-hardness and co-NP-hardness

Only *decision* problems are in **NP** or in **co-NP**.

- A decision problem is in **NP** if, given a "yes" instance¹ x , there exists a certificate (or witness) c that can be verified in polynomial time. Formally, the problem is in **NP** if there exists a polynomial-time verification algorithm $A(x, c)$ such that:

¹a YES instance is a input on which a correct algorithm should output YES

- If x is a "yes" instance, then there exists a certificate c such that $A(x, c)$ outputs yes.
 - Reciprocally, if x is a "no" instance, then for any certificate c , $A(x, c)$ will output no.
- A decision problem is in **co-NP** if, given a "no" instance x , there exists a certificate c that can be verified in polynomial time. Formally, the problem is in **co-NP** iff there exists a polynomial-time verification algorithm $A(x, c)$ such that:
 - If x is a "no" instance, then there exists a certificate c such that $A(x, c)$ outputs yes.
 - Reciprocally, if x is a "yes" instance, then for any certificate c , $A(x, c)$ will output no.

A decision problem is NP-complete if it belongs to the class NP *and* it is NP-hard. Same for co-NP.

B.4.2 Hardness of the RG decision problem

Theorem B.4.1.

The RG (H, ω, α) problem is co-NP complete

Proof. To show a problem is co-NP complete, we must first show it belongs to co-NP, and then that it is co-NP hard.

The problem belongs to co-NP. To show it belongs to co-NP, we must show that there exist certificates for all NO instances. For all NO instances (H, ω, α) we have $FP(H, \omega) - \alpha < IP(H, \omega)$. Here, there must exist a packing q (which will be our certificate) such that $FP(H, \omega) - \alpha < \omega^T q$. So for any NO instance, it suffices to pick any packing q satisfying this inequality to convince a verification algorithm that our instance is in fact a NO instance. Thus, the problem belongs to co-NP.

The problem is co-NP hard. To show it is co-NP-hard, we build a trivial reduction from the IP (H, ω, α) problem. Assume by contradiction that there exists a polynomial time algorithm $Alg(H, \omega, \alpha)$ able to solve the RG problem. Let us show that this would immediately imply the existence of a polynomial time algorithm to solve the IP problem, which contradicts the well known NP-hardness of IP. More precisely, Let D be the least common denominator of $\omega_1 \dots \omega_{|V|}$ and define the function $\text{floor}(z) =$

$\frac{1}{D} \lfloor z \times D \rfloor$. Let (H, ω, α') be an arbitrary instance of the IP problem. Define $\alpha = \text{FP}(H, \omega) - \text{floor}(\alpha')$. Observe that these conditions are equivalent:

$$\text{IP}(H, \omega, \alpha') = \text{No}$$

$$\text{IP}(H, \omega) < \alpha'$$

$$\text{IP}(H, \omega) \leq \text{floor}(\alpha')$$

$$\text{FP}(H, \omega) - \text{IP}(H, \omega) \geq \text{FP}(H, \omega) - \text{floor}(\alpha')$$

$$\text{RG}(H, \omega, \alpha) = \text{Yes}$$

Thus, we can run $\text{Alg}(H, \omega, \alpha)$ in polynomial time and return the opposite of the boolean answer to solve $\text{IP}(H, \omega, \alpha')$ also in polynomial time. This contradicts NP-hardness of IP. Thus, our problem is co-NP-Hard, thus co-NP-complete.

Experimental setup and additional experiments for Chapter 3

Contents

C.1	Synthetic experiments	135
C.1.1	ARC and EOL-PGD against two classifiers in the plane at the same distance from the origin	135
C.1.2	LCA, EOL-PGD and ARC with different number of classifiers with respect to the distribution of the bias of the linear classifiers	136
C.2	CIFAR-10 experiments	138
C.2.1	Main results	138
C.2.2	Comparing strong versions of the attacks	139
C.3	On adversarial training for mixtures of classifiers	141

C.1 Synthetic experiments

In this section we provide more details on the synthetic experiments presented in Section 3.2.3. The first experiment is the one in $\mathbb{R}^2$ with two classifiers that produced Figure 3.5 in which we vary the angle between the normal vectors of the two linear classifiers and leave their distance to the center point fixed. The second experiment is the one in which we sample m linear models in $\mathbb{R}^d$ with different values for the dimension d , and the linear classifiers are sampled with different distances to the center point.

C.1.1 ARC and EOL-PGD against two classifiers in the plane at the same distance from the origin

These are the parameters used for the experiment that produced Figure 3.5. The parameters of LCA are

- Norm: ℓ_2 .
- Attack budget ϵ : 1.

- Number of steps T for EOL-PDG: 100.

ARC in the binary linear setting has no parameters to tune

C.1.2 LCA, EOL-PGD and ARC with different number of classifiers with respect to the distribution of the bias of the linear classifiers

We tested EOL-PGD, ARC and LCA against randomly sampled mixtures of different sizes. To control the difficulty of the setting, we introduced the parameters α and β , which control the average and the variance of the distance from the center point x to the decision boundaries of the linear classifiers, respectively. Figures C.1.1 to C.1.3 are similar to Figure 3.12 in the main text but with other values of α and β , where it can be seen that the hardness of crafting an attack can change drastically depending on the configuration of the classifiers around the point. Recall that α is the average distance from the center point to the decision boundaries, and β is the variance of the distances. In all cases, note that the y -axis values are not the same, and have been adapted for visual purposes.

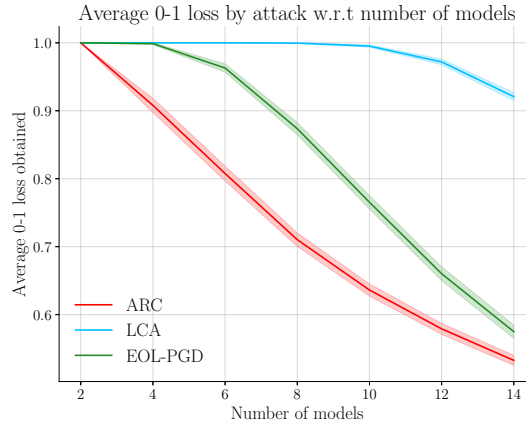


Fig. C.1.1.: Biases sampled from $-\mathcal{N}(0.25, 0.05)$. Classifiers are very close to the center point (low α), and they are almost all at the same distance (low β). In this particular case LCA performs very well, being almost always optimal. EOL-PGD also performs really well up to 6 models, which means that until that point, almost all linear classifiers are well aligned, and the common vulnerability region can be found by attacking the average loss. ARC on the other hand degrades sharply as the number of models increases.

In order to check that LCA was performing better than ARC and EOL-PGD, we computed the average over 100 runs of the ratio between the score obtained by each attack and the score that LCA obtained in the same configuration. Figure C.1.4 shows the evolution of such ratio with respect to the number of models. The colored

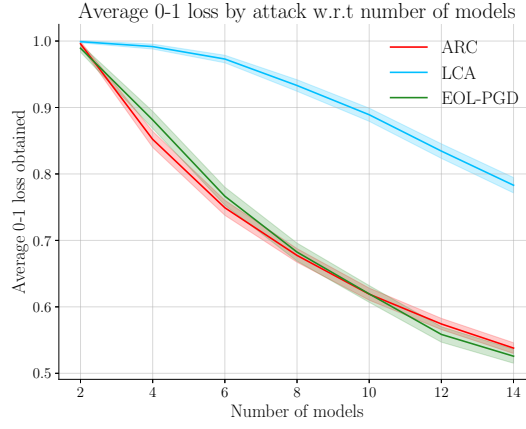


Fig. C.1.2.: Biases sampled from $-\mathcal{N}(0.25, 0.2)$. Classifiers are on average very close to the point but with a higher variance in their distance.

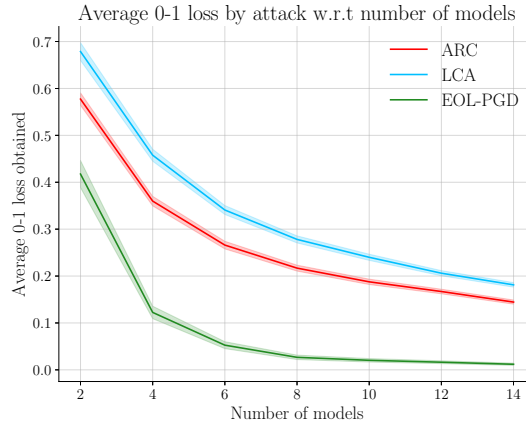


Fig. C.1.3.: Biases sampled from $-\mathcal{N}(0.75, 0.2)$. This configuration is hard because classifiers are close to the boundary of the ϵ -ball (far from x), and the probability of intersection is less than in the cases seen before. LCA is on average superior, but the gap between LCA, ARC and EOL-PGD is smaller.

area around the lines corresponds to a 99% confidence interval for the average ratio. We can see that LCA is superior by a rate that tends to stabilize as the number of models increase.

Here are the parameters used for all these experiments:

General parameters:

- Norm (threat model): ℓ_2 .
- Attack budget ϵ : 1.

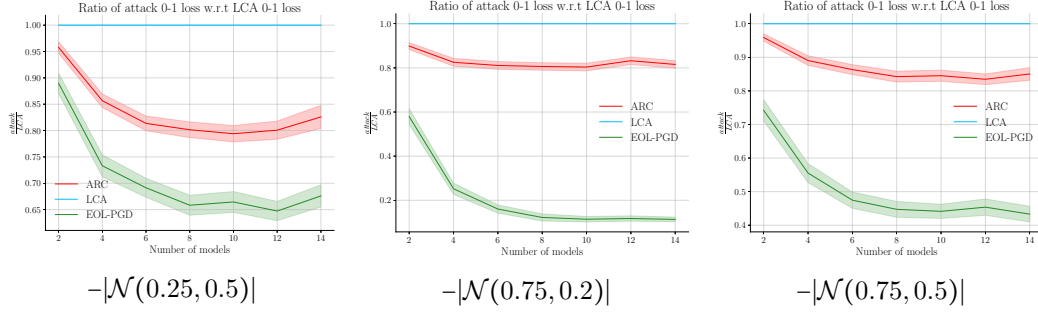


Fig. C.1.4.: Ratios for different values of α and β . y -axis are adapted for visual purposes.

- ARC for linear classifiers has no extra parameters. That is why it is extremely fast compared to EOL-PGD and LCA.
- numpy seed was set to 42
- Weights $\mu^{(i)}$ of each random mixture were also sampled randomly as follows:
 1. Sample m i.i.d standard normal values z_i .
 2. Compute the softmax with temperature $t = 10$, i.e $\mu^{(i)} = \frac{\exp(z_i/10)}{\sum_j \exp(z_j/10)}$.

Parameters for LCA:

- Number of steps T for inner PDG: $\max(2 \cdot m^2, 200)$.
- Step size η for inner PDG: $\frac{\epsilon}{\sqrt{T}}$.

Parameters for EOL-PGD:

- Number of steps T for inner PDG: $\max(2 \cdot m^2, 200)$.
- Step size η for inner PDG: $\frac{\epsilon}{\sqrt{T}}$.

C.2 CIFAR-10 experiments

C.2.1 Main results

For the experiments with the pre-trained models available in the DVERGE repository¹, we attacked the models with different versions of PGD, EOL-PGD, ARC and LCA, trying to cover a reasonable range of hyperparameters for all attacks such as the step size η and the number of iterations T . For the number of iterations, we tested 50, 100 and 500 for EOL-PGD, and 50, 100 and 200 for ARC. LCA was only tested

¹<https://github.com/zjysteven/DVERGE/tree/main>

with 50, to avoid making an unfair comparison. We tested LCA with a constant step size, and also with a changing step size that is divided by 2 at the iteration 40. The best results were used for the results in Table 3.1.

