



**HAL**  
open science

# Neural networks security under realistic scenario

Thibault Maho

► **To cite this version:**

Thibault Maho. Neural networks security under realistic scenario. Other [cs.OH]. Université de Rennes, 2023. English. NNT : 2023URENS121 . tel-04620428

**HAL Id: tel-04620428**

**<https://theses.hal.science/tel-04620428v1>**

Submitted on 21 Jun 2024

**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.

# THÈSE DE DOCTORAT DE

L'UNIVERSITÉ DE RENNES

ÉCOLE DOCTORALE N° 601

*Mathématiques, Télécommunications, Informatique, Signal, Systèmes,  
Électronique*

Spécialité : *Signal, Image, Vision,*

Par

**Thibault Maho**

## Security of Neural Networks under Realistic Scenario

Thèse présentée et soutenue à Inria, Rennes, le 14/12/2023

Unité de recherche : IRISA, CNRS, UMR 6074

### Rapporteurs avant soutenance :

William Puech                      Professeur, Université de Montpellier  
Fernando Pérez-Gonzalez      Professeur, Université de Vigo (Espagne)

### Composition du Jury :

|                    |                                |   |
|--------------------|--------------------------------|---|
| Président :        | Elisa Fromont                  | Université de Rennes  |
| Examineurs :       | Seyed-Mohsen Moosavi-Dezfooli, | Assistant professeur, Imperial College London (Royaume-Uni) |
|                    | Maura Pintor                   | Assistant professeur, Université de Cagliari (Italie)       |
| Dir. de thèse :    | Teddy Furon                    | Directeur de Recherche, Inria Rennes                        |
| Co-dir. de thèse : | Erwan Le Merrer                | Chargé de Recherche, Inria Rennes                           |



# ACKNOWLEDGEMENT

---

Au cours de cette thèse, de nombreuses personnes m'ont soutenu. Toutes ne seront pas mentionnées dans ce court message, mais beaucoup se reconnaîtront.

Tout d'abord, je tiens à sincèrement remercier mes encadrants exceptionnels. Teddy Furon et Erwan Le Merrer ont joué un rôle essentiel dans la réalisation de cette thèse. La première chose que Benoît Bonnet m'ait dite à mon arrivée, c'est à quel point j'avais de la chance de les avoir comme directeurs de thèse. Il a eu indubitablement raison. Ils ont rapidement su me plonger dans le monde de la recherche et grâce à eux j'ai rapidement publié. Leur présence, leur inspiration, leur mentorat, leur écoute et leur bienveillance en font des chercheurs et des personnes remarquables. Je souhaite à tous les doctorants de bénéficier d'un même encadrement.

Ensuite, je tiens à remercier mes collègues de l'équipe Linkmedia, qui sont devenus aujourd'hui mes amis. Merci à Benoît Bonnet, avec qui partager un bureau a été un véritable plaisir, ainsi que pour nos parties de pédantix dès que sonnait midi. Je remercie Pierre Fernandez pour nos discussions toujours passionnantes. Je tiens également à remercier Carolina Stephanie, Morgane Casanova, Hugo Thomas, Kassem Kallas, Karim Tit et tous les autres pour nos parties endiablées de baby-foot. Un grand merci à Aurélie Patier, avec qui un problème ne résiste pas longtemps, et qui fait preuve de patience avec des personnes parfois peu douées pour les tâches administratives. Je remercie enfin mes amis que je me suis faits à Rennes, Aurélie, Jérôme, Florent et les autres du Landry, pour ces moments de rire et de sport.

Enfin, je veux remercier des personnes qui m'ont plus que soutenu, sans qui je ne me serais peut-être jamais lancé dans cette aventure. Aujourd'hui, je ne regrette rien et les en remercie. Tout d'abord, le plus évident est ma famille avec des parents toujours présents et aimants. Ma thèse s'inscrit aussi dans une certaine continuité. Je remercie mes frères, Thomas et Pierre, qui m'ont inspiré et ont suscité en moi le désir de poursuivre cette thèse. Comme tout petit frère qui admire ses grands frères, j'ai eu envie de les imiter. Enfin, parmi ces personnes, une place particulière est réservée à Adélaïde Mouret qui a été mon pilier durant ces trois ans. Elle m'a apporté une aide quotidienne, m'a rassuré, motivé, soutenu, et parfois même poussé au-delà de mes limites. Sans elle, ma thèse aurait été différente et le chemin difficile. La vie aura eu raison de nous, mais je prends cette thèse comme héritage de cette période. J'espère avoir rendu toutes ces personnes fières de moi.



# TABLE OF CONTENTS

---

|   |           |
|---|-----------|
| <b>Résumé en Français</b>   | <b>11</b> |
| <b>Introduction</b>   | <b>19</b> |
| <b>I Introduction to Realistic Vulnerabilities of Neural Networks</b> | <b>24</b> |
| <b>1 Classifier Model</b>   | <b>25</b> |
| 1.1 Introduction . . . . .  | 25        |
| 1.2 Neural Networks . . . . .   | 26        |
| 1.2.1 Their Hegemony . . . . .  | 26        |
| 1.2.2 Mathematical Definition . . . . .                               | 28        |
| 1.2.3 Model Notation . . . . .  | 29        |
| 1.3 Decision Boundary of Neural Network Classifiers . . . . .         | 30        |
| 1.3.1 Definition . . . . .  | 30        |
| 1.3.2 Visualization . . . . .   | 30        |
| 1.3.3 Characteristics . . . . .                                       | 31        |
| 1.4 The Vulnerabilities of Neural Network Classifiers . . . . .       | 34        |
| 1.4.1 Definition of Security . . . . .                                | 34        |
| 1.4.2 A Realistic Scenario . . . . .                                  | 35        |
| 1.4.3 The Existing Vulnerabilities . . . . .                          | 38        |
| 1.5 Conclusion . . . . .  | 42        |
| <b>2 Confidentiality of Machine Learning Models</b>                   | <b>45</b> |
| 2.1 Introduction . . . . .  | 45        |
| 2.2 Attacks . . . . .   | 46        |
| 2.2.1 Extract Exact Model Attributes . . . . .                        | 46        |
| 2.2.2 Approximate behavior . . . . .                                  | 48        |
| 2.3 Defenses . . . . .  | 51        |
| 2.3.1 Reactive Defenses . . . . .                                     | 52        |

TABLE OF CONTENTS

---

|           |  |           |
|-----------|--|-----------|
| 2.3.2     | Proactive defenses . . . . .                                 | 54        |
| 2.4       | Conclusion . . . . .   | 54        |
| <b>3</b>  | <b>Adversarial Examples</b>                                  | <b>57</b> |
| 3.1       | Introduction . . . . .                                       | 57        |
| 3.1.1     | Illustration . . . . .                                       | 57        |
| 3.1.2     | Definition . . . . .   | 58        |
| 3.1.3     | Quality of Evasion Attacks . . . . .                         | 59        |
| 3.2       | Existence of Adversarial Examples . . . . .                  | 61        |
| 3.3       | Properties of Adversarial Examples . . . . .                 | 62        |
| 3.4       | Adversarial Attacks . . . . .                                | 65        |
| 3.4.1     | Transfer-Based Attacks . . . . .                             | 65        |
| 3.4.2     | Model Surrogate-Based Attacks . . . . .                      | 66        |
| 3.4.3     | Query-Based Attack . . . . .                                 | 67        |
| 3.4.4     | Comparison of Numbers of Queries Required . . . . .          | 69        |
| 3.5       | Defenses . . . . .   | 71        |
| 3.5.1     | Pre-preprocessing defenses . . . . .                         | 71        |
| 3.5.2     | Detection . . . . .  | 72        |
| 3.5.3     | Data Augmentation . . . . .                                  | 74        |
| 3.6       | Conclusion . . . . .   | 75        |
| <b>II</b> | <b>Contribution on Adversarial Examples</b>                  | <b>77</b> |
| <b>4</b>  | <b>SurFree [146]: a fast surrogate-free black box attack</b> | <b>79</b> |
| 4.1       | Introduction . . . . .                                       | 79        |
| 4.2       | Problem statement . . . . .                                  | 81        |
| 4.3       | Our Approach . . . . .                                       | 81        |
| 4.3.1     | Basic idea . . . . .   | 81        |
| 4.3.2     | Iterations over orthonormal directions . . . . .             | 84        |
| 4.3.3     | Convex boundary . . . . .                                    | 84        |
| 4.4       | The SurFree [146] attack . . . . .                           | 85        |
| 4.4.1     | The algorithm . . . . .                                      | 85        |
| 4.4.2     | Distribution of the directions . . . . .                     | 85        |
| 4.4.3     | Interpolation . . . . .                                      | 86        |

|          |   |            |
|----------|---|------------|
| 4.5      | Experimental Work . . . . .   | 86         |
| 4.5.1    | Datasets and Experimental Setup . . . . .                             | 88         |
| 4.5.2    | Ablation Studies . . . . .  | 89         |
| 4.5.3    | Benchmarking . . . . .  | 91         |
| 4.6      | Conclusion . . . . .  | 93         |
| <b>5</b> | <b>How to choose your best allies for a transferable attack?</b>      | <b>95</b>  |
| 5.1      | Introduction . . . . .  | 95         |
| 5.2      | Methodology . . . . .   | 97         |
| 5.2.1    | Notations . . . . .   | 97         |
| 5.2.2    | Measurement . . . . .   | 97         |
| 5.2.3    | Transferability . . . . .   | 97         |
| 5.2.4    | Practical implementation . . . . .                                    | 98         |
| 5.3      | Triad of transferability: data, model, attack . . . . .               | 99         |
| 5.3.1    | Experimental setup . . . . .  | 99         |
| 5.3.2    | Model dependence . . . . .  | 100        |
| 5.3.3    | Image dependence . . . . .  | 101        |
| 5.3.4    | Attack dependence . . . . .   | 102        |
| 5.4      | How to choose the best source . . . . .                               | 104        |
| 5.4.1    | Criterion $\text{ModSim}(s, t)$ . . . . .                             | 104        |
| 5.4.2    | Criterion $\text{TransQ}(s, x)$ . . . . .                             | 105        |
| 5.4.3    | Results . . . . .   | 105        |
| 5.5      | Conclusion . . . . .  | 109        |
| <b>6</b> | <b>Randomized Smoothing Under Attack: How Good Is It In Practice?</b> | <b>111</b> |
| 6.1      | Introduction . . . . .  | 111        |
| 6.2      | Related Work - Randomized smoothing (RS) . . . . .                    | 112        |
| 6.3      | Randomized smoothing: from theory to practice . . . . .               | 112        |
| 6.3.1    | A primer on random smoothing . . . . .                                | 113        |
| 6.3.2    | A critical point of view . . . . .                                    | 113        |
| 6.3.3    | Pushing the frontiers . . . . .                                       | 114        |
| 6.3.4    | Adversarial example with confidence level . . . . .                   | 115        |
| 6.3.5    | Jeopardizing black box attacks . . . . .                              | 116        |
| 6.4      | Black Box Attacks vs. RS . . . . .                                    | 116        |
| 6.4.1    | Experimental Setup . . . . .  | 116        |



|            |  |            |
|------------|--|------------|
| 6.4.2      | Evaluation Results . . . . .   | 117        |
| 6.5        | Conclusion . . . . .   | 119        |
| <b>III</b> | <b>Contribution on Model Confidentiality</b>   | <b>121</b> |
| <b>7</b>   | <b>Fingerprinting Classifiers with Benign Inputs</b>                                     | <b>123</b> |
| 7.1        | Introduction . . . . .   | 123        |
| 7.1.1      | Challenges . . . . .   | 123        |
| 7.1.2      | Our Rationale and Contributions . . . . .  | 124        |
| 7.2        | Threat Model . . . . .   | 126        |
| 7.2.1      | Bob: Keeping his Model Anonymous . . . . .   | 126        |
| 7.2.2      | Alice: Disclosing the Remote Model . . . . .   | 128        |
| 7.2.3      | The Classifier in the Black-Box . . . . .  | 129        |
| 7.2.4      | Summary . . . . .  | 129        |
| 7.3        | Walled Garden  |            |
|            | The Black-Box is a Known Model . . . . .   | 130        |
| 7.3.1      | Detection ( $\text{det}, \mathcal{F}, \mathcal{A} = \mathcal{B}, k$ ) . . . . .          | 130        |
| 7.3.2      | Identification ( $l, \mathcal{F}, \mathcal{A} = \mathcal{B}, k$ ) . . . . .              | 131        |
| 7.3.3      | Experimental Work . . . . .  | 132        |
| 7.4        | Open World:  |            |
|            | The Black-Box is an Unknown Model . . . . .  | 136        |
| 7.4.1      | Modeling . . . . .   | 136        |
| 7.4.2      | Detection ( $\text{det}, \mathcal{F}, \mathcal{A} \subsetneq \mathcal{B}, k$ ) . . . . . | 137        |
| 7.4.3      | Identification ( $l, \mathcal{F}, \mathcal{A} \subsetneq \mathcal{B}, k$ ) . . . . .     | 140        |
| 7.4.4      | Experimental Work . . . . .  | 141        |
| 7.5        | State-of-the-Art Benchmark . . . . .   | 147        |
| 7.5.1      | Previous Works . . . . .   | 147        |
| 7.5.2      | Fragile Fingerprinting . . . . .   | 148        |
| 7.5.3      | Robust Fingerprinting . . . . .  | 149        |
| 7.6        | Related Work . . . . .   | 149        |
| 7.7        | Conclusion . . . . .   | 150        |
|            | <b>List of Figures</b>   | <b>157</b> |
|            | <b>Notations</b>   | <b>161</b> |

|  |            |
|--|------------|
| <b>Bibliography</b>  | <b>162</b> |
| <b>A RoBIC [144]: A benchmark suite for assessing classifiers robustness</b> | <b>193</b> |
| A.1 Introduction . . . . .   | 193        |
| A.2 Difficulties . . . . .   | 194        |
| A.2.1 Notation . . . . .   | 194        |
| A.2.2 The best-effort mode . . . . .   | 194        |
| A.2.3 worst-case attacks . . . . .   | 195        |
| A.2.4 The choice of the metric . . . . .                                     | 195        |
| A.3 The benchmark . . . . .  | 195        |
| A.4 Fast Attacks . . . . .   | 196        |
| A.4.1 Fast black-box attacks . . . . .                                       | 196        |
| A.4.2 Fast white-box attacks . . . . .                                       | 197        |
| A.4.3 Quantization . . . . .   | 198        |
| A.5 Experiments . . . . .  | 198        |
| A.5.1 Selecting the worst case attacks . . . . .                             | 198        |
| A.5.2 Comparison with other benchmarks . . . . .                             | 199        |
| A.5.3 Benchmarking models . . . . .  | 200        |
| A.6 Conclusion . . . . .   | 201        |
| <b>B Appendix: Fingerprinting Classifiers with Benign Inputs</b>             | <b>203</b> |
| B.1 Description of the Set of Models . . . . .                               | 203        |
| B.2 Distance between Models . . . . .  | 204        |
| B.3 Proof of Equation (10) . . . . .   | 205        |
| B.4 Lower bound of the distance dist . . . . .                               | 206        |
| <b>C Appendix: How to choose your best allies for a transferable attack?</b> | <b>211</b> |
| C.1 Experimental Setup . . . . .   | 211        |
| C.2 Preliminaries . . . . .  | 212        |
| C.2.1 Epsilon Parameter . . . . .  | 212        |
| C.3 Transferability Dependences . . . . .                                    | 212        |
| C.4 Results . . . . .  | 213        |
| C.4.1 Fingerprinting . . . . .   | 213        |
| C.4.2 Ensemble model attack . . . . .  | 213        |

# CONTRIBUTIONS

---

**Thibault Maho**, Teddy Furon, and Erwan Le Merrer, « Surf-free: a fast surrogate-free black-box attack », in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. (See Chapter 4)

**Thibault Maho**, Teddy Furon, and Erwan Le Merrer, « RoBIC: A benchmark suite for assessing classifiers robustness », in: *IEEE International Conference on Image Processing*, 2021. (See Appendix A)

**Thibault Maho**, Teddy Furon, and Erwan Le Merrer, « Randomized Smoothing sous attaque: Théorie vs Pratique », in: *GRETSI 2022-XXVIIème Colloque francophone de traitement du signal et des images*, 2022.

**Thibault Maho**, Teddy Furon, and Erwan Le Merrer, « Randomized Smoothing under Attack: How Good is it in Practice? », in: *ICASSP*, 2022. (See Chapter 6)

**Thibault Maho**, Teddy Furon, and Erwan Le Merrer, « FBI: Fingerprinting models with Benign Inputs », in: *IEEE Transactions on Information Forensics and Security*, 2023. (See Chapter 7)

**Thibault Maho**, Teddy Furon, and Erwan Le Merrer, « Empreinte de réseaux avec des entrées authentiques », in: *Conference on Artificial Intelligence for Defense*, 2022.

**Thibault Maho**, Seyed-Mohsen Moosavi-Dezfooli, and Teddy Furon, « How to choose your best allies for a transferable attack? », in: *Proceedings of the IEEE/CVF international conference on computer vision*, 2023. (See Chapter 5)

**Thibault Maho**, Seyed-Mohsen Moosavi-Dezfooli, and Teddy Furon, « Comment choisir son meilleur allié pour une attaque transférable ? », in: *GRETSI 2023-XXVIIIème Colloque francophone de traitement du signal et des images*, 2023.

# RÉSUMÉ EN FRANÇAIS

---

Le terme d'*Intelligence Artificielle* (IA) est apparu en 1956. Il peut être défini comme le développement d'algorithmes, de modèles ou de systèmes capables d'effectuer des tâches qui requièrent traditionnellement l'intelligence humaine. Mais comme l'a dit Pamela McCorduck, "l'IA a commencé avec l'ancien désir de jouer à Dieu" et créer une machine imitant la cognition humaine remonte à plus loin, comme on peut le voir dans les automates d'Héphaïstos, les premières tentatives de réplique d'entités vivantes. Bien que la terminologie ait évolué, l'ambition sous-jacente est restée la même.

Ce n'est qu'au cours des dernières années que ces contes ont commencé à prendre vie. L'IA est devenue une révolution comparable au Web 2.0 du début des années 2000. Selon Bill Gates, l'IA est la progression la plus remarquable depuis l'introduction de l'interface utilisateur graphique pour les ordinateurs personnels. Cette perspective se reflète dans l'intérêt que nous portons à l'IA. Cette expansion englobe des domaines tels que les véhicules autonomes [260] et des innovations plus récentes, notamment des modèles génératifs tels que ChatGPT [171] et StableDiffusion [192]. Par exemple, selon les estimations de McKinsey, l'IA générative pourrait générer une valeur économique annuelle comprise entre 2,6 et 4,4 billions de dollars [30].

La tâche d'IA la plus courante est la classification, dans laquelle un modèle est formé et utilisé pour prédire les labels pour des données d'entrée. Cependant, l'IA englobe de nombreuses autres tâches telles que la segmentation [275], le clustering [32] et la régression [76]. Ces systèmes d'IA apprennent des modèles et des relations à partir de données d'apprentissage en utilisant diverses techniques. Ils impliquent la création de modèles mathématiques capables de se généraliser à partir des données d'apprentissage pour faire des prédictions sur de nouvelles données. L'essor de l'IA dans notre vie quotidienne est en partie dû au succès des réseaux de neurones, qui excellent dans la manipulation de divers types de données tels que les images [262, 23], l'audio [165, 222] et le texte [274, 159, 286]. Par conséquent, cette thèse se concentrera sur les réseaux de neurones, le domaine des images et la tâche de classification en raison de leur importance dans la recherche moderne en IA.

Chaque révolution comporte des défis et des faiblesses, et l'IA ne fait pas exception. L'attention importante entourant l'IA a incité à renforcer sa législation. Le récent acte européen sur l'IA<sup>1</sup>

---

1. Acte de l'UE sur l'IA : <https://www.europarl.europa.eu/news/en/headlines/society>

en est un excellent exemple, puisqu'il s'agit de la première loi réglementant l'IA. La priorité du parlement est de veiller à ce que les systèmes d'IA utilisés en Europe soient "sûrs, transparents, traçables, non discriminatoires et respectueux de l'environnement". La législation proposée introduit une classification des risques liés à l'IA. Bien que l'IA soit aujourd'hui principalement utilisée dans des domaines à faible risque ou sans risque, la liste des domaines à haut risque est impressionnante, englobant des domaines tels que la catégorisation des individus, l'utilisation de l'IA dans l'éducation, les services publics et privés, et l'application de la loi.

À mesure que les gouvernements s'impliquent dans le débat, la sécurité de l'IA et de ses composantes devient une préoccupation majeure dans son développement et sa prolifération. Cela devient un véritable problème lorsqu'il s'avère que l'IA peut être facilement manipulée, modifiée ou volée. Cette thèse s'aligne parfaitement sur le problème de la sécurité de l'IA dans les applications du monde réel et s'y attaque.

Dans un scénario réaliste de la sécurité de l'IA, une configuration en boîte noire prévaut, comme il sera expliqué dans le chapitre 1.4.2. Cela implique une absence d'informations sur le modèle. Même dans cette hypothèse, les vulnérabilités potentielles sont nombreuses:

- Récupération des données d'apprentissage: elles peuvent être inférées ou récupérées par des méthodes telles que l'inférence d'appartenance [84, 29] et l'inversion de modèle [48, 107, 63].
- Vol de propriété intellectuelle: les algorithmes d'IA peuvent être volés [276, 71, 102], ce qui porte atteinte à la propriété intellectuelle et aux efforts investis dans leur conception.
- Attaque par évadation [187]: les données d'entrée peuvent être manipulées et engendrer une mauvaise classification en utilisant des exemples adverses [221].
- Exploitation des ressources: les entrées peuvent être falsifiées pour augmenter la consommation d'énergie, pouvant entraîner un ralentissement du système voire des pannes, comme le montrent les exemples éponges [213].
- Porte dérobée: les données d'entraînement peuvent être manipulées pour insérer une porte dérobée [10, 232], permettant à une entité en possession de la clé de manipuler les sorties.

Une vue plus complète des possibilités est présentée dans Table 1

Sur cette base, nous entrons dans l'organisation de ce document. La première partie de cette thèse comprend trois chapitres qui, ensemble, établissent le contexte de l'ensemble du travail. Ces chapitres comprennent des définitions essentielles et un examen de la littérature scientifique sur ce sujet. Les deux parties suivantes seront respectivement consacrées aux contributions sur

|                                | <b>Confidentialité</b>  | <b>Intégrité</b>                                      | <b>Disponibilité</b>      |
|--------------------------------|---|---|---------------------------|
| <b>Modèle</b>                  | Extraction de Modèle [71, 102]<br>et d'Attributs [11, 53]         |   |                           |
| <b>Données d'Apprentissage</b> | Inversion de modèle [48, 63]<br>Inférence d'Appartenance [84, 29] | Données Radioactives [196]<br>Porte Dérobée [10, 232] |                           |
| <b>Inférence</b>               |   | Attaque par Evasion [146, 221]                        | Données Eponges [213, 31] |

Table 1 – Vulnérabilités des réseaux de neurones sous scénario réaliste (Voir Section 1.4.2) organisé en fonction de la triade CID (Voir Section 1.4.1).

les exemples adverses et à la contribution sur la confidentialité des modèles.

## Organisation du Manuscrit

### Partie I: Introduction aux Vulnérabilités des Réseaux de Neurones

**Le Chapitre 1** jette les bases de ce manuscrit. Il présente les réseaux de neurones largement utilisés pour la tâche de classification ainsi que les différentes propriétés de leurs frontières de décision. Les principes de la sécurité sont également abordés, en utilisant la triade CID (Confidentialité, Intégrité, Disponibilité) couramment employée dans le domaine de la cybersécurité pour classer les différentes vulnérabilités. Cette triade est adaptée ici pour traiter les vulnérabilités liées à l'apprentissage automatique. Enfin, le chapitre introduit le scénario réaliste qui sera étudié tout au long du manuscrit, suivi d'une brève liste des vulnérabilités potentielles dans ce contexte.

**Le Chapitre 2** se concentre sur les vulnérabilités pouvant compromettre la confidentialité du modèle. Il présente diverses attaques, classifiées selon les objectifs des attaquants tels que le vol de paramètres, le vol d'architecture ou la création d'un modèle copiant le modèle attaqué. Ensuite, les défenses sont introduites, en mettant l'accent sur la vérification des propriétés d'un modèle et les défenses visant à augmenter la complexité des attaques.

**Le Chapitre 3** suit une structure similaire au chapitre précédent en se concentrant sur les exemples adverses. Après une brève définition et une introduction à leurs propriétés, l'hypothèse

principale de leur existence est exposée. Ensuite, les attaques et les défenses sont présentées.

## Partie II: Contribution aux Exemples Adverses

**Le Chapitre 4** présente une nouvelle attaque boîte noire basée sur la décision. Il met en lumière l'utilisation excessive de requêtes dans les attaques boîtes noires actuelles pour estimer le gradient de la frontière de décision. En utilisant une approche géométrique et en réduisant le coût d'estimation au minimum, ce chapitre introduit `SurFree` [146], qui permet la génération d'exemples adverses avec un nombre significativement plus faible de requêtes que les attaques boîtes noires existantes.

**Le Chapitre 5** explore la transférabilité des exemples adverses. Parmi les nombreuses attaques proposées, le choix du modèle source et cible est souvent arbitraire. Cette contribution met en évidence la grande variabilité du succès du transfert en fonction du couple de modèles considéré, de l'image attaquée et de l'attaque utilisée. En se basant sur cette observation, ce chapitre introduit une estimation de la qualité de la transférabilité de l'exemple adverse trouvé par l'attaque. Cette mesure est ensuite utilisée pour permettre à l'attaquant de sélectionner le meilleur modèle source et la meilleure attaque, permettant la plus faible perturbation pour une image donnée.

**Le Chapitre 6** évalue une méthode de certification appelée *Randomized Smoothing* en tant que défense contre les attaques d'évasion en boîte noire. Il met en évidence son efficacité, mais également que les paramètres recommandés pour garantir une bonne certification s'opposent à ceux qui assurent son efficacité contre les attaques d'évasion.