Here is a summary of the parameters used for the main results in Table 3.1:

General parameters:

- Norm: ℓ_∞ .
- Attack budget ϵ : 0.01 or 0.03.

Parameters for LCA:

- Number of steps T for inner PDG: 50.
- Step size η for inner PDG:
 - For $\epsilon = 0.01$: 0.0005, 0.001, 0.002
 - For $\epsilon = 0.03$: 0.0015, 0.003, 0.006

Parameters for EOL-PGD:

- Number of steps T : 50, 100 and 500.
- Step size η :
 - For $\epsilon = 0.01$: 0.0015, 0.002, 0.006
 - For $\epsilon = 0.03$: 0.0015, 0.006

Parameters for ARC:

- Number of steps T : 50, 100, 200.
- Step size η :
 - For $\epsilon = 0.01$: 0.0005, 0.001, 0.002, 0.006
 - For $\epsilon = 0.03$: 0.0015, 0.003, 0.006

C.2.2 Comparing strong versions of the attacks

We wanted to make sure that the gap between EOL-PGD, ARC and LCA was not influenced by the simple setting we considered for the main experiments in Table 3.1. We also wanted to test stronger versions of LCA to evaluate the possibility of further improvement.

Firstly, we launched stronger versions of both ARC and LCA against the model DV+AT/5 in the $\epsilon = 0.03$ setting. We added random initialization and 5 restarts to

both attacks, and tested with $T \in \{50, 100, 250\}$. For LCA, we also tested dividing the step size of the inner PGD by 2 at iterations $\frac{T}{2}$ and $\frac{3T}{4}$.

From all the different evaluations, we report the lowest robust accuracy (strongest attack) and compare it to the results obtained in Table 3.1.

Tab. C.2.1.: Mixture robustness evaluation of DV+AT/5 against stronger versions of ARC and LCA.

Attack	From Table 3.1	Stronger
ARC	44.6%	43.6%
LCA	39.9%	36.5%

Including random initialization and restarts only changed the results for ARC by 1% with respect to the results shown in Table 3.1. On the other hand, LCA improved by 3.4%, showing that the robustness evaluation of LCA could further improve with more hyperparameter tuning.

Secondly, we performed a more particular experiment varying only the number of iterations, similar to a convergence check when verifying that a proposed defense is not overestimating the reported robustness due to the non-convergence of the attack [Tra+20]. Again, we used the model DV+AT/5 in the $\epsilon = 0.03$ setting. We set the step size to 0.006 for all attacks, no random initialization, and LCA was tested with and without the reducing step size at iterations $\frac{T}{2}$ and $\frac{3T}{4}$ by half.

It can be seen that further increasing the number of iterations up to 1000 does not reduce the robustness evaluation of EOL-PGD or ARC by more than 0.2%. On the other hand, for this fixed setting, LCA with the reducing step size obtains an improvement of 0.8%, suggesting again that the reasonable version of LCA that we used for the main results, that already beats EOL-PGD and ARC, can still be enhanced.

Tab. C.2.2.: Convergence check. Robustness evaluation with an increasing number of iterations. LCA[†] corresponds to the version of LCA with the reducing step size.

Attack	T		
	100	500	1000
EOL	42.0%	41.9%	41.9%
ARC	44.1%	44.0%	43.9%
LCA	38.5%	38.0%	38.0%
LCA [†]	38.5%	37.9%	37.7%

Also note that other ways of exploring the lattice could be tested. Using multiple climbs, as is done by LCA with restarts, is akin to a naive version of ALL_MSS in [GMS97]. One possibility would be to implement the real ALL_MSS, with the *min-ortho-elements*. Another way of improving from LCA would be to consider a more exhaustive enumeration of the elements of the semi-lattice, with pruning techniques like the ones used in Apriori [AS+94] or MaxMiner [BJ98]. These improvements will however come at a great price in terms of execution time, leading to attacks that should be better for robustness evaluation, but impractical for other applications like Adversarial Training.

C.3 On adversarial training for mixtures of classifiers

A natural second step, after proposing an attack, is to see if it is beneficial for adversarial training. Our standard training for mixtures optimizes the sum of losses, which makes the training independent even if models are trained at the same time (not sequentially like in Dbouk and Shanbhag [DS23]). To add dependence, we test a vanilla adversarial training with a mixture attack: the attack is crafted for the mixture, which breaks the independence in the training process.

We tested adversarial training with EOL-PGD, ARC, LCA and vanilla PGD on the ensemble that averages logits, in CIFAR-10, with different architectures for our sub-models. We tested with very small convolutional networks (CNN), with the ResNet20 models used in the experiments in Table 3.1, and also with a bigger ResNet110. All these experiments were made under a different threat model (ℓ_2 with $\epsilon = 0.5$), so they are not comparable with the results from Table 3.1.

We can see that, as is already noted in the seminal paper [Mad+18], models with very low capacity are less able to learn when faced with more complex, stronger attacks. In the case of the CNN, training with EOL-PGD produces models with an individual robustness similar to what independent adversarial training would produce. On the other hand, training with LCA produces models with drastically less robustness. This was noted for all the tested versions of LCA (from 5 to 30 iterations, and various step sizes) and even with different mixtures sizes.

When increasing the capacity, the performance of AT with EOL-PGD and LCA becomes almost on par. It is intriguing that AT with ARC and LCA yields models with higher standard accuracy. We expected the opposite, considering that LCA and ARC are supposed to be stronger attacks against mixtures. This behavior led authors in

Tab. C.3.1.: Clean and robust accuracy of mixtures with different base models. Mixtures have been trained with adversarial training using different attacks during training under the ℓ_2 threat model with $\epsilon = 0.5$ as in standard practice. Higher is better (more robust). Robust accuracy reported is the minimum expected accuracy under any of the tested attacks (EOL-PGD, ARC, LCA, AutoAttack and PGD on the average logit ensemble).

Model	Attack for AT	Natural	Robust
CNN	EOL-PGD	59.9%	34.1%
	LCA	64.0%	27.2%
	PGD	61.8%	32.8%
ResNet20	EOL-PGD	81.2%	50.0%
	LCA	83.1%	49.6%
	ARC	84.5%	46.5%
	PGD	81.5%	43.4%
ResNet110	EOL-PGD	77.8%	47.2%
	LCA	80.7%	46.3%
	PGD	78.2%	40.6%

[DS23] to use EOL-PGD for their training method, BARRE, instead of using the stronger ARC-R that they propose.

The problem of training a mixture that increases the robustness with respect to the best individual sub-model remains open and is an interesting track for future work.

Résumé en français

Contents

D.1	Contexte et motivations	143
D.2	Énoncé du problème	145
D.2.1	Attaquer des mélanges de classifieurs	145
D.2.2	Intérêt de la randomisation en classification binaire . . .	146
D.2.3	Randomisation en classification multi-classe	146
D.3	Apprentissage automatique sous attaques adverses : Contexte et travaux connexes	146
D.3.1	Algorithmes d'attaque	151
D.3.2	Mécanismes de défense	155
D.4	Contributions de la thèse	160
D.4.1	Chapitre 3: Attaque des mélanges finis de classifieurs . .	160
D.4.2	Chapitre 4: Étude théorique du rôle de la randomisation (cas binaire)	160
D.4.3	Chapitre 5: Extension au cas multiclasse	161

Cette annexe présente un résumé détaillé de la thèse en français, en mettant l'accent sur les motivations, les problématiques abordées, ainsi que la structure de la thèse et les grandes lignes des mes contributions.

D.1 Contexte et motivations

Le domaine de l'intelligence artificielle (IA) et du machine learning (ML) a connu des avancées majeures ces dernières décennies, notamment avec l'essor du deep learning [LBH15]. Ces techniques trouvent des applications dans des domaines critiques comme la médecine [Lev+19; Pic+21], l'agriculture [KPB18], la pharmacie [Che+18] et les transports [Gri+20]. Cette progression est alimentée par l'explosion des données disponibles et la puissance de calcul croissante [YSD23].

Cependant, ces progrès soulèvent des défis, notamment la robustesse des modèles face aux perturbations adverses [LJ23]. Les attaques par empoisonnement altèrent les données d'entraînement pour dégrader les performances [BNL12], tandis que les

attaques par inversion de modèle [FJR15] ou inférence d'appartenance [Sho+17] visent à extraire des informations sensibles. Enfin, les *attaques par évasion* introduisent des perturbations imperceptibles lors de l'inférence pour induire des erreurs [Big+13; Sze+14]. Ces dernières ont suscité un intérêt majeur, avec près de 12 000 articles académiques sur le sujet début 2025¹.

Attaques adverses. Une attaque adverse est une perturbation imperceptible induisant une erreur du modèle sur une entrée autrement bien classée [Big+13; Sze+14]. Depuis leur réintroduction par Szegedy et al. [Sze+14], ces attaques révèlent un paradoxe : les réseaux de neurones excellents en reconnaissance vocale ou visuelle sont vulnérables à des modifications infimes. Elles touchent divers domaines : classification d'images [Sze+14], reconnaissance vocale² [CW18], et le traitement automatique du langage [Ebr+18].



Fig. D.1.1.: Exemples adverses pour le modèle AlexNet [KSH12] sur ImageNet [Den+09]. Les colonnes de gauche montrent les images originales, celles de droite les exemples adverses de Szegedy et al. [Sze+14], tous classés comme “autruche”. La colonne centrale affiche les différences. Source : [Sze+14, Section 4.2].

La dernière décennie a vu une course entre attaques sophistiquées [CW17a; CH20] et défenses empiriques [Mad+18]. Pour mieux comprendre le problème, une approche théorique modélise cette interaction comme un jeu à deux joueurs : un apprenant et un adversaire modifiant les entrées.

Perspective théorie des jeux. L'apprentissage robuste peut être formulé comme un jeu entre l'apprenant (classifieur) et l'adversaire (perturbations) [PJ21]. Cette approche a permis des résultats théoriques sur l'existence de classifieurs optimaux [TJK24], le comportement de l'attaquant [TJK23], et inspiré des défenses pratiques [Pin+20; Meu+21]. Un insight clé est l'absence fréquente d'équilibre de Nash pur, résolu par des *stratégies mixtes* [Osborne+04].

¹Voir le blog de Nicholas Carlini pour un décompte détaillé : <https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html>

²https://nicholas.carlini.com/code/audio_adversarial_examples/

Transposé aux classifieurs, ce concept mène aux *classifieurs randomisés* : un mélange sélectionne aléatoirement un classifieur déterministe (parmi une famille fixe) pour chaque prédiction [Pin+19; Meu+21].

Modèles randomisés en pratique. L'utilisation de la randomisation pour améliorer la robustesse adverse n'est pas nouvelle. Les classifieurs randomisés ont été employés dans diverses techniques défensives, notamment par transformation aléatoire des entrées [Xie+18; HRF19]. D'autres approches reposent sur la sous-sélection de modèles, via l'élagage stochastique de neurones [Dhi+18] (similaire au *dropout* [Sri+14]) ou des fonctions d'activation complexes favorisant la compétition neuronale [Pan+21; PCT21].

Bien que la flexibilité des classifieurs randomisés suggère une supériorité théorique face aux attaques adverses, leur évaluation reste problématique³. En pratique, ces modèles échouent souvent contre des attaques adaptatives intégrant leur randomisation [ACW18; Tra+20], révélant une résistance superficielle plutôt qu'une robustesse intrinsèque. Une clarification des bénéfices réels de la randomisation s'avère nécessaire.