## Partie III: Contribution à la Confidentialité du modèle

**Le Chapitre 7** est la seule contribution qui s'intéresse à prouver la propriété d'un modèle en proposant une nouvelle empreinte de réseaux n'utilisant que des images naturelles. Cette approche repose sur une observation simple: bien que les méthodes d'empreintes de réseaux existantes soient efficaces, elles nécessitent la construction d'exemples adverses avec des propriétés particulières, parfois complexes. L'idée d'utiliser des images bénignes a été rapidement abandonnée dans la littérature car jugée inefficace et peu informative. Cette contribution montre cependant l'inverse en utilisant la théorie de l'information, qui permet de mesurer la dépendance

statistique entre deux réseaux. Le travail s'appuie sur l'hypothèse que deux modèles qui ont des prédictions statistiquement dépendantes proviennent d'un même modèle parent.

## **Conclusion**

L'objectif principal de cette thèse était d'étudier les vulnérabilités de l'IA dans un cadre réaliste. Dans ce scénario particulier, les capacités de l'attaquant sont sévèrement limitées, à la fois en termes de connaissances et de ressources.

Dans le premier chapitre, ce scénario et ses définitions associées ont été présentés. Il est remarquable que, même dans ce contexte restreint, un nombre important de vulnérabilités identifiées à l'origine dans la littérature "boîte blanche" soient également réalisables dans la configuration "boîte noire". Le scénario dans lequel nous nous sommes placés a donné lieu à trois axes principaux: la confidentialité du modèle, la manipulation des entrées d'inférence et la confidentialité des données d'entraînement.

Chacun de ces domaines a été brièvement présenté, l'accent étant ensuite mis sur deux aspects importants, à savoir les exemples adverses et la confidentialité des modèles, explorés respectivement dans les chapitres 2 et 3. Ce sont des domaines dans lesquels des contributions importantes ont été apportées tout au long de cette thèse.

Le chapitre 1 aborde également les subtilités de la frontière de décision et ses caractéristiques, qui servent de porte d'entrée aux attaques. Étant donné le manque d'informations accessibles dans ce scénario, de nombreuses stratégies tirent parti de ses propriétés. Elle reflète essentiellement le processus d'apprentissage du modèle et ses interactions avec les données. Une compréhension approfondie de la frontière de décision donne lieu à une meilleure compréhension des vulnérabilités du modèle. Par exemple, les exemples adverses occupent une position centrale dans cette thèse, à la fois dans l'aperçu de l'état de l'art dans Chapter 3 et à travers les nombreuses contributions présentées dans la partie II. La prémisse fondamentale de leur existence et les nombreuses méthodes présentées soulignent l'importance critique de la compréhension de la frontière de décision en termes de vulnérabilités du modèle.

Le point focal de cette thèse a été la sécurité des réseaux de neurones, en se concentrant spécifiquement sur la tâche de classification d'images. Ce choix est motivé par l'importance de ces tâches dans la littérature existante. Néanmoins, les menaces et les méthodologies présentées tout au long de cette thèse sont applicables à diverses tâches et ensembles de données. Par exemple, une étude récente [291] démontre que le concept d'exemples adverses s'étend aux modèles de langage. Dans de tels contextes, les entrées peuvent être manipulées pour produire



des sorties indésirables ou interdites.

## Limitations

Enfin, cette thèse a pris en compte un scénario réaliste. Cependant, il existe des raisons de remettre en question si le scénario présenté dans Section 1.4.2 est réellement "réaliste". Plusieurs facteurs peuvent susciter des doutes quant à ce scénario et à notre définition associée. Tout d'abord, l'utilisation d'ImageNet dans la majorité de nos expériences peut être remise en question. En effet, les images, en raison de leur taille et surtout de leurs classes, ne représentent pas nécessairement des images du monde réel. Par exemple, au sein d'ImageNet, il existe 118 races différentes de chiens sur un total de 1000 classes. Certaines classes présentent une grande similitude, comme "projectile, missile" et "missile", entraînant, aussi bien pour l'humain labelisant le jeu de données que pour le modèle d'apprentissage, des erreurs fréquentes [163]. Cela soulève des questions sur l'applicabilité de toutes les expériences de nos contributions et de la plupart des travaux dans un contexte réel.

De plus, le déploiement futur des modèles en production devrait être dynamique plutôt que statique. Les pratiques MLOps, conçues pour faciliter l'intégration des modèles d'IA dans les applications de la vie quotidienne et s'adapter à l'apport continu de nouvelles données, gagnent en importance. Les capacités de MLOps permettent la révision automatique et continue des modèles d'apprentissage selon un calendrier défini. Par conséquent, les modèles sont soumis à des changements fréquents, rendant les attaques potentiellement plus complexes. Par exemple, dans une configuration en boîte noire, les attaques basées sur l'historique des requêtes peuvent devenir obsolètes. De plus, les résultats d'une attaque d'inférence d'appartenance pourraient varier en fonction du moment où l'attaque a été menée.

Une autre limite vient des attaques par évasion. Lors de l'élaboration d'exemples adverses, les attaques boîte noire basées sur la décision impliquent une dichotomie pour se placer sur la frontière de décision, comme le montre la section 3.4.3. Au cours de cette phase, l'attaquant oscille entre la classe qu'il souhaite éviter et la classe cible. Dans le monde que nous avons défini, il n'y a pas de distinction dans les résultats des requêtes; toutes les réponses sont égales. Cependant, dans un contexte plus sensible, cela peut poser des problèmes. Dans de tels scénarios, l'attaquant peut générer de nombreuses requêtes "indésirables" au cours du processus de création d'exemples adverses. Par exemple, dans un cas décrit dans [45], un attaquant visant à télécharger une image avec un contenu inapproprié pourrait utiliser une attaque par évasion contre le modèle conçu pour détecter un tel contenu. Néanmoins, les méthodes actuelles d'attaque

par boîte noire basées sur la décision sont susceptibles de générer un grand nombre de ces requêtes "indésirables", principalement au cours de l'étape de dichotomie. Une seule de ces requêtes peut entraîner l'interdiction de l'attaquant ou, pire, une action en justice. Le développement futur des attaques de boîte noire basées sur la décision doit se concentrer sur l'atténuation de ces risques en limitant le nombre de requêtes associées à une classe particulière.

## **Perspectives**

Le domaine des exemples adverses est le sujet le plus exploré dans le domaine de la sécurité de l'apprentissage automatique. De nouvelles méthodologies pour les générer, de nouveaux mécanismes de défense et des théories sur leur existence sont fréquemment formulées. Cependant, il est plausible que ces défis ne possèdent pas de solution universellement applicable. Les caractéristiques inhérentes aux données d'entrée de haute dimension que nous traitons, la surparamétrisation qui prévaut dans la plupart des modèles, ainsi que la qualité et la quantité des données d'entraînement disponibles contribuent à leur existence et pourrait ne pas donner lieu à une solution. Mais la question est peut-être de savoir s'il est problématique de ne pas disposer d'une solution unique. L'interrogation elle-même est peut-être mal formulées. Les exemples adverses, tout comme leurs équivalents humains dans le monde réel (comme l'illustrent les illusions d'optique dans le chapitre 3), pourraient être un aspect inhérent de l'IA. L'idée n'est peut-être pas de les éliminer purement et simplement, mais plutôt de coexister avec eux, d'en démêler les subtilités et de les gérer par des stratégies spécifiques au cadre d'utilisation.

Le paragraphe précédent s'interroge sur la pertinence des exemples adverses et le désir de les éliminer. Cependant, cela ne remet pas en question leur valeur. La guerre entre l'attaquant et le défenseur n'a pas de fin, mais les exemples adverses constituent également un outil précieux pour comprendre le fonctionnement interne des réseaux de neurones [170]. Par exemple, considérons les exemples contrefactuels [103]. Ces exemples révèlent les changements nécessaires dans une entrée pour obtenir une sortie prédéfinie. Bien qu'ils soient adverses, leur objectif n'est pas de tromper un modèle, mais plutôt de l'interpréter. De plus, c'est une relation gagnant-gagnant; les exemples adverses contribuent à l'interprétation du modèle, et cette interprétation, à son tour, améliore notre compréhension et notre capacité à lutter contre les exemples adverses [17, 62]. Ainsi, la recherche sur les exemples adverses doit se poursuivre, mais peut être sous un angle différent.

L'ambition d'une mesure d'évaluation complète et équitable a été exprimée par les contributions faites dans cette thèse. Les contributions de `ROBIC` [144] dans Appendix A et `FIT` dans

Chapter 7 introduisent respectivement une nouvelle mesure de la robustesse du modèle et un cadre pour évaluer la transférabilité entre modèles source et cible. Compte tenu du flux constant de documents de recherche, l'absence de mesures objectives est souvent flagrante.

Dans le domaine de l'apprentissage automatique, la confidentialité est d'une importance capitale, notamment en ce qui concerne les modèles et les données. La protection des modèles est vitale en raison des implications financières et des avantages concurrentiels technologiques. Mais il est tout aussi important de s'aligner sur les demandes et les attentes des clients. Une statistique de l'INSEE révèle qu'environ 82% des individus protègent leurs données personnelles sur internet<sup>2</sup>. Des initiatives telles que la loi de l'Union européenne sur l'IA et le règlement général sur la protection des données (RGPD) soulignent l'impératif de la sécurité des données. Au cours de cette thèse, nous n'avons pas eu le temps d'étudier en profondeur la confidentialité des données. Cependant, dans la section 1.4.3, nous avons présenté l'inférence d'appartenance et les attaques par inversion de modèle comme des méthodes permettant d'extraire des informations sur les données d'apprentissage dans un scénario réaliste. Des solutions visant à sécuriser, contrôler et superviser ces actifs ont été proposées pour répondre à ces préoccupations. De grands travaux restent encore à faire.

## **Derniers Mots: Nouveaux Défis, Anciens Problèmes**

Cette thèse est parfaitement en phase avec son époque. L'IA est sur toutes les lèvres, et sa sécurité est devenue un sujet actuel et pressant. Cependant, bon nombre des problèmes abordés ne sont pas nouveaux. Tout au long de cette thèse, nous avons fait référence à des travaux antérieurs qui précèdent l'utilisation généralisée des réseaux de neurones. Par exemple, le concept de tatouage de modèle puise son inspiration dans des techniques de tatouage d'images datant de 1990 [226]. Les attaques contre les tatouages peuvent être considérées comme des exemples adverses, où l'objectif est de tromper un détecteur. Les moyens de s'en prémunir sont toujours d'actualité et pertinentes contre les attaques par évocation actuelles, mais elles ne sont pas prises en compte (voir Section 3.5). La notion d'empreinte de réseau a été appliquée précédemment, dans le domaine des sites web, pour ne citer que cet exemple. De nombreux exemples existent. De nombreuses méthodes ont été réintroduites et appliquées aux réseaux de neurones, même si des améliorations avaient été proposées précédemment.

Certains travaux antérieurs ont tenté d'établir des liens avec des recherches antérieures, servant ainsi de rappel sur ce qui a d'ores et déjà été réalisé et montrant que la recherche

---

2. <https://www.insee.fr/fr/statistiques/6475020>

en sécurité n'a pas commencé en 2012. Un exemple parmi les contributions est la méthode `SurFree` [146]. Il s'appuie sur une idée de John Earl de 2007 [55]. Dans le but de combler le fossé entre les domaines du tatouage numérique et de l'apprentissage automatique, une étude menée par Erwin Quiring *et al.* [183] a présenté une notation unifiée pour les attaques en boîte noire sur l'apprentissage automatique et sur le tatouage numérique. Ils ont démontré l'efficacité des stratégies de chaque domaine dans leurs homologues respectives. Par exemple, ils ont souligné comment les contre-mesures du tatouage numérique pouvaient atténuer les récentes attaques d'extraction de modèle.

Dans la course à la publication, l'examen des travaux précédents est parfois négligé. Cependant, plonger dans la littérature existante, même la plus ancienne, est essentiel pour comprendre les origines des problèmes et des solutions proposées. Ce processus permet d'identifier des lacunes dans les connaissances et d'éviter de réinventer des solutions existantes. La négligence des recherches antérieures peut résulter d'un manque de familiarité avec la littérature ou même d'un désir de dissimuler les origines d'une méthode. Dans les deux cas, une telle négligence est préjudiciable à la progression de la recherche et à la communauté scientifique dans son ensemble.

Naviguer sur ce terrain est certainement un défi. Néanmoins, le domaine de la sécurité de l'IA est étroitement lié à divers autres domaines. Par conséquent, il est essentiel de ne pas négliger les contributions des chercheurs précédents, mais plutôt de les exploiter afin d'améliorer et de peaufiner les méthodologies existantes. Cette approche pourrait entraîner des gains de temps et des améliorations significatives.

# INTRODUCTION

---

The term *artificial intelligence* (AI) emerged in 1956. It may be defined as the development of algorithms, models, or systems capable of performing tasks that traditionally need human intelligence. But as Pamela McCorduck said 'AI began with the ancient wish to forge the gods' and create a machine mimicking human cognition dates back further as observed in the automations of Hephaestus, early attempts at replicating living entities. Although the terminology may have evolved, the underlying ambition remains consistent.

It's only in recent years that these tales have started to come to life. AI has become a revolution comparable to the Web 2.0 of the early 2000s, and we are merely at the beginning of it. According to Bill Gates, AI is the most remarkable progression since the introduction of the graphical user interface for personal computers. This perspective is mirrored in our fervent interest in AI. This expansion encompasses domains like autonomous vehicles [260], and more recent innovations, including generative models like ChatGPT [171] and StableDiffusion [192]. For instance, McKinsey's estimation suggests that generative AI could generate an annual economic value ranging from 2.6 to 4.4 trillion dollars [30].

The most common AI task is classification, wherein a model is trained and employed to predict labels for a given input data. However, AI encompasses numerous other tasks such as segmentation [275], clustering [32], and regression [76]. These AI systems learn patterns and relationships from training data using diverse techniques. They involve the creation of mathematical models capable of generalizing from training data to make predictions on unseen data. The rise of AI in our daily lives is partly due to the success of neural networks, which excel in manipulating various data types such as images [262, 23], audio [165, 222] and text [274, 159, 286]. Consequently, this thesis will concentrate on neural networks, the image domain, and the classification task due to their prominence in AI research.

Revolution brings challenges and weaknesses, and AI is no exception. The sensitivity surrounding AI has prompted the strengthening of its legislation. The recent European AI Act<sup>3</sup> is a prime example, being the first law regulating AI. The priority for the parliament is to ensure that AI systems used in Europe are 'safe, transparent, traceable, non-discriminatory, and envi-

---

3. EU AI Act: <https://www.europarl.europa.eu/news/en/headlines/society/20230601STO93804/eu-ai-act-first-regulation-on-artificial-intelligence>

ronmentally friendly’. The proposed legislation introduces a classification of AI risks. Although AI is predominantly employed in low-risk or non-risk domains today, the list of high-risk domains is impressive, encompassing areas such as the categorization of individuals, AI usage in education, public and private services, and law enforcement.

As governments become involved in the AI debate, the security of AI and its components emerges as a paramount concern in its development and proliferation. It becomes a real issue when AI is shown to be easily manipulated, modified, or stolen. This thesis aligns perfectly and tackles the problem of the security of AI in real-world applications.

In a realistic perspective of AI security, a black-box setup prevails, as elucidated in Section 1.4.2. This implies a lack of information about the model. Even under this assumption, the potential vulnerabilities are numerous and include:

- Retrieval of training data: Training data can be inferred or retrieved through methods such as membership inference [84, 29] and model inversion [48, 107, 63].
- Intellectual property theft: AI algorithms can be stolen [276, 71, 102], undermining intellectual property and the efforts invested in their design.
- Evasion attacks [146, 187]: Input can be manipulated to induce misclassification using adversarial examples [221].
- Resource exploitation: Inputs can be forged to increase power consumption, potentially leading to system crashes, as demonstrated by sponge examples [213].
- Backdoor: Training data can be manipulated to insert a backdoor [10, 232], allowing an entity in possession of the key to manipulate outputs.

A more complete view of the possibilities is depicted in Table 2

|                      | <b>Confidentiality</b>                                       | <b>Integrity</b>                             | <b>Availability</b>                            |
|----------------------|--|--|--|
| <b>Model</b>         | Model Extraction [71, 102]<br>Attributes Extraction [11, 53] |  |  |
| <b>Training Data</b> | Model Inversion [48, 63]<br>Membership Inference [84, 29]    | Radioactive Data [196]<br>Backdoor [10, 232] |  |
| <b>Inference</b>     |  | Evasion Attack [221]                         | Sponge Examples [213]<br>Sponge Poisoning [31] |

Table 2 – Vulnerabilities of neural networks under a realistic scenario (See Section 1.4.2) organized in function of the CIA triad (See Section 1.4.1).

With this foundation, we delve into the organization of this manuscript. The initial part of this thesis comprises three chapters, which collectively establish the context for the entire work. These chapters encompass essential definitions and a comprehensive review of related literature.

The two following parts will respectively focus on the contribution on adversarial examples and the contribution on model confidentiality.

## Organization of the Manuscript

### Part I: Introduction to Neural Network Vulnerabilities

**Chapter 1** lays the foundation of this manuscript. It introduces widely used neural networks for classification tasks and discusses various properties of their decision boundaries. Principles of security are also introduced, employing the CIA triad (Confidentiality, Integrity, Availability) commonly used in the cybersecurity domain to categorize different vulnerabilities. This triad is adapted here to address vulnerabilities related to machine learning. Finally, the chapter introduces the realistic scenario that will be studied throughout the manuscript, followed by a brief list of potential vulnerabilities within this context.

**Chapter 2** focuses on vulnerabilities that could compromise the confidentiality of the model. It presents various attacks categorized according to attackers' objectives, such as parameter theft, architecture theft, or creating a model copying the target model. Subsequently, defenses are introduced, emphasizing model property verification and defenses aimed at increasing the complexity of attacks.

**Chapter 3** follows a similar structure to the previous chapter, focusing on adversarial examples. After a brief definition and an introduction to their properties, the main hypothesis behind their existence is presented. Then, both attacks and defenses are discussed.

### Part II: Contributions to Adversarial Examples

**Chapter 4** introduces a novel decision-based black-box attack. It highlights the excessive use of queries in current black-box attacks to estimate the decision boundary gradient. By employing a geometric approach and minimizing the estimation cost, this chapter introduces `SurFree` [146], which enables the generation of adversarial examples with significantly fewer queries than existing black-box attacks.

**Chapter 5** explores the transferability of adversarial examples. Among the numerous proposed attacks, the choice of source and target models is often arbitrary. This contribution reveals

---

the considerable variability in transfer success based on the considered model pair, attacked image and used attack. Based on this observation, the chapter introduces an estimation of the transferability quality of the adversarial example found by the attack. This measure is then utilized to enable the attacker to select the best source model and attack, resulting in the least perturbation for a given image.

**Chapter 6** assesses a certification method called *Randomized Smoothing* as a defense against black-box evasion attacks. It highlights its effectiveness but also reveals that the recommended parameters for ensuring good certification contradict those that ensure effectiveness against evasion attacks.

### **Part III: Contribution to Model Confidentiality**

**Chapter 7** stands as the sole contribution addressing model property verification by proposing a novel network fingerprint using only natural images. This approach is rooted in a simple observation: while existing network fingerprinting methods are effective, they often require the construction of adversarial examples with specific and sometimes complex properties. The idea of using benign images was quickly dismissed in literature due to being considered inefficient and uninformative. However, this contribution demonstrates the opposite by utilizing information theory to measure statistical dependence between two networks. The work builds upon the assumption that two models with statistically dependent predictions originate from a common parent model.

*A concise summary of all the notations is provided at the conclusion of this manuscript.. The reader can refer to it for a quick reminder. Readers seeking a quick reference can turn to this section, organized by contribution, with the global notation available at Table 7.9.*



PART I

# **Introduction to Realistic Vulnerabilities of Neural Networks**

---

# CLASSIFIER MODEL

---

## 1.1 Introduction

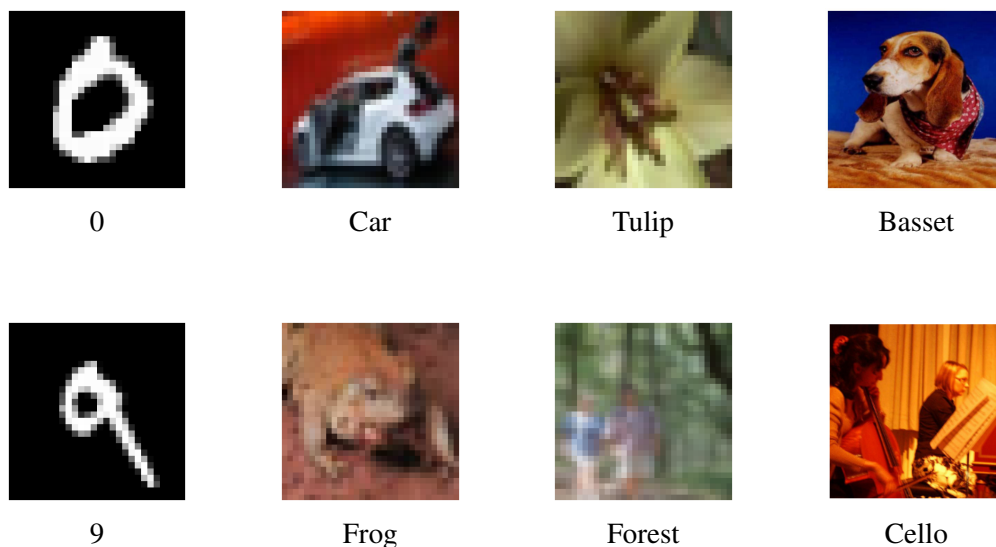


Figure 1.1 – Samples from different image datasets. From left to right: MNIST [229], CIFAR-10 [115], CIFAR-100 [115], and ImageNet [46].

Machine learning is a broad field that encompasses methods to enable computers to learn from data, identify patterns, and make decisions with minimal human intervention. It is a sub-field of artificial intelligence and covers a wide range of tasks, including classification. Classification is a fundamental learning task that aims to find a function that can predict the class of a given input. This task applies to various types of data, such as image with object identification, text with sentiment analysis, and sound with speaker identification, among others.

In the domain of Computer Vision, the main objective is image classification. Images are represented as tensors with dimensions  $D = N_c \times H \times W$ , where  $N_c$  denotes the number of channels,  $H$  represents the image's height, and  $W$  signifies its width. Several image datasets exist for this purpose, including MNIST [229], which contains 60,000 images of size  $1 \times 28 \times 28$

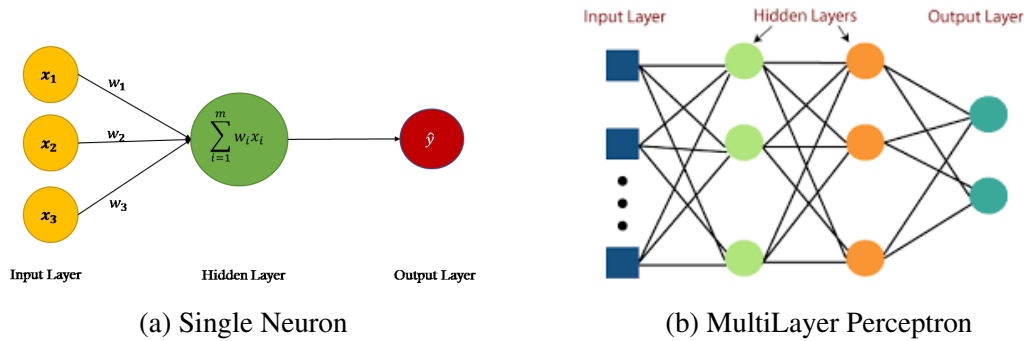


Figure 1.2 – From the idea of the neuron to the MultiLayer Perceptron.

distributed across 10 classes, and CIFAR-10 [115], comprising 60,000 images of size  $3 \times 32 \times 32$  across 10 classes. ImageNet [46] is another significant dataset in this field, consisting of 14 million images split into 1,000 classes. Examples of different image datasets are shown in Figure 1.1. ImageNet includes real-sized images, making it the widely used dataset in most recent works on classification tasks [23, 290, 24, 280, 144, 156]. In this thesis, the focus will be on images of size  $3 \times 224 \times 224$  from the ImageNet dataset.

As we delve into the field of machine learning, one method stands out and captures most of the attention: deep neural networks [44, 113]. Over the past decade, neural networks have emerged at the forefront of high-performing models in various tasks, particularly in classification. Let’s explore their history and some of their key properties.

## 1.2 Neural Networks

### 1.2.1 Their Hegemony

#### Difficult Beginnings

Neural networks, a type of machine learning model inspired by the human brain’s structure and function, were first introduced in 1943 when Warren McCulloch and Walter Pitts defined the initial mathematical model of a neuron [151]. The concept was simple, as depicted in Figure 1.2a: a neuron is composed of a weighted sum of its inputs and a non-linear activation function. The output was binary, determined by a threshold function. Initially, neural networks were capable of computing simple logical functions.

It was the first step in creating artificial neural networks that could learn from data. In the 1950s and 1960s, researchers like Frank Rosenblatt and Bernard Widrow pioneered early ver-

sions of neural networks, such as the perceptron [194] and the multi-layer perceptron [100] shown in Figure 1.2b. These networks were capable of learning to classify patterns. However, several limitations emerged, including the lack of computational power, large annotated datasets, and advancements in neural network training. As a result, this field declined in the 1970s and 1980s.

### A Sensational Comeback

The resurgence of interest in neural networks began at the end-1980s when Geoffrey Hinton and David Rumelhart laid the foundations for backpropagation [195, 88, 69], which enabled the training of deep networks. This breakthrough allowed algorithms, layers, and architectures that were conceptualized 40 years ago to be used for solving real-world problems.

An example of this revolution is LeNet-5 [121], which was developed by Yann LeCun and his colleagues in 1998. LeNet-5 combined the idea of convolutional neural networks (CNNs) imagined by Kunihiro Fukushima with the NeoCognitron [64] and the backpropagation algorithm. This CNN architecture was particularly well-suited for image classification tasks and found applications in various domains, including banks, where it was used to recognize handwritten numbers of size  $32 \times 32$  on checks [229].