D.2 Énoncé du problème

Cette thèse analyse théoriquement les avantages des classifieurs randomisés face aux attaques adverses. Nous étudions dans quelles conditions ils surpassent les classifieurs déterministes, en trois étapes :

- L'attaque des mélanges finis de classifieurs
- Les bénéfices en classification binaire
- L'extension aux problèmes multi-classes

D.2.1 Attaquer des mélanges de classifieurs

Face aux difficultés d'évaluation des classifieurs randomisés [DS22], nous posons :

Question 1: *Comment attaquer efficacement des mélanges de classifieurs ?*

³Robustbench [Cro+21] exclut explicitement les modèles randomisés, considérant qu'ils complexifient les attaques sans améliorer fondamentalement la robustesse.

Le Chapitre 3 propose une analyse géométrique du problème, reformulé comme l'énumération des éléments maximaux d'un treillis [DP02]. Cela éclaire la complexité théorique des attaques et positionne les méthodes existantes par rapport à l'attaque optimale. Nous y développons également un algorithme surpassant l'état de l'art.

D.2.2 Intérêt de la randomisation en classification binaire

Des exemples montrent que les classifieurs randomisés peuvent surpasser leurs homologues déterministes. Nous étudions :

Question 2: *Quand la randomisation améliore-t-elle la robustesse en classification binaire ?*

Le Chapitre 4 démontre que tout schéma de randomisation peut être surpassé par un classifieur déterministe, limitant son intérêt dans ce cadre. Nous construisons explicitement de tels classifieurs pour des techniques de randomisations courantes.

D.2.3 Randomisation en classification multi-classe

Nous étendons l'analyse au cas multi-classes :

Question 3: *Quand la randomisation améliore-t-elle la robustesse en classification multi-classe ?*

Contrairement au cas binaire, le Chapitre 5 révèle que pour certaines distributions, un classifieur randomisé peut largement surpasser le meilleur classifieur déterministe. Cela établit la nécessité théorique de la randomisation en multi-classe. Notre caractérisation partielle des distributions concernées mobilisera des outils de théorie des graphes et d'optimisation combinatoire.

D.3 Apprentissage automatique sous attaques adverses : Contexte et travaux connexes

L'apprentissage automatique contradictoire ou adverse étudie les méthodes d'apprentissage automatique (comme la classification) en présence d'un adversaire malveillant pouvant manipuler les données ou modèles lors de l'entraînement et/ou l'inférence. Par exemple, les attaques par empoisonnement altèrent subtilement les données

d'entraînement pour créer des portes dérobées exploitables ultérieurement [BNL12], tandis que les *attaques par évasion* modifient imperceptiblement les entrées lors de l'inférence pour fausser les prédictions [Big+13].

Bien qu'il existe divers types d'attaques, cette thèse se concentre sur les attaques par évasion et le développement de défenses randomisées. Cette section présente d'abord les bases de la classification, puis définit formellement les attaques adverses et leurs algorithmes de génération. Ensuite, nous examinons les mécanismes de défense existants avant de nous focaliser sur les stratégies randomisées, cœur de cette thèse.

Introduction à la classification Une tâche de classification comprend :

- Un espace d'entrées $\mathcal{X} \subseteq \mathbb{R}^n$ muni d'une norme ℓ_p (généralement ℓ_2 ou ℓ_∞), induisant une topologie et une σ -algèbre de Borel $\mathcal{B}(\mathcal{X})$.
- Un jeu de données $S = \{(x_1, y_1), \dots, (x_N, y_N)\}$ où $y_i \in \mathcal{Y} = \{1, \dots, K\}$.
- Une distribution de données inconnue ρ sur $\mathcal{X} \times \mathcal{Y}$.

Le but est de trouver une fonction $h : \mathcal{X} \rightarrow \mathcal{Y}$. En pratique, on définit d'abord une *fonction de score* $f : \mathcal{X} \rightarrow \mathbb{R}^K$, puis le classifieur par :

$$h(x) = \operatorname{argmax}_{k \in \mathcal{Y}} f(x)^{(k)}. \quad (\text{D.1})$$

Ces classifieurs *déterministes* (notés $\mathcal{H}_{\text{det}}$) attribuent toujours la même étiquette à une entrée x donnée. La Fig. D.3.1 illustre ceci sur un exemple où les points des différentes classes sont séparables.

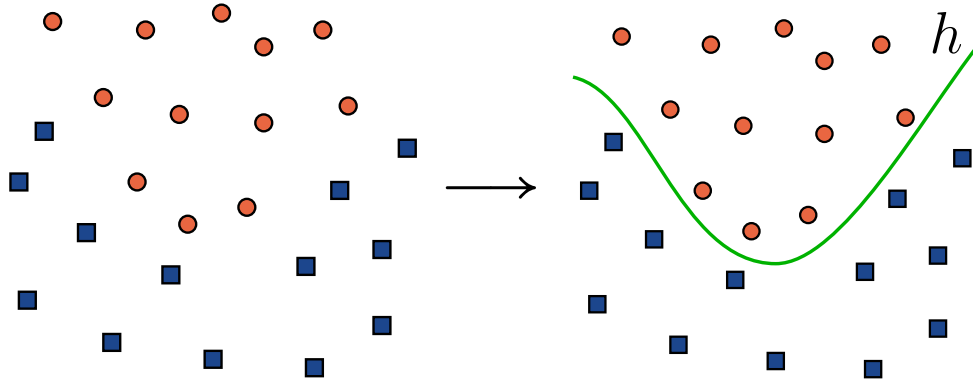


Fig. D.3.1.: Exemple de tâche de classification séparant des carrés bleus et cercles rouges. La ligne verte représente la frontière de décision du classifieur h .

Problème de minimisation du risque La qualité d'un classifieur se mesure par une fonction de perte $\mathcal{L} : \mathbb{R}^K \times \mathcal{Y} \rightarrow [0, \infty)$. En classification, on utilise typiquement la perte zero-one :

$$\mathcal{L}^{0-1}(f(x), y) := 1 \left\{ \operatorname{argmax}_{k \in \mathcal{Y}} f(x)^{(k)} \neq y \right\}. \quad (\text{D.2})$$

Le risque d'un classifieur est :

$$\mathcal{R}^{\mathcal{L}}(h, \rho) := \mathbb{E}_{(x,y) \sim \rho} [\mathcal{L}(h(x), y)]. \quad (\text{D.3})$$

L'apprenant cherche à minimiser ce risque sur une classe d'hypothèses $\mathcal{H} \subseteq \mathcal{H}_{\text{det}}$. En pratique, on minimise le *risque empirique* sur un jeu de données S :

$$\hat{\mathcal{R}}^{\mathcal{L}}(h, S) := \frac{1}{N} \sum_{i=1}^N \mathcal{L}(h(x_i), y_i). \quad (\text{D.4})$$

Le choix de $\mathcal{H}$ est crucial : trop riche, elle mène au *sur-apprentissage* (cf. classifieurs h_k dans (2.6) qui mémorisent S sans généralisation).

La perte zero-one, bien que théoriquement idéale, est difficile à optimiser. On utilise donc des *pertes substitués* différentiables comme l'entropie croisée :

$$\mathcal{L}^{\text{CE}}(f(x), y) = -\log \left(\frac{\exp(f(x)^{(y)})}{\sum_{j \in [K]} \exp(f(x)^{(j)})} \right). \quad (\text{D.5})$$

Une propriété clé est la *consistance* : une perte substitut est dite *consistante* si sa minimisation entraîne celle de la perte zero-one [BJM06].

Classifieurs randomisés Dans cette thèse, nous considérons également des *classifieurs randomisés*, qui associent à chaque entrée $x \in \mathcal{X}$ une distribution de probabilité catégorielle sur $\mathcal{Y}$. L'ensemble des distributions de probabilité sur $\mathcal{Y}$ peut être représenté par le simplexe standard K -dimensionnel :

$$\Delta^K = \{u \in \mathbb{R}^K \mid u \geq 0, \sum_{k=1}^K u^{(k)} = 1\},$$

où $u^{(k)}$ représente la k -ème composante du vecteur u . Un classifieur randomisé est donc une fonction mesurable $h : \mathcal{X} \rightarrow \Delta^K$, et nous notons $\mathcal{H}_{\text{rand}}$ l'ensemble de tous les classifieurs randomisés⁴.

⁴Les classifieurs déterministes sont un cas particulier où la probabilité 1 est attribuée à une seule étiquette. Formellement, à partir d'un classifieur déterministe h , on peut définir un classifieur

Les classifieurs randomisés produisent des prédictions aléatoires : pour une entrée x , une étiquette $\tilde{y}$ est tirée selon la distribution discrète de paramètre $h(x)$. Formellement :

$$\tilde{y} \sim \text{Categorical}(h(x)). \quad (\text{D.6})$$

Par abus de notation, nous écrivons simplement $\tilde{y} \sim h(x)$.

L'erreur d'un classifieur randomisé h en (x, y) se mesure par la *perte zero-one espérée*, où l'espérance est prise par rapport à l'aléa de $h(x)$:

$$\mathcal{L}^{0-1}(h(x), y) = \mathbb{E}_{\tilde{y} \sim h(x)} [\mathbb{1} \{\tilde{y} \neq y\}] = 1 - h(x)^{(y)} \quad (\text{D.7})$$

Nous utilisons le même symbole pour les versions déterministe et randomisée, cette dernière en étant une extension naturelle. Le risque d'un classifieur randomisé est alors défini comme dans (D.3).

Classification adverse Dans cette section, nous généralisons le problème de classification au cadre adversarial des *attaques par évacion*, où un adversaire peut manipuler les entrées lors de l'inférence pour tromper le classifieur. Nous introduirons le concept d'*exemples adverses*, présenterons les algorithmes développés pour les générer, et ferons un état de l'art des méthodes d'attaque et de défense, en accordant une attention particulière aux défenses randomisées (au cœur de cette thèse). Nous conclurons par une discussion sur l'importance d'utiliser des attaques puissantes pour évaluer la robustesse empirique.

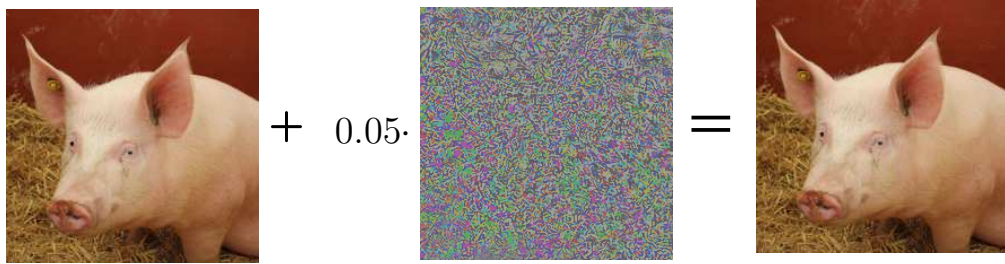


Fig. D.3.2.: Perturbation adverse d'une image de porc (ImageNet [Den+09]) pour un ResNet50 pré-entraîné de torchvision. L'image de gauche est correctement classée comme « porc » avec une haute confiance, tandis que celle de droite est erronément classée comme « avion de ligne ». La perturbation, imperceptible à l'œil humain, a été amplifiée d'un facteur 20 pour visualisation. Détails et code : <https://adversarial-ml-tutorial.org/introduction/>.

Qu'est-ce qu'un exemple adverse ? Un exemple adverse est une petite perturbation ajoutée à une entrée, induisant une erreur de classification avec une confiance élevée. La Fig. 2.2 illustre ce processus pour un réseau de neurones : l'image

randomisé équivalent h' qui ne produit que des vecteurs de la base canonique $\{e_1, \dots, e_K\} \subset \mathbb{R}^K$: $h'(x) = e_{h(x)}$.

originale (à gauche) est correctement classée comme « porc », tandis que la version perturbée (à droite) est classée comme « avion de ligne ». La perturbation, bien qu'imperceptible, suffit à tromper le modèle. Une telle perturbation peut être générée pour de nombreuses autres classes cibles.