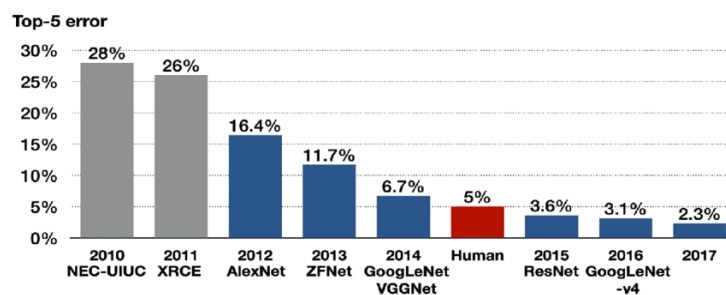


Figure 1.3 – Top-5 error evolution for ImageNet Classification Competition.

With the increasing computational power and the new advances in training, two of the three former limitations have been solved. The lack of labeled data, the last remaining challenge, was overcome with the introduction of the ImageNet dataset [46] in 2009. This dataset was used for the ImageNet Large Scale Visual Recognition Challenge (ILSVRC).

In 2012, the ILSVRC competition was won by the CNN AlexNet [116] with a large margin over the other methods. In subsequent years, neural networks' presence in the competition grew steadily, culminating in a major milestone in 2015 when the winning model's error rate was lower than the human error rate. This significant achievement is illustrated in Figure 1.3.

From this point, the number of publications on neural networks has exploded. Each year, the number of publications on neural networks has doubled. It reached 100 new publications on machine learning per day in 2019 [44, 113].

This excitement in neural networks has also led to a plethora of new architectures. Time series, like images with CNNs, have seen significant advancements with the introduction of recurrent layers [90], while overfitting issues have been mitigated through the use of regularization layers [217]. Attention layers, a recent innovation, have further revolutionized model performance [238] by capturing important parts of the input data. However, the detailed construction of the neural networks is not the main focus of this manuscript.

## 1.2.2 Mathematical Definition

Despite significant advancements, the fundamental concept behind neural networks remains largely unchanged. A neural network can be viewed as a sequence of feature maps, denoted as  $z^l$ , for each layer  $l = 1, \dots, L$ , where  $L$  represents the total number of layers. The term *Deep Neural Networks* (DNN) will refer to neural networks with more than two layers. Mathematically, a layer can be defined as follows:

$$z^{l+1} = \sigma(W^l \times z^l + b^l). \quad (1.1)$$

Here,  $\sigma$  represents the activation function,  $W^l$  denotes the weight matrix, and  $b^l$  represents the bias vector for the  $l$ -th layer. The activation function, a non-linear function such as the Rectified Linear Unit (ReLU) function ( $ReLU(x) = \max(x, 0)$ ), introduces non-linearity to the network. The weight matrix and bias vector are the parameters of the neural network.

The classifier can be seen as the composition of  $L$  layers, each transforming the input to produce increasingly complex representations. Despite their simplicity, neural networks can learn highly complex functions. They are known as universal approximators [91, 39]. This means that neural networks can approximate any continuous function on a compact subset of  $\mathbb{R}^n$  with arbitrary precision, given sufficient training. The power of neural networks lies in their capability to learn a wide range of functions, including complex ones. They can even memorize the entire data set [279]. For instance, a DNN trained on a random dataset can perform 0 training errors. However, it will not generalize well. This inherent flexibility and approximation ability have contributed to the widespread success and popularity of neural networks in various domains.

In practice, the statements made in the previous discussions are partly true. If that were the case, a basic DNN with two layers could effectively learn any functions, and modern models

would either witness a decline or stagnation in the number of their parameters. However, in reality, the exponential growth of parameters in current models is explained by their better generalization. This ability is called the model capacity [237] and the number of parameters is one feature among others playing a crucial role. One way to measure the model capacity is through the Vapnik-Chervonenkis (VC) dimension, which quantifies the largest number of data points that a model can correctly categorize for all possible binary classifications. In simpler terms, the VC dimension represents the maximum number of points that a model can successfully classify into two classes, regardless of how those points are labeled or combined.

### 1.2.3 Model Notation

Let's assume an image classifier  $f$  represented as a function that maps an input  $\mathbf{x}$  to an output  $\mathbf{y}$  as follows:

$$f: \mathbf{x} \mapsto \mathbf{y}$$

$$[0, 1]^D \rightarrow \mathbb{R}^C.$$

where  $D$  represents the input dimension and  $C$  the number of classes. Considering the emphasis on ImageNet images in this manuscript, the input  $\mathbf{x}$  is indeed a vector of dimension  $D = 3 \times 224 \times 224$ , and the classification task involves  $C = 1000$  classes.

When images are displayed on a screen, they are represented as tensors of pixels, with each pixel comprising three values corresponding to the red, green, and blue (RGB) channels. Each channel is represented by an integer value ranging from 0 to 255. In this work, we consider the input of the classifier to be a tensor of continuous values in  $[0, 1]$ . To achieve this normalization, the pixel values are divided by 255. This normalization is a common practice in neural network applications [23]. For real images, therefore, only a small set of possible values is taken.

The classifier function  $f$  takes the normalized input tensor and produces a vector of *logits* in  $\mathbb{R}^C$ . To obtain probabilities for each class, the *SoftMax* function is commonly applied to the logits vector. The predicted class is determined by selecting the index corresponding to the highest probability (or highest logit value) in the output vector. The predicted class  $\text{cl}(x)$  is given by:

$$\text{cl}(x) = \arg \max_{i \in [1, C]} f_i(x), \quad (1.2)$$

where  $f_i(x)$  represents the  $i$ -th element of the vector  $f(x)$ .

## 1.3 Decision Boundary of Neural Network Classifiers

### 1.3.1 Definition

The concept of *decision boundary* is not exclusive to neural networks but is a fundamental concept in all classification models. When a classifier is trained on a dataset, it learns to divide the input space into distinct regions, known as *classification regions*. Each of these regions corresponds to a specific label, and the classifier assigns that label to any input falling in this region. The boundary that separates these regions, known as the *decision boundary* denoted  $\partial f$ , determines where the classifier changes its predictions. For instance, for the decision boundary between the class  $k$  and  $l$ , the logits vector will satisfy  $f_k(x) = f_l(x)$ . In other words, it is also the set of points where there exist at least a small perturbation  $\eta$  can cause the classifier to change labels:

$$\mathbf{x} \in \partial f, \text{cl}(\mathbf{x} + \eta) \neq \text{cl}(\mathbf{x} - \eta). \quad (1.3)$$

Once the model is trained, the decision boundary remains fixed and applies to all inputs. Understanding decision boundaries could help to train models more efficiently [4, 112, 86] or compare models [97, 189].

### 1.3.2 Visualization

The visualization of decision boundaries can naively be approached by randomly sampling two directions in the input space. However, it is important to consider the findings highlighted in [215, 169] and shown in Figure 1.4. According to these findings, the behavior of the model beyond the data manifold is almost uniform. In other words, the model produces similar outputs for inputs outside the data manifold, constituting a huge portion of the input space. DNNs only create decision boundaries in regions where they identify discriminative features in the training data. As a result, the visualization of decision boundaries should primarily concentrate on the data manifold and center the visualization around a specific image. This approach allows to focus on the representation of the decision boundaries in a relevant region.

Different methods have been proposed for visualizing decision boundaries in machine learning models. One-dimensional (1D) visualization approaches exist. The method described in [83] measures the distance to the boundary in multiple orthogonal random directions and [169] proposed to use directions on particular frequencies given by the *Discrete Cosine Transforms* (DCT) which is commonly used in image compression. This provides insights into the global structure of the margin to the decision boundary for a given point. However, two-dimensional

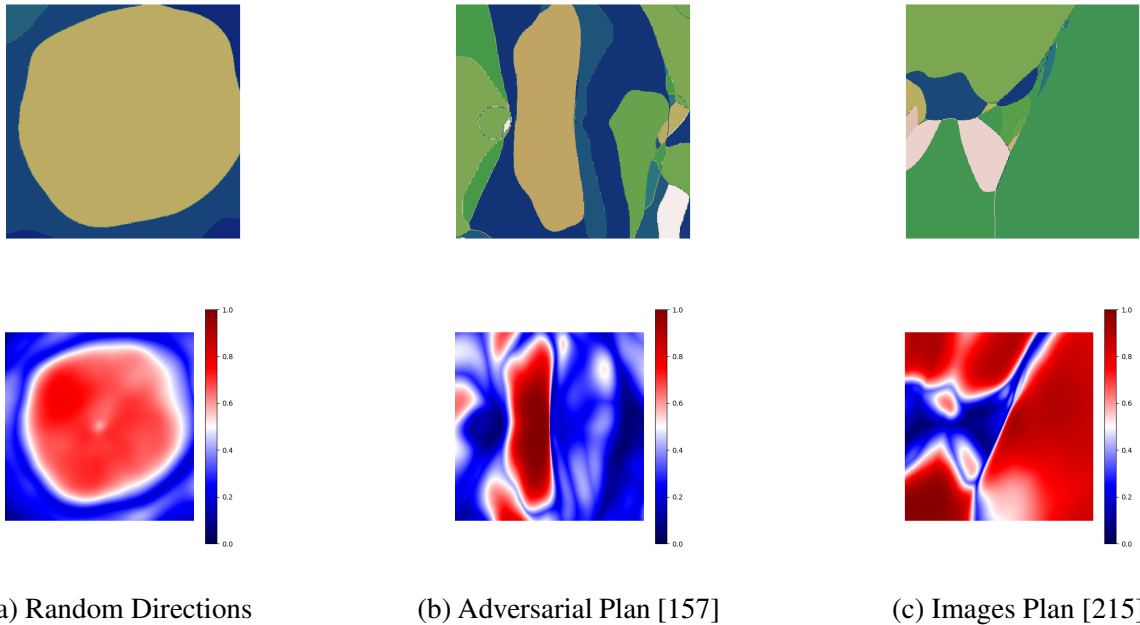


Figure 1.4 – Visualizations of the decision boundary centered on the image. The first row shows the predicted label in the input space. The second row shows the probability of the top-1 label in the input space.

(2D) representations are more commonly used.

In the context of understanding decision boundaries, [215] provides a suitable method inspired by the MixUp method. Because decision boundaries are created in the data manifold, they propose to visualize the decision boundary by considering a plan given by three images. It allows for a detailed examination of the decision boundary and its behavior in various input regions. A similar method is used by [271] to visualize the boundary between two classes.

In contrast, when the focus is on evaluating the local behavior of the model around a specific input, the approach used in [157] is more appropriate. This method considers the directions given by the closest point of a different label from the central image and a random direction. By doing so, it allows for an analysis of how the model reacts when the input is perturbed slightly. This visualization will be particularly relevant in this thesis, as it seeks to understand a concept explained in Chapter 3: the model’s robustness to adversarial examples.

### 1.3.3 Characteristics

Characterizing the decision boundary is a key focus in understanding the behavior of classifiers. Adversarial examples have been widely explored to probe the decision boundary [73, 108,



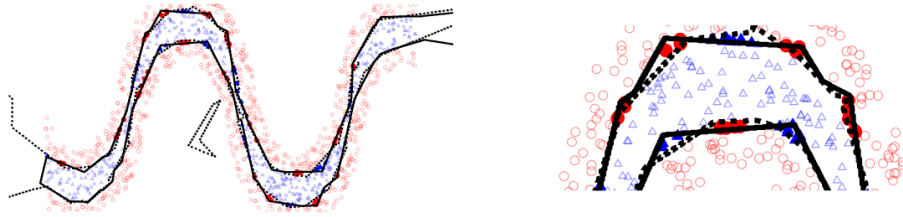


Figure 1.5 – Binary classification using a shallow model with 20 hidden units (solid line) and a deep model with two layers of 10 units each (dashed line) obtained in [154].

86]. But more exotic techniques have been employed to define and analyze it such as tropical geometry [4] and power diagrams [8] to define the partitioning of the input space.

These techniques provide insights into the structure and complexity of the decision boundary, enabling a deeper understanding of how the classifier separates different classes in the input space. This section sums up some important characteristics such as the curvature, the partitioning, and the geometric properties of the decision boundary.

### Succession of Hyperplanes

If we revisit the definition of a layer as discussed in Section 1.2.2, the linear component  $W^l \times x + b^l = \text{cst}$  corresponds to the equation on a hyperplane. From a local perspective, the model effectively identifies hyperplanes to create regions in the input space. The effectiveness of DNNs can be attributed to the abundance of hyperplanes they generate.

Let's consider a neural network with a single hidden layer, commonly called a shallow network. The input size of the layer is denoted as  $n$ , and the output size as  $m$ , denoting  $W^0 \in \mathbb{R}^{m \times n}$  and  $b^0 \in \mathbb{R}^m$ . This layer creates  $m$  hyperplanes to separate the input space. If we assume that DNNs utilize piece-wise linear activation functions, the configuration of these hyperplanes can yield  $\sum_{j=0}^n \binom{m}{j}$  regions [175, 154].

As we increase the number of hidden layers, the count of hyperplanes and regions grows exponentially. For example, a network with 3 hidden layers, each consisting of 100 neurons, produces 1 million hyperplanes and an astronomical number of regions. This phenomenon explains why DNNs can learn complex functions, as they leverage an extensive array of hyperplanes to partition the input space.