Pour définir formellement les exemples adverses, nous commençons par introduire le voisinage adversarial d'une entrée x , c'est-à-dire l'ensemble des perturbations valides. Un choix courant est la boule ℓ_p fermée de rayon ϵ autour de x , notée $B_p(x, \epsilon)$, où ϵ représente le budget de l'adversaire. Ce paramètre est choisi suffisamment petit pour que les perturbations restent imperceptibles, les normes ℓ_∞ et ℓ_2 étant les plus utilisées. Dans ce travail, nous nous concentrons sur la boule ℓ_p fermée pour $p \in (1, \infty]$.

Pour une fonction de perte $\mathcal{L}$ donnée, un *attaquant optimal* prend un point (x, y) et trouve $x' \in B_p(x, \epsilon)$ qui maximise la perte du classifieur. La perte sous attaque s'écrit:

$$\mathcal{L}_{\epsilon,p}(h(x), y) := \sup_{x' \in B_p(x, \epsilon)} \mathcal{L}(h(x'), y) \quad (\text{D.8})$$

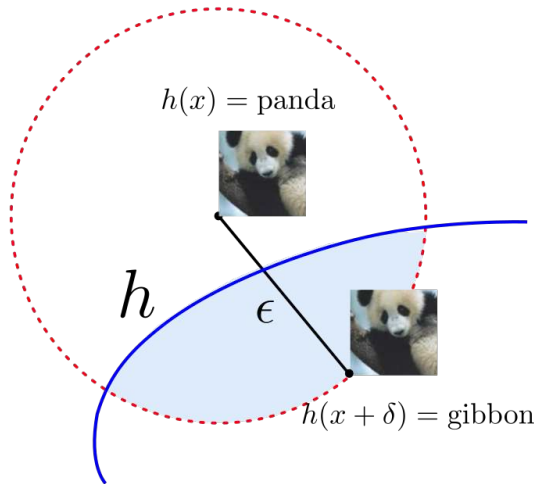


Fig. D.3.3.: Illustration d'une image originale x (ImageNet [Den+09]) et de son voisinage adversarial $B_2(x, \epsilon)$. Le panda perturbé est classifié comme un gibbon. Voir [GSS15].

L'étude des attaques adverses remonte à 2004, lorsque Dalvi et al. [Dal+04] introduisit la classification adverse pour le filtrage de spams. Globerson and Roweis [GR06] étendit ensuite ce cadre aux attaques par suppression de caractéristiques. Le terme *attaques par évasion* fut formalisé par Biggio et al. [Big+13], qui proposa une attaque par descente de gradient.

Les exemples adverses pour les réseaux de neurones furent popularisés par Szegedy et al. [Sze+14], révélant que ces modèles, bien que performants, étaient vulnérables

à des perturbations imperceptibles. Cette découverte intervint peu après le succès d’AlexNet [KSH12], qui surpassa l’état de l’art sur ImageNet [Den+09] avec une avance de 10% en 2012. Szegedy et al. [Sze+14] démontrèrent cette vulnérabilité sur AlexNet (Fig. D.1.1).

Minimisation du risque adversarial Dans cette thèse, nous considérons la classification adverse face à un adversaire optimal, tel que décrit dans (D.8). Pour une fonction de perte $\mathcal{L}$, une norme p et un budget $\epsilon > 0$, le *risque adversarial* d’un classifieur h par rapport à la distribution ρ s’écrit :

$$\mathcal{R}_{\epsilon,p}^{\mathcal{L}}(h, \rho) := \mathbb{E}_{(x,y) \sim \bar{\rho}} \left[\sup_{x' \in B_p(x, \epsilon)} \mathcal{L}(h(x'), y) \right]^5. \quad (\text{D.9})$$

Le problème de *minimisation du risque adversarial* sur une classe d’hypothèses $\mathcal{H}$ s’exprime alors :

$$\mathcal{R}_{\epsilon,p}^{\mathcal{L}}(\mathcal{H}, \rho) := \inf_{h \in \mathcal{H}} \mathcal{R}_{\epsilon,p}^{\mathcal{L}}(h, \rho) \quad (\text{D.10})$$

Pour la perte 0-1, nous noterons simplement $\mathcal{R}_{\epsilon,p}$. Le *risque adversarial empirique* sur un ensemble fini S se définit comme :

$$\hat{\mathcal{R}}_{\epsilon,p}^{\mathcal{L}}(h, S) := \frac{1}{N} \sum_{i=1}^N \sup_{x' \in B_p(x_i, \epsilon)} \mathcal{L}(h(x'), y_i) \quad (\text{D.11})$$

Figure D.3.4 illustre ce cadre : les boules autour des points représentent le voisinage adversarial. Un classifieur initialement performant devient vulnérable près de la frontière de décision, où des perturbations minimales induisent des erreurs. Lorsque les boules adverses de classes différentes se chevauchent, une séparation robuste devient impossible.

D.3.1 Algorithmes d’attaque

Le processus de conception d’attaques implique diverses techniques d’optimisation. Il existe deux approches principales : résoudre approximativement (D.8) en utilisant une fonction de perte substitut et une descente de gradient projetée, ou utiliser une relaxation lagrangienne du problème pour trouver une perturbation de norme minimale qui force une mauvaise classification. Dans ce qui suit, nous nous concentrons sur les algorithmes d’attaque dans le cadre du modèle de menace en boîte blanche.

⁵L’espérance est prise par rapport à $\bar{\rho}$, la complétion de la mesure de Borel ρ restreinte à la σ -algèbre universelle $\mathcal{U}(\mathcal{X} \times \mathcal{Y})$. Ce détail technique est important car la fonction de perte sous attaque peut ne pas être mesurable (cf. [AFM21; PJ23]), notamment pour la perte 0-1. Voir Appendix A.2.2 pour les preuves nécessaires à Theorem 4.1.3 dans Chapter 4.

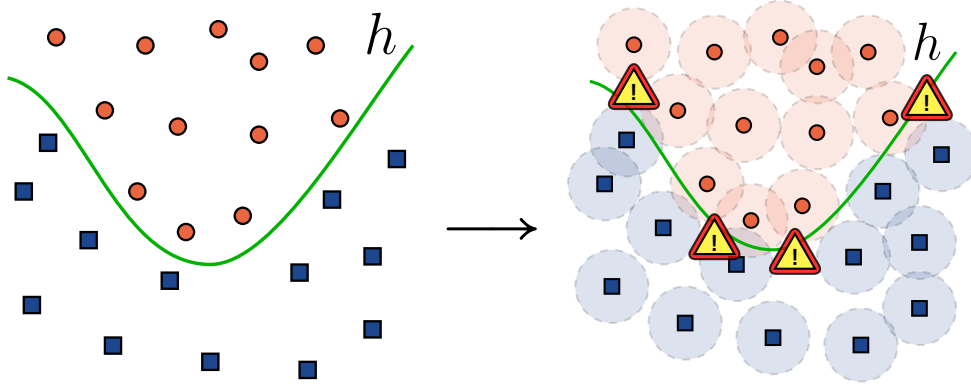


Fig. D.3.4.: Classification robuste face aux attaques : séparer les carrés bleus des cercles rouges malgré les perturbations dans les boules adverses.

Approche initiale de Szegedy et al. [Sze+14]. Pour un classifieur h basé sur la fonction de logit f , un point d'entrée $(x, y) \in \mathbb{R}^d \times [K]$ et une étiquette cible $y_{\text{adv}} \in [K]$, Szegedy et al. [Sze+14] ont tenté de résoudre le problème d'attaque ciblée ℓ_2 suivant dans leur travail original :

$$\begin{aligned} \min \quad & \|\delta\|_2 \\ \text{s.t.} \quad & h(x + \delta) = y_{\text{adv}}, \\ & x + \delta \in \mathcal{X}. \end{aligned} \tag{D.12}$$

Cependant, étant donné la difficulté à résoudre (D.12), les auteurs l'approximent en utilisant L-BFGS avec contraintes de boîte. Plus précisément, les auteurs considèrent une fonction de perte continue $\mathcal{L}$ (par exemple l'entropie croisée) et proposent le problème d'attaque ℓ_1 suivant :

$$\begin{aligned} \min \quad & c\|\delta\|_1 + \mathcal{L}(f(x + \delta), y_{\text{adv}}) \\ \text{s.t.} \quad & x + \delta \in \mathcal{X}. \end{aligned} \tag{D.13}$$

Ils effectuent une recherche linéaire pour trouver le c minimum pour lequel la solution de (D.13) satisfait $h(x + \delta) = y_{\text{adv}}$. Une grande valeur de c force la perturbation à avoir une petite norme, auquel cas il peut être difficile de changer la prédiction du classifieur. Une petite valeur de c permet une perturbation plus grande, ce qui favorise le changement de prédiction au risque de rendre la perturbation perceptible. En pratique, les perturbations trouvées par les auteurs étaient toujours imperceptibles à l'œil humain.

FGM. La *méthode du gradient rapide* (FGM) est une attaque en boîte blanche simple et efficace basée sur l'hypothèse que les réseaux de neurones sont localement linéaires autour de leurs entrées. Si nous considérons une fonction de perte substitut

$\mathcal{L}$ et son approximation du premier ordre autour d'une entrée x , nous obtenons que pour une petite perturbation δ ,

$$\mathcal{L}(f(x + \delta), y) \approx \mathcal{L}(f(x), y) + \nabla_x \mathcal{L}(f(x), y)^T \delta. \quad (\text{D.14})$$

Nous pouvons chercher la perturbation qui maximise cette approximation du premier ordre en la contraignant à avoir une norme d'au plus ϵ en considérant le problème d'optimisation suivant :

$$\begin{aligned} \max \quad & \delta^T \nabla_x \mathcal{L}(f(x), y) \\ \text{s.t.} \quad & \|\delta\|_p \leq \epsilon. \end{aligned} \quad (\text{D.15})$$

Dans le cas de la norme ℓ_2 , la solution de (D.15) est donnée par le maximiseur bien connu du produit scalaire :

$$\delta_{\text{FGM}} = \epsilon \cdot \frac{\nabla_x \mathcal{L}(f(x), y)}{\|\nabla_x \mathcal{L}(f(x), y)\|_2}. \quad (\text{D.16})$$

Dans le cas de la norme ℓ_∞ , qui était le problème considéré par Goodfellow et al. [GSS15], il existe une solution explicite : la célèbre attaque *méthode du signe du gradient rapide* (FGSM) :

$$\delta_{\text{FGSM}} = \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(f(x), y)). \quad (\text{D.17})$$

Notons que si le classifieur est linéaire, alors la propriété suivante est vérifiée dans la norme ℓ_∞ : si le classifieur est vulnérable, alors le signe du gradient est garanti d'être un exemple adversarial. Ce n'est pas le cas avec le gradient lui-même. La Fig. D.3.5 présente une visualisation du lien entre le gradient et le signe du gradient dans la norme ℓ_∞ pour un classifieur linéaire f .