It also reveals why current neural networks are deep. With the same number of parameters, a DNN can create a more complex decision boundary than a shallow neural network. In Figure 1.5, the function found by the neural network of two layers of 10 units each is more complex

than the function found by the neural network of 20 units.

### **Curvature of the Decision Boundary**

Understanding the curvature of the decision boundary is crucial for gaining insights into the behavior of neural networks. Various measures and techniques have been proposed to analyze the curvature, including those based on tropical geometry [4], power diagrams [8], points close to the boundary [73] or analyze the features of the last layer [271, 157].

We could expect that the decision boundary would be highly complex [206] because neural networks are the resultant of a succession of millions of hyperplanes. However, this is not the case. Contrary to expectations, overparameterization of DNNs leads to simple decision boundaries. Instead, prior research consistently demonstrates that the decision boundary is predominantly flat [60, 157, 108, 206]. For natural images, the decision boundary is mostly flat in the majority of directions, with only a small subset of directions exhibiting significant curvature. Notably, the directions with significant curvature are data-dependent, indicating that they are shared among different images.

While the decision boundaries of well-trained deep models tend to be approximately linear, the complexity of the boundaries during the training process may initially appear. As the model becomes well trained, the decision boundaries gradually become linear [271]. This observation is echoed in the findings of [73], which establishes a correlation between boundary complexity and the generalizability of the model.

### **Stability**

Despite the vast number of parameters in DNNs, the decision boundary exhibits remarkable stability across different random initializations and diverse architectures. Prior works [215, 169] have shown that the decision boundary remains consistent and reproducible under these variations. However, the width of the model can impact the level of reproducibility, with wider models tending to exhibit higher consistency.

Interestingly, even within the same architecture family, such as Convolutional Neural Networks (CNNs), the decision boundary remains close across different architectures [215]. This implies that the overall separation of the classification regions is similar, regardless of important architectural differences, which is particularly impressive given the vast number of ways to distinctly separate the training data.

But, fortunately, this statement does not hold comparing models from different architecture

families. For models with the same accuracy, the classification regions and decision boundaries are distinct [215]. This observation suggests that achieving significant performance improvements, such as increased accuracy, in deep learning may require substantial changes to the architecture of new models rather than minor tweaks or adjustments.

## 1.4 The Vulnerabilities of Neural Network Classifiers

### 1.4.1 Definition of Security

The remarkable capabilities of neural networks also come with inherent vulnerabilities. In the context of machine learning security, vulnerabilities refer to weaknesses in models that can be exploited by malicious actors. These vulnerabilities can lead to various security risks. Some common vulnerabilities in machine learning security include the manipulation of the input for certain profit, the stealing of the model, or the retrieval of the training data that are supposed to be confidential. Machine learning security aims to safeguard these models and data, preventing unauthorized access, manipulation, or exploitation, and thereby ensuring the reliability and trustworthiness of the entire system.

To delve into the security aspects of machine learning, it is essential to define what we mean by *security*. While security is a broad concept with various interpretations, its application in the context of machine learning is relatively new. To grasp the essence of security, insights can be drawn from the field of cybersecurity. This section briefly introduces how these principles can be adapted and applied to machine learning systems. A more detailed explanation of these concepts in the scenario considered in this thesis will be provided in Section 1.4.3.

**CIA Triad.** In the 1970s, the concept of cybersecurity began to take shape, and the United States Department of Defense played a significant role in defining its fundamental principles [200]. They articulated computer security as the safeguarding of computer systems and information from harm, theft, and unauthorized use. To establish a comprehensive framework, they introduced the CIA triad, which comprises three essential principles: Confidentiality, Integrity, and Availability. Even though the CIA triad was conceived fifty years ago, well before the machine learning revolution, its relevance in today's cybersecurity landscape remains intact. The principles of confidentiality, integrity, and availability can be effectively applied to secure machine learning systems. Let's explore what each element of the CIA triad entails.

**Confidentiality** refers to the assurance that information is accessible only to authorized individuals or entities. It involves preventing unauthorized access, disclosure, or leakage of sensitive information. In the context of machine learning, confidentiality may involve protecting proprietary models, algorithms, or datasets. For example, a company developing a deep learning model for facial recognition may take measures to prevent the extraction of its training data in order to maintain its confidentiality.

**Integrity** focuses on the accuracy, consistency, and completeness of data and models. It ensures that information is trustworthy and has not been tampered with or modified in an unauthorized manner. For instance, applied to machine learning, it may verify the integrity of models, ensuring that they have not been altered during transmission or storage. A potential consequence of this lack of integrity is a significant decrease in accuracy, as each input may intentionally be modified to be misclassified by the model. Additionally, this integrity can be applied to the training and testing data, ensuring that the data remains unmodified.

**Availability** refers to ensuring that information and systems are accessible and usable when needed. When considering the unavailability of a model, the first thought might be about *Denial of Service* (DoS) attacks, which are common when models are accessed through an API. However, from a machine learning perspective, attackers can also optimize inputs to increase the energy consumption and latency of neural networks during testing [213, 31]. Once these crafted inputs are submitted to the model, it will take more time to process them, resulting in slow access or even a system crash.

### 1.4.2 A Realistic Scenario

Now that we have delved into the realm of security, it is evident that a vast array of vulnerabilities exists, each with its unique potential to disrupt the smallest grain that makes up any machine learning model. However, many of these vulnerabilities may seem unrealistic due to their assumptions. For example, some attacks assume that the attacker has full access to the training data or can modify the model at will. While these approaches serve their purpose of discovering and comprehending the vulnerabilities of DNNs, they do not align with real-world scenarios and the genuine threats that DNNs face.

In contrast, a growing body of research focuses on developing realistic attacks that avoid such assumptions and are more aligned with practical scenarios. Realistic security takes center stage in multiple papers, as they offer a better representation of how the security of DNNs

is truly threatened in the real world [218] such as self-driving cars [260] or medical applications [89, 50].

A realistic scenario in machine learning security does not exclusively pertain to physical attacks, although such attacks have been explored in research [57, 117, 257]. Many models are commonly deployed on the cloud through APIs or integrated into various applications, and they continue to be used in a real-world setting.

We have already taken a step towards the goal of examining security in a realistic scenario by centering our results on real-sized images from the ImageNet dataset (See Section 1.1). This section now provides a definition of a realistic scenario for the security of classifiers.

### Black-Box Setup

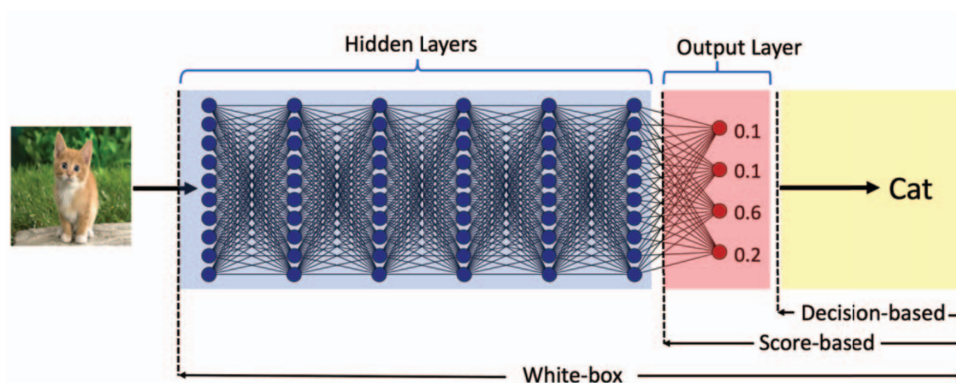


Figure 1.6 – Different setup illustration from [23].

The initial consideration when assessing the security is to determine what knowledge the attacker possesses about the model and what information is accessible from the model’s outputs. Commonly, three setups are examined, depicted in Figure 1.6. The level of security in DNNs is influenced by the extent of information available to the attacker in each setup:

- White-box setup: It is the most permissive and idealistic case. In this setup, the attacker has full knowledge of the model. He has access to the architecture and the weights. It leads to the most powerful attacks such as BP [280, 15], DeepFool [156], and CW [22] as far as adversarial examples are concerned, i.e. integrity of testing data.
- Grey-box setup: The attacker has partial knowledge of the model. In this scenario, the attacker possesses some information about the target model depending on the assumptions. For instance, the architecture and the weights may be partially known [276], the

model may be unknown but the training phase known [269, 182], or the model and its parameters may be known but the attacker ignores the defense mechanisms [158]. Grey-box setup lay between white-box and black-box setup, without having a common definition on the prior knowledge of the attacker about the attacked DNN.

- Black-box setup: The attacker has no access to the internal details of the target model. In other words, the attacker only has access to the input-output behavior of the model and does not possess knowledge about its architecture, parameters, or defenses applied to overcome vulnerabilities [174]. This setup is often divided into two sub-categories. First, the *score-based* scenario refers to the case where the attacker has access to the score resulting from the input. Second, the *decision-based* scenario refers to the case where the attacker only has access to the final decision of the model. It may also be referred to as *label-only* setup [289, 231] and *hard-label* setup [27, 24, 28].

A clarification is necessary. Currently, there is a substantial number of models trained on large datasets that are easily accessible. Companies might opt to use these pre-trained models as a foundation and fine-tune them to suit their specific tasks<sup>1 2</sup>. One often picks a model available on the shelf as a pre-trained model that one fine-tunes for a specific use case. However, the model is still unknown to the attacker, even if the architecture is publicly available.

**Among all scenarios considered in the literature, the decision-based black-box setup stands out as the most realistic and relevant for real-world applications concerning DNN security. Therefore, this thesis focuses on the decision-based black-box setup to address these concerns.**

### Limited Attacker Capabilities

In a black-box setup, the attacker’s ability to understand and exploit the model’s vulnerabilities is based on observing the model’s outputs by submitting inputs. By analyzing the outputs, the attacker tries to gain insights into the model’s behavior and identify potential weaknesses that can be exploited.

However, in a real-world scenario, there are limitations on the number of queries the attacker can make to the model. In our context, a query refers to the process of sending inputs to the model and recording the corresponding observations. These limitations are driven by factors

---

1. <https://blogs.nvidia.com/blog/2022/12/08/what-is-a-pretrained-ai-model/>  
2. <https://medium.com/the-web-tub/how-to-build-an-intelligent-chatbot-using-openai-pre-trained-models-with-javascript-207558b62b2f>

such as cost [23], the risk of detection [61, 128, 26], and being banned by the API hosting the black-box attacked model. For example, some vulnerabilities may require a significant number of queries, as much as 250,000 [283], to be successfully exploited. Each attack made by the attacker comes at a cost, for instance, 200\$ for a single attack<sup>3</sup>, which adds up to the overall attack expense.

This limitation forces the attacker to be strategic and selective in choosing the queries to maximize the chances of successful attacks while minimizing the number of queries made. It also highlights the importance of effective defense mechanisms that can detect and mitigate attacks even under limited query access.

### **No Access to the Training**

The access to the training data can vary in terms of its extent, ranging from partial to total access, and read or write capabilities. In certain cases, the attacker may even have access to the training phase itself, allowing them to exploit vulnerabilities that are typically associated with white-box attacks. However, it is important to note that the confidentiality of the training data is a crucial aspect of the success of the security of DNNs, and is generally in consequence well-protected. Infrastructure breaches are not considered in this thesis; we will consider these to be robust. It means that there is no access to the training data, and there is no leak. While the attacker may have access to publicly available data, such as images, text, or audio, he cannot ascertain whether this data has been used for training the model or not. Unless the attacker discovers this independently (during attacks for instance, such as *membership inference* attacks [84, 29]), he has no clue of whether the data has been part of the model's training set.

For this thesis, we have chosen to focus on realistic scenarios where the attacker does not have access to the training data [6, 174]. This constraint, although not always explicitly considered in the literature, clearly implements the more stringent and hardest setup for attackers in the security of DNN-embedding applications.

### **1.4.3 The Existing Vulnerabilities**

When companies invest in machine learning-based software and models, they access three valuable assets. The first and most apparent asset is the model itself. An accurate DNN holds significant industrial value due to the expertise required to train it and the computational resources needed to optimize its parameters during the training phase.

---

3. <https://cloud.google.com/vision/pricing?hl=fr>

The second asset is the training data. Their value stems from two reasons. Firstly, the quality and quantity of labeled data highly contribute to the model’s accuracy. Acquiring such data is challenging and often costly. Secondly, the training data often comprise sensitive or confidential information. Any unauthorized access or data breach could lead to legal and ethical consequences. Lastly, the infrastructure supporting the trained model is another valuable asset. Once the model is trained, it needs to be deployed in an efficient and scalable manner. The infrastructure hosting the model should be capable of handling a large volume of requests effectively. The infrastructure is out of the scope of this thesis; we prefer to focus on the vulnerabilities associated with the data at testing time.

The next section provides a brief overview of the different vulnerabilities of each asset that can arise in real-world scenarios, focusing on the CIA triad (See Section 1.4.1). Certain use cases, such as the confidentiality of the testing data, have been intentionally excluded from consideration due to their absence in recent legal cases.