PGD. L'attaque par *descente de gradient projetée* (PGD) est une extension de la FGM qui considère plusieurs étapes d'ascension de gradient. L'attaque PGD est définie récursivement comme suit :

$$x_{t+1} = \Pi_{B_p(x, \epsilon)}(x_t + \alpha \cdot \nabla_x \mathcal{L}(f(x_t), y)), \quad (\text{D.18})$$

où $\Pi_{B_p(x, \epsilon)}$ désigne la projection dans l'ensemble des perturbations adverses admissibles $B_p(x, \epsilon)$, et α est un paramètre de l'attaque qui contrôle la taille du pas à chaque itération. Notons que l'hypothèse de linéarité utilisée pour dériver l'attaque FGSM est moins critique pour l'attaque PGD. Il est important de noter que le nom PGD a été employé par Madry et al. [Mad+18], tandis qu'à peu près à la même époque Kurakin et al. [KGB17] proposaient une attaque presque identique qu'ils ont appelée *méthode itérative basique* (BIM). Notons que l'attaque PGD peut également être utilisée dans la norme ℓ_∞ , auquel cas il peut être judicieux de remplacer le

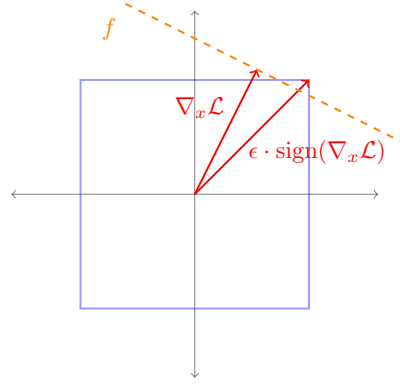


Fig. D.3.5.: Illustration de l'attaque FGSM. La perturbation est calculée comme le signe du gradient de la fonction de perte par rapport à l'entrée, mis à l'échelle par le budget adversarial ϵ . Notons que suivre le gradient de la perte pour créer un exemple adversarial produira une perturbation en dehors de la boule ℓ_∞ de rayon ϵ autour du point original x , tandis que suivre le signe du gradient produira une perturbation valide à l'intérieur de la boule ℓ_∞ .

gradient dans (D.18) par le signe du gradient, pour bénéficier de la solution explicite décrite dans (D.17).

Carlini-Wagner. L'attaque *Carlini-Wagner* (CW) [CW17b] revisite la formulation initiale à norme minimale de Szegedy et al. [Sze+14] et se concentre sur le choix de la fonction de perte tout en généralisant à différentes normes ℓ_p . Considérons un classifieur h basé sur la fonction de score f . Carlini and Wagner [CW17b] considèrent le problème d'optimisation suivant :

$$\begin{aligned} \min \quad & \|\delta\|_p + c \cdot g(x + \delta) \\ \text{s.t.} \quad & x + \delta \in [0, 1]^d. \end{aligned} \quad (\text{D.19})$$

où g est une fonction telle que $g(x + \delta) \leq 0$ si et seulement si $h(x + \delta) = y_{\text{adv}}$ pour une étiquette cible y_{adv} , par exemple

$$g(z) = \max \left(\max_{k \neq y_{\text{adv}}} f(z)^{(k)} - f(z)^{(y_{\text{adv}})}, 0 \right). \quad (\text{D.20})$$

Après avoir testé différentes améliorations, les auteurs proposent le problème d'optimisation non contraint suivant pour la norme ℓ_2 :

$$\min_w \quad \left\| \frac{1}{2}(\tanh(w) + 1) - x \right\|_2^2 + c \cdot g \left(\frac{1}{2}(\tanh(w) + 1) \right), \quad (\text{D.21})$$

où le changement de variable $\delta^{(i)} = \frac{1}{2}\tanh(w^{(i)} + 1) + x^{(i)}$ est appliqué, et la fonction g est celle définie dans (2.23). Pour la norme ℓ_∞ , ils proposent le problème d'optimisation suivant :

$$\min_{\delta} \quad c \cdot g(x + \delta) + \sum_i \max(\delta^{(i)} - \tau, 0), \quad (\text{D.22})$$

où τ est un paramètre qui contrôle quelles composantes de δ sont pénalisées. Il commence à 1 et est diminué d'un facteur 0,9 à chaque itération si $\delta^{(i)} < \tau$ pour tout i . L'attaque CW obtenue en résolvant soit (2.24) soit (2.25) selon la norme utilisée, est considérée comme l'une des attaques en boîte blanche les plus puissantes, et elle est largement utilisée pour évaluer la robustesse des classifieurs.

D.3.2 Mécanismes de défense

Un large corpus de recherche a été consacré au développement de mécanismes de défense contre les attaques adverses.

La méthode de défense la plus répandue est l'*apprentissage adversarial*, introduit par Goodfellow et al. [GSS15] puis formalisé par Madry et al. [Mad+18]. Cette approche consiste à enrichir l'entraînement du modèle avec des exemples adverses, améliorant ainsi sa robustesse aux perturbations. En pratique, on approxime souvent le risque adversarial empirique (Éq. (2.13)) à l'aide d'attaques comme PGD ou FGSM.

Bien qu'efficace, cette méthode est coûteuse en calculs. Bartoldson et al. [Bar+24] observent ainsi un surcoût computationnel d'un facteur 9 avec l'utilisation de PGD en 10 étapes. Diverses solutions ont été proposées pour accélérer le processus, comme le recyclage des gradients par Shafahi et al. [Sha+19] ou la restriction des calculs aux couches initiales du réseau [Zha+19].

Son efficacité combinée à d'autres techniques défensives fait débat. Tramer et al. [Tra+20] et Athalye et al. [ACW18] ont montré que certaines combinaisons pouvaient même réduire la robustesse.

Une avancée récente exploite les modèles de diffusion pour générer des données synthétiques [Gow+21; Reb+21]. Cette approche, qui produit parmi les modèles les plus robustes selon Robustbench [Cro+21], a été encore améliorée par Wang et al. [Wan+23] en utilisant des modèles de diffusion de meilleure qualité (mesurée par le FID).

Pour conclure, mentionnons que d'autres méthodes couramment étudiées dans la littérature - telles que les approches de robustesse certifiée (*randomized smoothing*, réseaux lipschitziens), les ensembles diversifiés ou les réseaux neuronaux bayésiens - seront abordées dans le corps du texte. Dans cette synthèse, nous nous concentrerons principalement sur le type de mécanisme de défense au cœur de cette thèse : les défenses randomisées.

Transformations aléatoires de l'entrée Les premières utilisations de la randomisation pour améliorer la robustesse aux attaques adverses remontent à la compétition *Adversarial Attacks and Defences* de NeurIPS 2017. Zhang and Liang [ZL19] ont remporté la première place (sur 107 soumissions) en proposant une technique de « discrétisation randomisée » consistant à injecter du bruit gaussien avant de quantifier l'entrée. La deuxième place a été obtenue par Xie et al. [Xie+18] avec une approche de redimensionnement et remplissage aléatoires des images, bien que cette défense ait ensuite été jugée inefficace par Athalye et al. [ACW18]. Cette idée de transformations aléatoires a été reprise par Raff et al. [Raf+19] qui ont utilisé 25 transformations différentes, atteignant ce qui semblait être une robustesse impressionnante sur ImageNet [Den+09], avant que Sitawarin et al. [SGSW22] ne démontre ses limites trois ans plus tard.

Parmi ces transformations, l'*injection de bruit* - ajout de bruit aléatoire à l'entrée avant prédiction - s'est révélée particulièrement prometteuse. Ces classifieurs randomisés combinent simplicité d'implémentation, efficacité computationnelle et fondements théoriques solides. Pinot et al. [Pin+22] ont proposé un cadre formel pour analyser leur robustesse via la notion de (ϵ, η, D) -robustesse : pour une norme p et une métrique probabiliste D (e.g. variation totale), un classifieur randomisé $h : \mathcal{X} \rightarrow \Delta^K$ est dit (ϵ, η, D) -robuste si pour tout $x, x' \in \mathcal{X}$:

$$\|x - x'\|_p \leq \epsilon \implies D(h(x), h(x')) \leq \eta. \quad (\text{D.23})$$

Leurs travaux établissent un lien crucial entre cette robustesse et l'écart entre risque standard et adversarial. Notant TV la distance en variation totale et $\mathcal{H}_{\epsilon, \eta}^{\text{TV}}$ l'ensemble des classifieurs $(\epsilon, \eta, \text{TV})$ -robustes, ils montrent que pour toute distribution ρ :

$$h \in \mathcal{H}_{\epsilon, \eta}^{\text{TV}} \implies \mathcal{R}_{\epsilon, p}(h, \rho) - \mathcal{R}(h, \rho) \leq \eta. \quad (\text{D.24})$$

Ce résultat, complété par une borne sur le gap de généralisation, offre une perspective optimiste quant à la minimisation du risque adversarial via le risque empirique. Il

nuance ainsi les conclusions de Tsipras et al. [Tsi+19] sur l'incompatibilité supposée entre précision standard et robuste, en fournissant des garanties théoriques pour une classe spécifique de classifieurs randomisés.

Sélection implicite de sous-modèles Une approche pour créer un classifieur randomisé consiste à fixer un classifieur de base puis à échantillonner un sous-modèle en élaguant certains neurones pendant l'inférence, suivant un principe similaire au dropout [Hin+12]. La première défense de ce type a été proposée par Dhillon et al. [Dhi+18], qui élaguent aléatoirement les activations à chaque couche selon leurs magnitudes, en utilisant une distribution multinomiale. Une illustration de cette procédure est présentée dans la Figure D.3.6. Cependant, cette défense s'est avérée inefficace, comme démontré par Athalye et al. [ACW18].

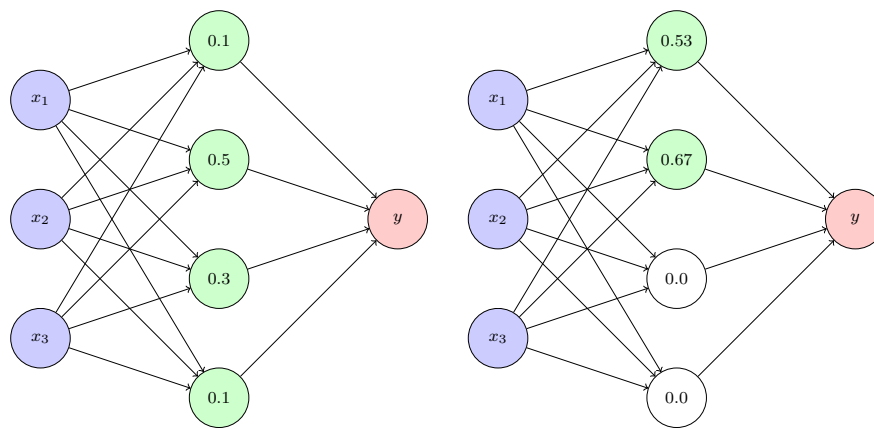


Fig. D.3.6.: Illustration de la procédure d'élagage proposée par Dhillon et al. [Dhi+18]. La figure de gauche montre le réseau neuronal original après le calcul des activations de la couche cachée (en vert), et la figure de droite montre le réseau élagué.

Dans une approche similaire, Panousis et al. [Pan+21] proposent des fonctions d'activation stochastiques de type k -winner-take-all, s'appuyant sur des travaux antérieurs ayant exploré ces activations compétitives dans un cadre déterministe [XZZ20]. Plutôt que d'élaguer les neurones indépendamment à chaque couche, Panousis et al. [Pan+21] regroupent les neurones d'une couche donnée en groupes compétitifs dont seulement k sont sélectionnés aléatoirement, avec une probabilité proportionnelle à leur activation. Ils développent également une méthode permettant d'élaguer les connexions entre neurones de couches consécutives, ouvrant la voie à des stratégies d'élagage plus agressives.

Cependant, ces techniques défensives ont été contournées : l'élagage proposé par Dhillon et al. [Dhi+18] a été invalidé par Athalye et al. [ACW18], et les fonctions k -winner-take-all de Panousis et al. [Pan+21] n'ont pas été retenues dans le benchmark Robustbench après qu'une adaptation mineure ait révélé une

robustesse bien inférieure aux allégations initiales (voir la discussion : <https://github.com/fra31/auto-attack/issues/58>).

Mélanges de classifieurs Inspirées par la théorie des stratégies mixtes dans les jeux à somme nulle, les mélanges de classifieurs offrent un cadre général pour décrire de nombreuses approches randomisées. L'idée centrale consiste à définir une distribution de probabilité sur un ensemble de classifieurs déterministes $\mathcal{H}$ (généralement de support fini), puis à échantillonner un modèle de $\mathcal{H}$ à chaque inférence pour effectuer une prédiction.

Formellement, on définit les mélanges de classifieurs comme suit :

Definition D.3.1 (Mélange de classifieurs [Pin+20; Meu+21]).

Étant donné une tâche de classification sur l'espace $\mathcal{X} \times \mathcal{Y}$, un ensemble de classifieurs déterministes $\mathcal{H}$ et une distribution de probabilité μ sur $\mathcal{H}$, le mélange h_μ est un classifieur randomisé défini par : pour tout point $x \in \mathcal{X}$ et étiquette $y \in \mathcal{Y}$,

$$h_\mu(x)^{(y)} = \mathbb{P}_{h \sim \mu} [h(x) = y]. \quad (\text{D.25})$$

Dans le cas où les classifieurs déterministes sont vus comme des fonctions $h : \mathcal{X} \rightarrow \Delta^K$, cette définition se simplifie en :

$$h_\mu(x) = \mathbb{E}_{h \sim \mu} [h(x)]. \quad (\text{D.26})$$

Cette formulation unifiée permet d'analyser théoriquement les propriétés de robustesse de diverses méthodes randomisées sous un même cadre conceptuel. Les équations (2.40) et (2.41) établissent le lien fondamental entre la randomisation des prédictions et l'espérance sur l'ensemble des modèles sous-jacents.

Discussion sur l'importance des algorithmes d'attaque puissants La définition de robustesse aux attaques adverses basée sur (2.11) exige qu'un classifieur résiste à toute perturbation respectant les contraintes spécifiées. Cependant, le calcul du supremum dans (2.10) étant souvent intraitable, les évaluations empiriques du risque adversarial ne fournissent qu'une borne inférieure du risque réel. Cette borne devient précise uniquement si l'algorithme d'attaque est suffisamment puissant pour trouver la pire perturbation possible en chaque point.

Cette limitation a conduit à des résultats ambigus sur la robustesse des mécanismes de défense, avec de nombreuses méthodes initialement jugées robustes mais fi-

nalement contournées par des attaques plus sophistiquées. Nous passons en revue plusieurs exemples notables et discutons particulièrement le cas des classifieurs randomisés (mélanges finis de classifieurs).

Des travaux fondateurs ([CW17a], [ACW18]) ont montré que de nombreuses défenses reposaient sur des artefacts (comme l’obfuscation des gradients) plutôt que sur une véritable robustesse. Des initiatives comme RobustBench [Cro+21] et des guidelines ([Car+19]) tentent d’améliorer les pratiques, mais des lacunes persistent, comme le souligne [Pop+22]. Le domaine reste en évolution, avec des attaques toujours plus sophistiquées invalidant régulièrement de nouvelles défenses, y compris pour les réseaux bayésiens ([Fen+24]).

Cas concrets de défenses randomisées invalidées Les techniques de randomisation ont été largement utilisées pour contourner les attaques existantes, conduisant à des surestimations importantes de robustesse. Cependant, la communauté prend désormais conscience de l’importance d’attaques puissantes pour évaluer ces défenses randomisées. Les travaux de [Gao+22] montrent que la plupart des défenses basées sur du pré-traitement stochastique sont moins robustes qu’estimé initialement. [Luc+23] suggère même que les défenses randomisées peuvent être moins sûres que leurs équivalents déterministes face à un adversaire capable d’attaquer itérativement.

Plusieurs défenses randomisées notables ont été remises en question :

- La défense par fonctions d’activation compétitives locales de [PCT21], invalidée via une simple adaptation d’AutoAttack ([CH20]) comme discuté sur GitHub
- Parmi les 7 défenses cassées par [ACW18] figuraient la stochastic pruning de [Dhi+18] et le random resizing de [Xie+18] (podium du concours NeurIPS 2017)
- L’extension avec transformations multiples de [Raf+19], récemment contournée par [SGSW22]

Attaques contre les mélanges finis de classifieurs Bien que constituant l’une des défenses randomisées les plus simples, les mélanges finis de classifieurs présentent des challenges d’évaluation non triviaux. [Pin+20] avait adapté des attaques existantes pour tenir compte de l’aléatoire, mais [DS22] a montré que leur attaque restait faible, surestimant la robustesse réelle. Ce cas illustre parfaitement la difficulté d’évaluer même les défenses randomisées les plus élémentaires.

Ce travail s'attaque précisément à ce problème dans le Chapter 3, en développant une attaque théoriquement fondée et empiriquement efficace contre les mélanges finis de classifieurs, avec pour objectif de fournir une évaluation plus précise de leur robustesse réelle.

D.4 Contributions de la thèse

D.4.1 Chapitre 3: Attaque des mélanges finis de classifieurs

Nous développons une nouvelle méthode d'attaque pour les mélanges finis de classifieurs en reformulant le problème comme une énumération des éléments maximaux d'un treillis. L'algorithme résultant offre:

- De meilleures garanties théoriques dans le cas linéaire binaire
- Des performances à l'état de l'art sur des jeux de données réels avec réseaux de neurones

Cette contribution fournit un outil puissant pour évaluer rigoureusement la robustesse de ces défenses randomisées.

D.4.2 Chapitre 4: Étude théorique du rôle de la randomisation (cas binaire)

Nous établissons des résultats fondamentaux sur la classification binaire randomisée:

- Toute classification randomisée peut être reproduite par une famille appropriée de classifieurs déterministes
- Nous analysons les implications théoriques de cette équivalence, montrant un lien entre la complexité de la classe d'hypothèse sous-jacente à une technique de randomisation et les bénéfices de la randomization en termes de robustesse aux attaques adverses.

Ces résultats éclairent les limites fondamentales des approches randomisées dans le cadre binaire.

D.4.3 Chapitre 5: Extension au cas multiclasse

Notre analyse théorique est étendue aux problèmes multiclassés, avec une étude particulière:

- Cas des distributions de données à support fini
- Utilisation d'outils de théorie des graphes
- Résultat marquant: l'écart entre classifieurs optimal déterministe et randomisé peut approcher 0.5 (contrairement au cas binaire où il est toujours nul)

Cette thèse apporte ainsi des contributions à la fois algorithmiques (Chapitre 3) et théoriques (Chapitres 4-5) pour mieux comprendre et évaluer les défenses randomisées contre les attaques adverses. Les résultats révèlent des différences fondamentales entre les cas binaires et multiclassés, ouvrant de nouvelles perspectives pour le développement de défenses robustes.

Bibliography

- [Rob69] Fred S Roberts. “On the boxicity and cubicity of a graph”. In: *Recent progress in combinatorics* 1.1 (1969), pp. 301–310 (cit. on p. [123](#)).
- [Sch79] A Schrijver. “Fractional packing and covering”. In: *Packing and covering in combinatorics*. Vol. 106. Mathematisch Centrum Amsterdam, 1979, pp. 201–274 (cit. on p. [77](#)).
- [Fis83] Peter C Fishburn. “On the sphericity and cubicity of graphs”. In: *Journal of Combinatorial Theory, Series B* 35.3 (1983), pp. 309–318 (cit. on p. [124](#)).
- [Ber84] Claude Berge. *Hypergraphs: combinatorics of finite sets*. Vol. 45. Elsevier, 1984 (cit. on pp. [75](#), [84](#)).
- [EH85] Paul Erdős and András Hajnal. “Chromatic number of finite and infinite graphs and hypergraphs”. In: *Discrete Mathematics* 53 (1985), pp. 281–285 (cit. on p. [126](#)).
- [Rud87] Walter Rudin. *Real and complex analysis, 3rd ed.* USA: McGraw-Hill, Inc., 1987 (cit. on pp. [64](#), [98](#), [99](#)).
- [CCD93] Fan RK Chung, Richard Cleve, and Paul Dagum. “A note on constructive lower bounds for the Ramsey numbers $R(3, t)$ ”. In: *Journal of Combinatorial Theory, Series B* 57.1 (1993), pp. 150–155 (cit. on pp. [88](#), [122](#), [127](#)).
- [AS+94] Rakesh Agrawal, Ramakrishnan Srikant, et al. “Fast algorithms for mining association rules”. In: *Proc. 20th int. conf. very large data bases, VLDB*. Vol. 1215. Santiago, Chile. 1994, pp. 487–499 (cit. on pp. [51](#), [141](#)).
- [AK95] Edoardo Amaldi and Viggo Kann. “The complexity and approximability of finding maximum feasible subsystems of linear relations”. In: *Theoretical computer science* 147.1-2 (1995), pp. 181–210 (cit. on p. [95](#)).
- [GMS97] Dimitrios Gunopulos, Heikki Mannila, and Sanjeev Saluja. “Discovering all most specific sentences by randomized algorithms extended abstract”. In: *Database Theory—ICDT’97: 6th International Conference Delphi, Greece, January 8–10, 1997 Proceedings* 6. Springer. 1997, pp. 215–229 (cit. on pp. [51](#), [141](#)).
- [BJ98] Roberto J Bayardo Jr. “Efficiently mining long patterns from databases”. In: *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*. 1998, pp. 85–93 (cit. on pp. [51](#), [141](#)).

- [FSA99] Yoav Freund, Robert Schapire, and Naoki Abe. “A short introduction to boosting”. In: *Journal-Japanese Society For Artificial Intelligence* 14.771-780 (1999), p. 1612 (cit. on p. 67).
- [DP02] Brian A Davey and Hilary A Priestley. *Introduction to lattices and order*. Cambridge university press, 2002 (cit. on pp. 6, 48, 146).
- [Chu+03] Maria Chudnovsky, Neil Robertson, Paul D Seymour, and Robin Thomas. “Progress on perfect graphs”. In: *Mathematical Programming* 97 (2003), pp. 405–422 (cit. on pp. 82, 84, 117).
- [CLV03] G. Cornuejols, Xinming Liu, and K. Vuskovic. “A polynomial algorithm for recognizing perfect graphs”. In: *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings*. 2003, pp. 20–27 (cit. on p. 85).
- [Dal+04] Nilesh Dalvi, Pedro Domingos, Mausam, Sumit Sanghai, and Deepak Verma. “Adversarial classification”. In: *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2004, pp. 99–108 (cit. on pp. 15, 150).
- [Os+04] Martin J Osborne et al. *An introduction to game theory*. Vol. 3. 3. Springer, 2004 (cit. on pp. 4, 144).
- [BJM06] Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. “Convexity, classification, and risk bounds”. In: *Journal of the American Statistical Association* 101.473 (2006), pp. 138–156 (cit. on pp. 13, 148).
- [GR06] Amir Globerson and Sam Roweis. “Nightmare at test time: robust learning by feature deletion”. In: *Proceedings of the 23rd international conference on Machine learning*. 2006, pp. 353–360 (cit. on pp. 16, 150).
- [Den+09] Jia Deng, Wei Dong, Richard Socher, et al. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255 (cit. on pp. 4, 14–16, 28, 144, 149–151, 156).
- [JT11] Tommy R Jensen and Bjarne Toft. *Graph coloring problems*. John Wiley & Sons, 2011 (cit. on p. 126).
- [BNL12] Battista Biggio, Blaine Nelson, and Pavel Laskov. “Poisoning Attacks against Support Vector Machines”. In: *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress, 2012 (cit. on pp. 3, 9, 143, 147).
- [Hin+12] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. “Improving neural networks by preventing co-adaptation of feature detectors”. In: *arXiv preprint arXiv:1207.0580* (2012) (cit. on pp. 29, 157).
- [KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems* 25 (2012) (cit. on pp. 4, 16, 144, 151).
- [XM12] Huan Xu and Shie Mannor. “Robustness and generalization”. In: *Machine learning* 86 (2012), pp. 391–423 (cit. on p. 22).