## About Confidentiality

**Training Data.** We understand their confidentiality’s importance by looking at past legal cases involving the training data. In 2019, IBM sued Zillow for the use of more than 20,000 images from IBM’s Flickr account without permission from or compensation to IBM<sup>4</sup>. More recently, in 2023, Getty Images sued Stability AI for the use of more 12 million photographs from Getty Images’ collection without permission or compensation<sup>5</sup>.

Regarding the confidentiality of the training data, the attacker’s capability is measured by his ability to steal. These attacks based their findings on the model’s overfitting and its lack of generalization [279, 21, 212]. The high capacity of DNNs makes them memorize the training data (as random data in Section 1.2.2), which is exploited by the attacker to steal it.

In a realistic scenario, the attacker may use the model’s predictions to reconstruct the training data, known as *model inversion* [48, 107, 63] or *data extraction* attacks [21]. To reconstruct the data associated with a particular label of index  $i$ , the attacker aims to find an input  $\mathbf{x}$  that maximizes the model’s probability of predicting this label for that input [162]:

$$\hat{\mathbf{x}}_{\text{training}} = \arg \max_x f_i(x). \quad (1.4)$$

---

4. <https://www.nbcnews.com/tech/internet/facial-recognition-s-dirty-little-secret-millions-online-photos-scraped-n981921>

5. <https://news.bloomberglaw.com/ip-law/getty-images-sues-stability-ai-over-art-generator-ip-violations>





Figure 1.7 – An image obtained with a model inversion attack (left) and a corresponding training set image of the target model (right) from [63]. The attacker has access only to the person’s name and a facial recognition system.

Formulated this way, the attacker may find a local maximum outside the data manifold, which may lead to poor results. To mitigate this issue, the attacker can incorporate a *prior* on the input  $x$  to constrain the search space. For example, in the case of image inputs, Kahla *et al.* [107] use a style-GAN to generate an image that lies within the data manifold. As shown in Figure 1.7, this approach is applied to a facial recognition system, allowing the reconstruction of training data for a specific individual.

A simpler version is the *membership inference* attack [84, 29] where the attacker just determines whether a specific data point was part of the training dataset.

**Model.** The confidentiality of the model is determined by the attacker’s ability to steal the model and all its components, including the architecture, hyperparameters, and parameters. In a *model extraction* attack [276, 71, 102], the attacker aims to obtain a surrogate mimicking the target model. Both models exhibit the same behavior. During a model extraction attack, the attacker uses various techniques to gather information about the target model, querying it and analyzing its outputs. Once the attacker has successfully extracted the model, he can use it to make accurate predictions or to gain insights into the model’s decision-making process in order to prepare an evasion attack.

This aspect will be discussed in more detail in Chapter 2.

## About Integrity

**Training Data.** The integrity of training data encompasses various techniques aimed at either protecting the data or preventing the model from being trained effectively. This concept

covers a wide range of methods and approaches. One such technique is backdooring [10, 232], where the training data is deliberately poisoned with a specific signal. Once this signal is present in the input, the model's decision will always be biased toward a predetermined outcome. From a defensive standpoint, radioactive data [196] is an approach that allows us to determine if a particular dataset has been used to train a model.

While the integrity of training data may exist under realistic scenarios, certain challenges remain. For an attacker to modify the training data, he needs to have access to the data and be able to manipulate it. This level of access can be difficult to obtain in a real-world scenario. An attacker may poison a public dataset as well. This could be an effective strategy since models in production are typically based on models trained on large databases and fine-tuned for specific tasks [70, 56]. However, it still requires the attacker to specifically choose and manipulate the dataset among a vast number of existing datasets with the hope that this precise dataset will be leveraged for the training of the target system.

**Testing Data.** Evasion attacks, which focus on the integrity of the testing data through the creation of adversarial examples, have emerged as one of the most famous and extensively studied fields in the security of DNNs. Adversarial examples are crafted by introducing imperceptible perturbations to the input data, causing misclassification [146, 187, 221]. As depicted in Figure 1.8, the research on adversarial examples has garnered significant attention and numerous studies, making it a highly developed subject in DNN security. Between the start of this thesis in 2020 and its completion, the number of papers published in this field has quadrupled, from 1,500 to 6,000 papers per year. Understanding and addressing adversarial examples is crucial for enhancing the robustness and reliability of machine learning models in real-world applications. This subject will be discussed in more detail in Chapter 3.

## About Availability

The availability of the model refers to the attacker's ability to make the model unavailable, typically through a *Distributed Denial of Service* attack (DDoS attack). This type of attack is common in the field of cybersecurity and aims to overwhelm the model or the underlying infrastructure, causing it to become unresponsive or unavailable.

Traditional DDoS attacks, where a large number of bots overwhelm a system with requests to cause a crash, are out of the scope of this thesis. However, in the context of machine learning,

---

6. Source: <https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html>

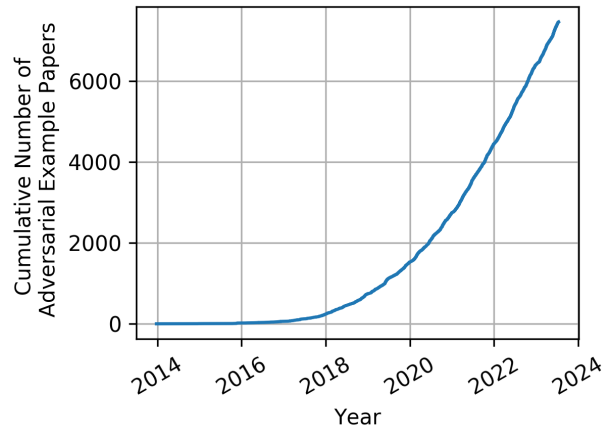


Figure 1.8 – Number of scientific papers investigating adversarial examples per year<sup>6</sup>.

there are inputs that can be crafted to exploit the resources of the model. An example of such attacks is *Sponge Examples* [213], where inputs are designed to increase the energy consumption and latency of DNNs at testing time. DNNs with a high number of parameters can be computationally expensive and time-consuming to infer. To improve efficiency, techniques like skipping zero operations are used to avoid unnecessary computations. To find *Sponge Examples*, [213] employs a genetic algorithm in a black-box setup to find inputs that significantly increase the model's latency. These crafted inputs activate more neurons than regular inputs.

Moreover, Antonio Cinà *et al.* [31] introduce *sponge poisoning* to demonstrate that these sponge examples can also be applied during training to increase the inference time for all inputs. Although it may be assumed that this attack is primarily related to the integrity of the training data, it turns out that the model can still be trained without any noticeable effects on its accuracy. The primary consequence of this attack is a significant increase in testing time. As a result, the model remains functionally effective but becomes somewhat inconvenient to use.

However, as far as we know, these works represent the only contributions that address availability concerns from a machine-learning perspective. This aspect mainly pertains to the infrastructure, which is not the primary focus of this thesis.

## 1.5 Conclusion

In this chapter, we delved into the world of machine learning and classification, emphasizing the significant role of neural networks in achieving remarkable performance in various domains over the last decade. Neural networks' capability to create effective decision boundaries in high-

dimensional data has propelled them to become the dominant tool in the field of classification.

However, despite their dominance, neural networks are not without weaknesses. These weaknesses become apparent when we delve into the inner workings of DNNs, such as their decision boundaries, which are exploited by malicious actors. In a realistic scenario, access to the model is constrained. Its weights and its outputs are locked in a black box. The only available means to comprehend the decision-making process of the neural network is by observing and analyzing the characteristics of the decision boundary itself. To better comprehend and exploit some vulnerabilities, a closer look will be done at their relation with the decision boundary. Understanding the decision boundary is instrumental in comprehending the origins of these vulnerabilities, providing valuable insights for the subsequent sections.

Upon introducing the vulnerabilities within DNNs in this chapter, one might be inclined to believe that these issues are novel and confined to DNNs. However, it is incorrect. Many of these vulnerabilities can trace their lineage back to previous studies that researchers have explored over the years, which might have occasionally been forgotten. Novel terminology may have hidden these historical roots, yet the underlying issues and potential solutions persist. An illustrative instance is the domain of watermarking and the corresponding attacks launched against watermarked content. Watermarking initially emerged in the 1990s, applied to images and audio. For instance, creators introduced minor perturbations to their content to assert intellectual property rights. Subsequently, detection mechanisms were employed to validate the presence of these watermarks. However, attacks were mounted on these mechanisms, termed as *oracle attacks*, which in essence constitute *evasion attacks*. Techniques emerged to eliminate these watermarks, rendering the content undetectable. Such methods involved manipulating the input through conventional image processing techniques or manipulating the decision boundary that segregates watermarked from non-watermarked content. Model fingerprinting presents another illustration of this historical continuum. It serves as a means of establishing model ownership and similar principles have been explored in the context of browser fingerprinting. Throughout this thesis, numerous references to the past will be drawn, emphasizing the connection between historical precedent and contemporary challenges.

The upcoming chapters focus on two major of these realistic vulnerabilities that pose significant threats to the security of machine learning models. Firstly, we address the issue of *confidentiality of the model*. In this context, each component of the model is susceptible to potential theft, ranging from the model's architecture to the actual parameter values. The extent of vulnerability depends on the number of queries the attacker can make to the model. Secondly, we explore the vulnerability related to the *integrity of the testing data* through *adversarial ex-*

*amples*. Their connection to the decision boundary of the model is inspected and an overview of various attack and defense mechanisms is introduced.

Although not covered in this thesis, it is crucial to acknowledge the significance of the confidentiality of the training data in a realistic scenario.

# CONFIDENTIALITY OF MACHINE LEARNING MODELS

---

## 2.1 Introduction

When we think about the biggest tech companies, names like Amazon, Google, Facebook and Alibaba, come to mind. They are known for their efficient machine learning models powering various services such as Google's search engine, Amazon's recommendation system, and Facebook's face recognition system, among others. Achieving top-level efficiency in machine learning models is a coveted goal for many companies, but it is not an easy task. It requires significant resources, time, and expertise, which not all companies possess.

Accordingly, some companies might be tempted to resort to reprehensible practices, such as stealing a competitor's model or simply obtaining information about it. Such actions may be seen as a way to gain a competitive advantage without investing the huge required efforts in research and development or simply as a means of reducing cost if the target system is a paid service [166]. However, it is important to emphasize that such practices are illegal and unethical.

Stealing a model not only constitutes a direct attack on the model's confidentiality but can also serve as an indirect means of compromising the model's inference in a black-box scenario. By constructing a similar model, the attacker effectively gains white-box access to the model's behavior even if the attacker initially had a total ignorance about the model's internal workings. This raises concerns about the integrity of model inference. A more in-depth exploration of this aspect will be made in Chapter 3.

Despite being illegal, the act of stealing a model or any valuable information is still possible. In this section, we explore various methods that attackers might employ to steal a model or its parameters, along with the different strategies available to protect against such threats.

## 2.2 Attacks

In a practical scenario, a potential attacker might endeavor to duplicate a target model or its attributes through querying it, a practice known as *model extraction*, *model stealing* or *attributes extraction*. Attackers might have their sights set on specific attributes of the model, such as its architecture, parameters, or hyperparameters. The level of difficulty in stealing each of these attributes varies, which is typically measured by the number of queries required by the attacker. The primary objective behind such endeavors is to gain a deep understanding of the model, facilitating its precise recreation.

As we delve into this, it will become evident that precisely extracting all model attributes can be a hard task, possibly even unattainable within a realistic context for large models. An alternative approach to undermining model confidentiality involves replicating its behavior independently of its internal working. Here, the attacker’s aim is not to replicate the model per se but rather to acquire a model that reproduces the same decision boundaries. Both of these attack types are expounded upon in this section.

### 2.2.1 Extract Exact Model Attributes

While many attacks under a realistic scenario primarily target the model’s output to compromise its confidentiality, *side-channel attacks* exist. These attacks exploit information that is not directly related to the model’s output. For instance, side-channel attacks can leverage extra information, such as the runtime at inference [53] or the power consumption [11] during the model’s execution. They are mostly used to steal a few properties of a model such as the architecture and can not be used to steal the model’s behavior. However, some of these attacks may not be practical in a real-world scenario, as they require access to the model’s execution environment. They will be shortly debate.

**Architecture.** Inferring the model architecture involves determining the type and number of layers, and any specific characteristics of these layers, such as the kernel size of a convolutional layer. In a hard-label setting, the attacker may need to assume some properties. For instance, the authors in [164] uses a meta-model. The meta-model is a classifier of classifiers that can estimate hyperparameters. It works by taking outputs for a given set of inputs and estimating the corresponding hyperparameters. The meta-model is also responsible for selecting the most informative inputs. However, the meta-model must know the global structure of the victim model. For example, if the victim model uses convolutional layers, the meta-model needs to be aware

of it.

Side-channel attacks have also been explored to infer the model architecture. For instance, in the work by Duddu *et al.* [53], the depth of the network is inferred by exploiting the inference time. Based on this information and a list of architecture candidates, they can eliminate models that do not fit. Another approach by Batina *et al.* [11] uses timing, power consumption, and electromagnetic emanations to retrieve the entire architecture and even the model’s parameters. For instance, they infer which activation function is used by measuring the minimum, maximum, and mean computation time.