- [Big+13] Battista Biggio, Igino Corona, Davide Maiorca, et al. “Evasion attacks against machine learning at test time”. In: *Joint European conference on machine learning and knowledge discovery in databases*. Springer. 2013, pp. 387–402 (cit. on pp. 3, 9, 16, 144, 147, 150).
- [Con+14] Michele Conforti, Gérard Cornuéjols, Giacomo Zambelli, et al. *Integer programming models*. Springer, 2014 (cit. on p. 84).
- [Sri+14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. “Dropout: a simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958 (cit. on pp. 5, 145).
- [Sze+14] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, et al. “Intriguing properties of neural networks”. In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2014 (cit. on pp. 3, 4, 16, 18, 20, 24, 144, 150–152, 154).
- [Bub+15] Sébastien Bubeck et al. “Convex optimization: Algorithms and complexity”. In: *Foundations and Trends® in Machine Learning* 8.3-4 (2015), pp. 231–357 (cit. on p. 96).
- [FJR15] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. “Model inversion attacks that exploit confidence information and basic countermeasures”. In: *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. 2015, pp. 1322–1333 (cit. on pp. 3, 144).
- [GSS15] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. “Explaining and Harnessing Adversarial Examples”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015 (cit. on pp. 15, 19, 21, 150, 153, 155).
- [LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *nature* 521.7553 (2015), pp. 436–444 (cit. on pp. 3, 143).
- [He+16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778 (cit. on p. 56).
- [CW17a] 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, pp. 3–14 (cit. on pp. 4, 35, 144, 159).
- [CW17b] Nicholas Carlini and David A. Wagner. “Towards Evaluating the Robustness of Neural Networks”. In: *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*. IEEE Computer Society, 2017, pp. 39–57 (cit. on pp. 20, 35, 54, 154).
- [Cis+17] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. “Parseval networks: Improving robustness to adversarial examples”. In: *International conference on machine learning*. PMLR. 2017, pp. 854–863 (cit. on pp. 22, 23).

- [HA17] Matthias Hein and Maksym Andriushchenko. “Formal guarantees on the robustness of a classifier against adversarial manipulation”. In: *Advances in neural information processing systems* 30 (2017) (cit. on p. 22).
- [KGB17] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. “Adversarial Machine Learning at Scale”. In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017 (cit. on pp. 19, 153).
- [Sho+17] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. “Membership inference attacks against machine learning models”. In: *2017 IEEE symposium on security and privacy (SP)*. IEEE. 2017, pp. 3–18 (cit. on pp. 3, 144).
- [ACW18] 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*. PMLR. 2018, pp. 274–283 (cit. on pp. 5, 21, 28, 30, 35, 36, 145, 155–157, 159).
- [Ath+18] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. “Synthesizing robust adversarial examples”. In: *International conference on machine learning*. PMLR. 2018, pp. 284–293 (cit. on pp. 34, 40).
- [CW18] Nicholas Carlini and David Wagner. “Audio adversarial examples: Targeted attacks on speech-to-text”. In: *2018 IEEE security and privacy workshops (SPW)*. IEEE. 2018, pp. 1–7 (cit. on pp. 4, 144).
- [Che+18] Hongming Chen, Ola Engkvist, Yinhai Wang, Marcus Olivecrona, and Thomas Blaschke. “The rise of deep learning in drug discovery”. In: *Drug discovery today* 23.6 (2018), pp. 1241–1250 (cit. on pp. 3, 143).
- [Dhi+18] Guneet S. Dhillon, Kamyar Azizzadenesheli, Zachary C. Lipton, et al. “Stochastic Activation Pruning for Robust Adversarial Defense”. In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018 (cit. on pp. 5, 29, 30, 36, 145, 157, 159).
- [Ebr+18] 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 (Volume 2: Short Papers)*. Ed. by Iryna Gurevych and Yusuke Miyao. Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 31–36 (cit. on pp. 4, 144).
- [KPB18] Andreas Kamilaris and Francesc X Prenafeta-Boldú. “Deep learning in agriculture: A survey”. In: *Computers and electronics in agriculture* 147 (2018), pp. 70–90 (cit. on pp. 3, 143).
- [Mad+18] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. “Towards Deep Learning Models Resistant to Adversarial Attacks”. In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018 (cit. on pp. 4, 19, 21, 56, 58, 141, 144, 153, 155).

- [TSS18] Yusuke Tsuzuku, Issei Sato, and Masashi Sugiyama. “Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks”. In: *Advances in neural information processing systems* 31 (2018) (cit. on pp. 22, 24).
- [Xie+18] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan L. Yuille. “Mitigating Adversarial Effects Through Randomization”. In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018 (cit. on pp. 5, 28, 34, 36, 145, 156, 159).
- [BCM19] Arjun Nitin Bhagoji, Daniel Cullina, and Prateek Mittal. “Lower bounds on adversarial robustness from optimal transport”. In: *Advances in Neural Information Processing Systems* 32 (2019) (cit. on pp. 70, 86).
- [Car+19] Nicholas Carlini, Anish Athalye, Nicolas Papernot, et al. “On evaluating adversarial robustness”. In: *arXiv preprint arXiv:1902.06705* (2019) (cit. on pp. 36, 159).
- [CRK19] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. “Certified adversarial robustness via randomized smoothing”. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 1310–1320 (cit. on p. 24).
- [HRF19] Zhezhi He, Adnan Siraj Rakin, and Deliang Fan. “Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 588–597 (cit. on pp. 5, 68, 145).
- [KQ19] Sanjay Kariyappa and Moinuddin K. Qureshi. “Improving Adversarial Robustness of Ensembles with Diversity Training”. In: *CoRR abs/1901.09981* (2019). arXiv: 1901.09981 (cit. on pp. 25, 56, 58).
- [Lec+19] Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. “Certified robustness to adversarial examples with differential privacy”. In: *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2019, pp. 656–672 (cit. on p. 23).
- [Lev+19] Adrian B Levine, Colin Schlosser, Jasleen Grewal, et al. “Rise of the machines: advances in deep learning for cancer diagnosis”. In: *Trends in cancer* 5.3 (2019), pp. 157–169 (cit. on pp. 3, 143).
- [Pan+19] Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. “Improving adversarial robustness via promoting ensemble diversity”. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 4970–4979 (cit. on pp. 24, 36, 56, 58).
- [PS19] Juan C Perdomo and Yaron Singer. “Robust attacks against multiple classifiers”. In: *arXiv preprint arXiv:1906.02816* (2019) (cit. on pp. 52–54, 96).
- [Pin+19] Rafael Pinot, Laurent Meunier, Alexandre Araujo, et al. “Theoretical evidence for adversarial robustness through randomization”. In: *Advances in Neural Information Processing Systems* 32 (2019) (cit. on pp. 5, 145).
- [Raf+19] Edward Raff, Jared Sylvester, Steven Forsyth, and Mark McLean. “Barrage of random transforms for adversarially robust defense”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 6528–6537 (cit. on pp. 28, 29, 34, 36, 37, 39, 156, 159).

- [Sal+19] Hadi Salman, Jerry Li, Ilya Razenshteyn, et al. “Provably robust deep learning via adversarially trained smoothed classifiers”. In: *Advances in Neural Information Processing Systems* 32 (2019) (cit. on p. 24).
- [Sha+19] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, et al. “Adversarial training for free!” In: *Advances in neural information processing systems* 32 (2019) (cit. on pp. 21, 155).
- [Tsi+19] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. “Robustness May Be at Odds with Accuracy”. In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019 (cit. on pp. 28, 157).
- [Wan+19] Yisen Wang, Xingjun Ma, James Bailey, et al. “On the Convergence and Robustness of Adversarial Training”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, 2019, pp. 6586–6595 (cit. on p. 59).
- [Zha+19] Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. “You only propagate once: Accelerating adversarial training via maximal principle”. In: *Advances in neural information processing systems* 32 (2019) (cit. on pp. 21, 155).
- [ZL19] Yuchen Zhang and Percy Liang. “Defending against whitebox adversarial attacks via randomized discretization”. In: *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR. 2019, pp. 684–693 (cit. on pp. 28, 156).
- [Car+20] Ginevra Carbone, Matthew Wicker, Luca Laurenti, et al. “Robustness of bayesian neural networks to gradient-based attacks”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 15602–15613 (cit. on pp. 27, 36).
- [CH20] Francesco Croce and Matthias Hein. “Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks”. In: *International conference on machine learning*. PMLR. 2020, pp. 2206–2216 (cit. on pp. 4, 34, 36, 144, 159).
- [Gri+20] Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. “A survey of deep learning techniques for autonomous driving”. In: *Journal of field robotics* 37.3 (2020), pp. 362–386 (cit. on pp. 3, 143).
- [Pin+20] Rafael Pinot, Raphael Ettedgui, Geovani Rizk, Yann Chevalere, and Jamal Atif. “Randomization matters how to defend against strong adversarial attacks”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 7717–7727 (cit. on pp. 4, 31–33, 37, 40, 41, 44, 69, 144, 158, 159).
- [Tra+20] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. “On adaptive attacks to adversarial example defenses”. In: *Advances in neural information processing systems* 33 (2020), pp. 1633–1645 (cit. on pp. 5, 21, 25, 26, 36, 41, 140, 145, 155).
- [XZZ20] Chang Xiao, Peilin Zhong, and Changxi Zheng. “Enhancing Adversarial Defense by k-Winners-Take-All”. In: *International Conference on Learning Representations*. 2020 (cit. on pp. 21, 30, 157).

- [Yan+20] Huanrui Yang, Jingyang Zhang, Hongliang Dong, et al. “DVERGE: diversifying vulnerabilities for enhanced robust generation of ensembles”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 5505–5515 (cit. on pp. 26, 56–58, 61).
- [Ara+21] Alexandre Araujo, Benjamin Negrevergne, Yann Chevaleyre, and Jamal Atif. “On lipschitz regularization of convolutional layers using toeplitz matrix theory”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 8. 2021, pp. 6661–6669 (cit. on p. 23).
- [AFM21] Pranjal Awasthi, Natalie S Frank, and Mehryar Mohri. “On the existence of the adversarial bayes classifier (extended version)”. In: *arXiv preprint arXiv:2112.01694* (2021) (cit. on pp. 16, 66, 69, 70, 99, 151).
- [Cro+21] Francesco Croce, Maksym Andriushchenko, Vikash Sehwal, et al. “Robust-Bench: a standardized adversarial robustness benchmark”. In: *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*. Ed. by Joaquin Vanschoren and Sai-Kit Yeung. 2021 (cit. on pp. 5, 22, 36, 93, 145, 155, 159).
- [Gow+21] Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, et al. “Improving robustness using generated data”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 4218–4233 (cit. on pp. 22, 155).
- [Meu+21] Laurent Meunier, Meyer Scetbon, Rafael B Pinot, Jamal Atif, and Yann Chevaleyre. “Mixed nash equilibria in the adversarial examples game”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 7677–7687 (cit. on pp. 4, 5, 31, 33, 34, 40, 76, 144, 145, 158).
- [Pan+21] Konstantinos Panousis, Sotirios Chatzis, Antonios Alexos, and Sergios Theodoridis. “Local competition and stochasticity for adversarial robustness in deep learning”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2021, pp. 3862–3870 (cit. on pp. 5, 30, 145, 157).
- [PCT21] Konstantinos P Panousis, Sotirios Chatzis, and Sergios Theodoridis. “Stochastic local winner-takes-all networks enable profound adversarial robustness”. In: *arXiv preprint arXiv:2112.02671* (2021) (cit. on pp. 5, 36, 145, 159).
- [Pic+21] Francesco Piccialli, Vittorio Di Somma, Fabio Giampaolo, Salvatore Cuomo, and Giancarlo Fortino. “A survey on deep learning in medicine: Why, how and when?”. In: *Information Fusion* 66 (2021), pp. 111–137 (cit. on pp. 3, 143).
- [PJ21] Muni Sreenivas Pydi and Varun Jog. “The many faces of adversarial risk”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 10000–10012 (cit. on pp. 4, 69, 144).
- [Reb+21] Sylvestre-Alvise Rebuffi, Sven Gowal, Dan A Calian, et al. “Fixing data augmentation to improve adversarial robustness”. In: *arXiv preprint arXiv:2103.01946* (2021) (cit. on pp. 22, 155).
- [Yan+21] Zhuolin Yang, Linyi Li, Xiaojun Xu, et al. “Trs: Transferability reduced ensemble via promoting gradient diversity and model smoothness”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 17642–17655 (cit. on pp. 25, 26).

- [DI22] Ewan Davies and Freddie Illingworth. “The χ -Ramsey problem for triangle-free graphs”. In: *SIAM Journal on Discrete Mathematics* 36.2 (2022), pp. 1124–1134 (cit. on p. 126).
- [DS22] Hassan Dbouk and Naresh Shanbhag. “Adversarial Vulnerability of Randomized Ensembles”. In: *International Conference on Machine Learning*. PMLR. 2022, pp. 4890–4917 (cit. on pp. 6, 33, 34, 37, 40, 42–44, 46, 49, 52, 53, 57, 58, 61, 145, 159).
- [Gao+22] Yue Gao, Ilia Shumailov, Kassem Fawaz, and Nicolas Papernot. “On the limitations of stochastic pre-processing defenses”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 24280–24294 (cit. on pp. 29, 36, 159).
- [Meu+22] Laurent Meunier, Blaise J Delattre, Alexandre Araujo, and Alexandre Allauzen. “A dynamical system perspective for lipschitz neural networks”. In: *International Conference on Machine Learning*. PMLR. 2022, pp. 15484–15500 (cit. on p. 23).
- [Pin+22] Rafael Pinot, Laurent Meunier, Florian Yger, et al. “On the robustness of randomized classifiers to adversarial examples”. In: *Machine Learning* 111.9 (2022), pp. 3425–3457 (cit. on pp. 28, 29, 156).
- [Pop+22] Nikola Popovic, Danda Pani Paudel, Thomas Probst, and Luc Van Gool. “Gradient Obfuscation Checklist Test Gives a False Sense of Security”. In: *arXiv preprint arXiv:2206.01705* (2022) (cit. on pp. 36, 159).
- [PL22] Bernd Prach and Christoph H Lampert. “Almost-orthogonal layers for efficient general-purpose lipschitz networks”. In: *European Conference on Computer Vision*. Springer. 2022, pp. 350–365 (cit. on p. 23).
- [SGSW22] Chawin Sitawarin, Zachary J Golan-Strieb, and David Wagner. “Demystifying the adversarial robustness of random transformation defenses”. In: *International Conference on Machine Learning*. PMLR. 2022, pp. 20232–20252 (cit. on pp. 28, 29, 36, 37, 39, 156, 159).
- [Zha+22] Dinghuai Zhang, Hongyang Zhang, Aaron Courville, et al. “Building Robust Ensembles via Margin Boosting”. In: *Proceedings of the 39th International Conference on Machine Learning*. Vol. 162. Proceedings of Machine Learning Research. PMLR, June 2022, pp. 26669–26692 (cit. on pp. 56, 58).
- [Ara+23] Alexandre Araujo, Aaron J. Havens, Blaise Delattre, Alexandre Allauzen, and Bin Hu. “A Unified Algebraic Perspective on Lipschitz Neural Networks”. In: *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023 (cit. on p. 23).
- [Bor+23] Endre Boros, Vladimir Gurvich, Martin Milanič, and Yushi Uno. “Dually conformal hypergraphs”. In: *arXiv preprint arXiv:2309.00098* (2023) (cit. on p. 85).
- [DS23] Hassan Dbouk and Naresh R. Shanbhag. “On the Robustness of Randomized Ensembles to Adversarial Perturbations”. In: *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*. Ed. by Andreas Krause, Emma Brunskill, Kyunghyun Cho, et al. Vol. 202. Proceedings of Machine Learning Research. PMLR, 2023, pp. 7303–7328 (cit. on pp. 53, 57–59, 61, 141, 142).

- [Del+23] Blaise Delattre, Quentin Barthélemy, Alexandre Araujo, and Alexandre Allauzen. “Efficient bound of Lipschitz constant for convolutional layers by gram iteration”. In: *International Conference on Machine Learning*. PMLR. 2023, pp. 7513–7532 (cit. on p. 23).
- [LJ23] Jia Liu and Yaochu Jin. “A comprehensive survey of robust deep learning in computer vision”. In: *Journal of Automation and Intelligence* (2023) (cit. on pp. 3, 143).
- [Luc+23] Keane Lucas, Matthew Jagielski, Florian Tramèr, Lujo Bauer, and Nicholas Carlini. “Randomness in ML Defenses Helps Persistent Attackers and Hinders Evaluators”. In: *arXiv preprint arXiv:2302.13464* (2023) (cit. on pp. 36, 159).
- [PJ23] Muni Sreenivas Pydi and Varun Jog. “The many faces of adversarial risk: An expanded study”. In: *IEEE Transactions on Information Theory* (2023) (cit. on pp. 16, 70, 99, 151).
- [TJK23] Nicolás García Trillos, Matt Jacobs, and Jakwang Kim. “The multimarginal optimal transport formulation of adversarial multiclass classification”. In: *Journal of Machine Learning Research* 24.45 (2023), pp. 1–56 (cit. on pp. 4, 69, 74, 86, 87, 93, 144).
- [Wan+23] Zekai Wang, Tianyu Pang, Chao Du, et al. “Better diffusion models further improve adversarial training”. In: *International Conference on Machine Learning*. PMLR. 2023, pp. 36246–36263 (cit. on pp. 22, 155).
- [YSD23] İbrahim Yazıcı, İbraheem Shaye, and Jafri Din. “A survey of applications of artificial intelligence and machine learning in future mobile networks-enabled systems”. In: *Engineering Science and Technology, an International Journal* 44 (2023), p. 101455 (cit. on pp. 3, 143).
- [Bar+24] Brian R. Bartoldson, James Diffenderfer, Konstantinos Parasyris, and Bhavya Kailkhura. “Adversarial Robustness Limits via Scaling-Law and Human-Alignment Studies”. In: *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024 (cit. on pp. 21, 155).
- [Bor+24] Luca Bortolussi, Ginevra Carbone, Luca Laurenti, et al. “On the robustness of bayesian neural networks to adversarial attacks”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2024) (cit. on pp. 27, 36).
- [Dai+24] Sihui Dai, Wenxin Ding, Arjun Nitin Bhagoji, et al. “Characterizing the Optimal 0 – 1 Loss for Multi-class Classification with a Test-time Attacker”. In: *Advances in Neural Information Processing Systems* 36 (2024) (cit. on pp. 70, 74, 75, 77–79, 86, 87, 109).
- [Del+24] Blaise Delattre, Alexandre Araujo, Quentin Barthélemy, and Alexandre Allauzen. “The Lipschitz-Variance-Margin Tradeoff for Enhanced Randomized Smoothing”. In: *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024 (cit. on p. 24).
- [Fen+24] Yunzhen Feng, Tim GJ Rudner, Nikolaos Tsilivis, and Julia Kempe. “Attacking Bayes: On the Adversarial Robustness of Bayesian Neural Networks”. In: *arXiv preprint arXiv:2404.19640* (2024) (cit. on pp. 27, 36, 159).

- [Gur24] Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*. 2024 (cit. on p. 79).
- [TJK24] Nicolas Garcia Trillos, Matt Jacobs, and Jakwang Kim. “On the existence of solutions to adversarial training in multiclass classification”. In: *European Journal of Applied Mathematics* (2024), pp. 1–21 (cit. on pp. 4, 93, 144).

RÉSUMÉ

Le déploiement de modèles d'apprentissage automatique et profond dans diverses applications réelles nécessite une robustesse face aux attaques adversariales. Les attaques adversariales font référence à des manipulations délibérées des données d'entrée pour tromper ou induire en erreur les modèles d'apprentissage automatique, ce qui peut avoir des conséquences importantes dans des domaines critiques. Cette thèse se concentre sur l'étude du rôle de la randomisation dans la robustesse à ce type de perturbations. La thèse s'articule autour du concept de classificateurs aléatoires ou randomisés, qui diffèrent des classificateurs traditionnels en produisant des distributions de probabilité plutôt que des étiquettes. Ces classificateurs peuvent être considérés comme employant des stratégies mixtes, ce qui implique que les attaquer pourrait être plus difficile en raison de l'incertitude dans leurs réponses. Les objectifs premiers de cette recherche sont doubles. Premièrement, analyser les gains potentiels de robustesse que peuvent offrir les modèles randomisés par rapport à leurs homologues déterministes. Deuxièmement, identifier les conditions dans lesquelles ces modèles présentent des performances supérieures. La recherche met principalement l'accent sur l'analyse théorique pour comprendre les mécanismes sous-jacents responsables de la robustesse des classificateurs randomisés. En approfondissant les aspects théoriques, la thèse vise à mieux comprendre les raisons de la résilience accrue des modèles randomisés face aux attaques adversaires. Cette compréhension servira de base pour proposer des solutions pratiques pour entraîner des modèles robustes.

MOTS CLÉS

attaques adverses, perturbations adverses, attaques contradictoires, classification aléatoire, robustesse, stratégies mixtes, apprentissage automatique, apprentissage profond, classificateurs randomisés, randomisation

ABSTRACT

The deployment of machine and deep learning models in various real-life applications necessitates robustness against potential adversarial attacks. Adversarial attacks refer to deliberate manipulations of input data to deceive or mislead machine learning models, which can have significant consequences in critical domains. This thesis focuses on investigating the role of randomization in enhancing adversarial robustness. The thesis revolves around the concept of randomized classifiers, which differ from traditional classifiers by outputting probability distributions rather than hard labels. These classifiers can be seen as employing mixed strategies, implying that attacking them could be more challenging due to the increased uncertainty in their responses. The primary objectives of this research are twofold. Firstly, to analyze the potential gains in robustness that randomized models can offer compared to their deterministic counterparts. Secondly, to identify the conditions under which these models exhibit superior performance. The research primarily emphasizes theoretical analysis to understand the underlying mechanisms responsible for the enhanced robustness of randomized classifiers. By delving into the theoretical aspects, the thesis aims to shed light on the reasons behind the increased resilience of randomized models against adversarial attacks. This understanding will serve as a foundation for proposing practical solutions to train robust models based on the obtained theoretical insights. Ultimately, this thesis strives to contribute to the development of more robust machine learning models, capable of withstanding adversarial attacks. By bridging the gap between theory and practice, it seeks to provide practical guidelines and insights for training models that exhibit enhanced resilience against intentional manipulations.

KEYWORDS

adversarial classification, randomization, robustness, mixed strategies, machine learning, deep learning, adversarial attacks, randomized classifiers