**This thesis investigates this attack scenario under the assumption that the attacker possesses a list of models. Our contribution, FBI [147], which is developed in Chapter 7, calculates a distance metric between the black-box model and each model within its list. This approach allows us to infer the architecture of a known model and even detect any modifications made to the model in comparison to what the attacker possesses.**

**Parameters.** The attacker queries the black box and extracts the exact parameters of the model. The architecture of the model is usually known in this scenario. However, if the architecture is unknown, the attacker may need to perform a preliminary step: architecture extraction. Although some attempts have been successful, it remains a first challenging step [153, 20].

One approach proposed in [20] relies on the fact that ReLU neural networks are piecewise linear functions, and querying at critical points reveals information about the model parameters. However, in all these methods, when an image is submitted, the gradient for this image is obtained. Even in this case, [20] successfully extracted a neural network of 100,000 parameters trained on the MNIST dataset with more than three million queries.

Considering that modern models often have millions or even tens of millions of parameters, the task of stealing the exact parameters of such models seems unfeasible in a hard-label black-box setup.

**Hyperparameters.** Part of machine learning training relies on hyperparameters, which are configuration settings determined before the training process. Unlike model parameters that are learned during training, hyperparameters are predefined by humans. For instance, they include key settings like the optimizer, the learning rate, and the batch size. These hyperparameters



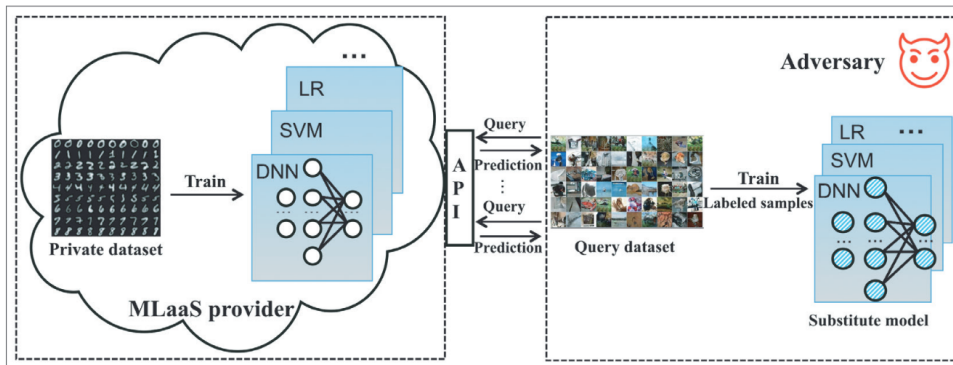


Figure 2.1 – Overview of a Model Extraction Attack.

significantly influence the model’s training dynamics and performance, such as the learning rate [228, 241].

A few works have been successful on images. For instance, MMA [164] uses a meta-model as described above. More than architecture it can predict any hyperparameters that significantly impact the model’s behavior such as the learning rate and the batch size.

To train the meta-model, multiple architectures are tested during training. For instance, it required testing 10,000 CNN architectures on MNIST to achieve 80% correct hyperparameters for a single model, and this process took 40 days to complete.

## 2.2.2 Approximate behavior

The primary objective of extracting the model from the black box is to replicate the behavior of the target model. The model constructed by the attacker is commonly referred to as the *substitute model*, *clone model*, or *surrogate model*. The process of building such a model shares similarities with other fields such as distillation, and some methods may draw inspiration from these related areas. These approaches leverage the available information to construct a substitute model that approximates another model’s behavior. For example, the distillation technique involves transferring the knowledge of a complex teacher model to a simpler student model [87]. This approach aims to create a smaller model that achieves comparable performance on the same task, potentially resulting in energy savings. However, in these scenarios, the teacher and student models are both under the control of the same entity, which possesses complete knowledge of all aspects (architectures, training data, logits, etc.). In the context of model extraction, the attacker lacks access to any such information or resources.

The efficiency of a model extraction attack is assessed through the surrogate model’s top-1

accuracy [167, 233, 102, 253] on the victim’s test dataset. While the surrogate model might not achieve the same performance as the victim model due to its inability to create new knowledge, the objective is to minimize the drop in accuracy as much as possible.

**Data is the Key.** Illustrated in Figure 2.1, the unique information accessible to the attacker is the data submitted, which serves as the foundation for constructing their training dataset. Let’s imagine that the attacker enjoys unrestricted query access. In such a scenario, the attacker could adopt a greedy strategy by querying every possible image in the input space, constructing a training set, and eventually creating a model that impeccably imitates the victim model’s behavior in that input domain. However, this notion is far from being realistic. The center of most research articles lies in addressing the question: "How can I intelligently select (or generate) the input data to submit to the black-box system in a way that I explore the input space and yield a trained model that effectively emulates the victim model’s behavior?"

Now, let’s consider the data acquisition aspect. Having complete access to the original training data is rarely assumed. Instead, some studies grant access to a subset of the data [106, 174], samples that resemble the original training data distribution [35], a distinct dataset but publicly available [167, 253], or a huge number of synthetic data generated using Generative Adversarial Networks (GANs) designed to align with the underlying data manifold [231, 202, 248]. Among these approaches, the last two are the most realistic since they eliminate the need for access to or understanding the original training data. The subsequent sections will delve into these two approaches in greater detail.

**Data from Publicly Available Dataset.** Utilizing publicly available datasets is a common strategy, although adjustments might be necessary to be accepted by the target model [253, 167] such as the input size. However, random sampling from this dataset may not provide sufficient information. This approach could also be easily detected since the submitted images might fall outside the data manifold [167].

In the work of [167], the approach begins with a pre-trained model on a publicly available dataset, which has labels that are not correlated with the labels of the victim model. They employ a reinforcement learning strategy to efficiently select queries. These queries are chosen to encourage images where the victim model displays high confidence, images to obtain a balanced training set for the attacker’s model, or images that cause disagreement between the substitute model and the victim model. This results in a newly created model with good accuracy on the victim model’s test set. However, this approach necessitates around 60,000 queries

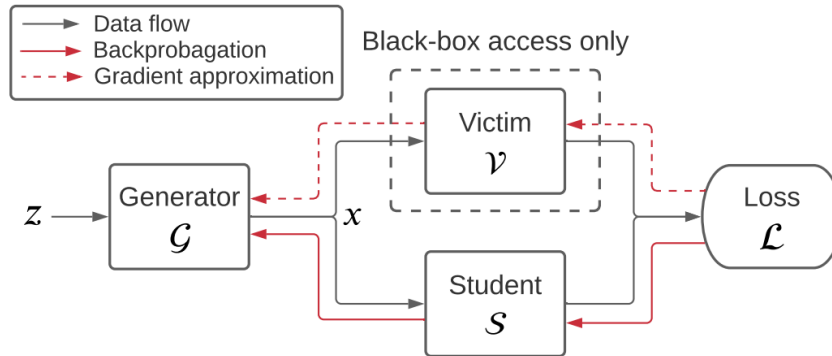


Figure 2.2 – Overview of the Data-Free Model Extraction Approach [233].

for dataset copying, such as Caltech256 with access to ILSVRC, an image dataset with 4 times more classes and 40 times more images than Caltech256.

Another approach under hard-label setup leverages the attention maps generated by the substitute model to selectively erase important portions of samples [253]. If both the substitute and target models focus on the same regions, removing the area of interest from a input would likely lead to a change in prediction by the victim model. Conversely, if the attention differs between the two models, the erased sample is integrated into the training set with a label chosen by the victim model.

**Synthetic Data.** Another strategy involves the utilization of GANs to generate data. In particular, we focus on Data-Free GANs [233, 202] as they operate without requiring access to actual data, making them particularly relevant in realistic scenarios.

In a GAN framework, there are two main components. The generator function is a forger, crafting synthetic data that do not come from real sources. On the other hand, the discriminator acts as an investigator, aiming to differentiate real data from fake data. These two components engage in a competitive process, intending to enhance each other's performance. In the context of model extraction, the student model takes on the role of the surrogate, while the generator is responsible for creating informative data that facilitates the efficient replication of the victim model.

DFME [233] uses the advancements in data-free knowledge distillation in a context where the attacker has access to the victim's output probabilities and depicted in Figure 2.2. The discriminator's role is to minimize disagreement between the surrogate and victim models, while the generator's role is to maximize these discrepancies. Although operating in a black-box environment without access to gradients, Zero-Order Optimization [25, 235] is employed to estimate

the victim’s gradients. The successful application of this technique led to the extraction of a model achieving 92% of the victim model’s accuracy with 20 million queries on the CIFAR-10 dataset, as demonstrated in Figure 2.2.

In a hard-label setup, DFMS-HL [202] uses a GAN to generate images close to the victim dataset but suppose access to a training dataset to pretrain his GAN. In this setting, the attacker does not have access to gradients, so the clone model is trained using the cross-entropy loss with the output of the victim model as the ground truth. These techniques and methods help the attacker generate or select informative data for the model extraction process, even when full access to the original training data is not available to align the attention between the substitute and the victim model.

**A Huge Cost.** To decrease the number of queries required to copy the model, attackers adopt strategies to select the most informative samples or make samples more informative. For example, they may pick samples close to the decision boundary [231] or even generate adversarial samples on the boundary [272]. Fortunately, extracting models is still highly expensive.

Notably, model stealing techniques have been tested on small datasets like MNIST, CIFAR-10, or CIFAR-100, providing an estimate of the cost involved on real-world models for which they use larger data. In terms of queries, [253] used 30,000 queries to create a substitute model on CIFAR-10. However, this number grows exponentially as the model’s size increases. For instance, DFMS-HL [202] requires 8 million queries to clone a model on CIFAR-100, which is a considerable improvement compared to ZSDB3KD [255], which requires 4 billion queries.

Another factor in the number of queries is the output available from the attacker. While most methods consider access to probabilities or logits [166, 231, 109, 35, 167], the hard-label setup significantly impacts the number of queries required [166, 53]. To illustrate this point, Dast [289] needed an enormous number of 20,000,000 queries on Microsoft Azure to attack a MNIST model in a hard-label setup. However, in the probability-only setup for the same scenario, *only* 2,000,000 queries were required. We understand why hard-label setup has been poorly studied.

## 2.3 Defenses

Defenses can be classified into two categories: *reactive* and *proactive*. Reactive methods are designed to detect ongoing attacks or if an attack has already occurred, and they typically rely on analyzing the model’s output. Proactive methods, on the other hand, aim to mitigate potential

attacks before they happen by modifying various aspects of the model, such as its architecture, learned parameters, or decision boundary.

### 2.3.1 Reactive Defenses

Reactive defenses aim to detect ongoing attacks, and one of the most commonly used approaches is ownership verification. It is a process used to demonstrate the ownership of a stolen model and is often accomplished through *model fingerprinting* or *watermarking*. Its primary purpose is to provide evidence of past attacks on the model. The defender suspects that the model has been stolen and wants to prove it. To resist model stealing, ownership proof must endure throughout the attack and appear in the substitute model as well.

**Watermarking.** Watermarking achieves ownership verification by incorporating hidden information into the model that only the legitimate owner can extract. A watermark detector is a two-class classifier checking for the presence or absence of the mark in an image. One approach to achieve this is by embedding secret backdoors during the model’s training. While watermarking has been a common practice for decades in the field of image processing [220, 34, 179, 256], it has just recently been incepted into the machine learning domain.

Uchida *et al.* [236] first proposed to watermark a DNN by embedding it into the weights and biases of the model. Quickly after this initial proposal, works instead focused on a black box model, where the presence of a watermark can be assessed by the owner from remote interaction with the suspected DNN. In [120], authors insert information by altering the decision boundaries through finetuning. In [2], authors also retrain the model to obtain the wrong labels for a so-called *trigger set* of inputs, that constitutes the watermark. Please refer to [135] for a complete overview of the domain.

The watermark itself can become a target for attacks. This community called oracle attacks what we now call black box attacks: the attacker has the secret-keyed detector in hand as a black-sealed box, and calls it iteratively to either estimate the secret key or remove the watermark from protected images. As the watermark serves as proof of ownership for a model, an attacker may attempt to remove or copy it. For instance, in the case of BNSA [34], the goal is to deceive the detector so that it identifies the input as watermarked, while it has been created by another party. By doing so, the attacker can falsely claim ownership of the model or, in this case, an image. While most of the research on this topic has been focused on images, there is no reason why it cannot be extended to other domains, such as machine learning models.

**Model Fingerprinting.** Fingerprinting is another technique used to uniquely identify a model. The idea is to identify a distinctive model property, but unlike watermarking, the owner does not actively embed this property, as it is an inherent characteristic of the model. It commonly provides a similarity score between two models by looking at similar decisions in submitted queries, such as cosine similarity score [134].

Most fingerprinting methods start with a small collection of benign inputs (except [247] starting from random noise images) and forges examples that lie close to the decision boundaries [22], which are considered as the signatures of a model. These examples are called adversarial and will be detailed in the next chapter.

Two trends are connected to two applications. The first one is linked to the integrity of the model, where the goal is to verify if a model has not been altered. The goal is to sense a *fragile fingerprint*. Any modification of a model is detectable because it changes the fingerprint. In that light, methods in [85, 119] create sensitive examples that are triggered only by modifications of the vanilla model.

The second application is called *robust fingerprint*, which refers to a fingerprint that remains valid for any modification of the model while being more specific to the victim model. IP-Guard [19] and Frontier Stitching [120] were the pioneers in providing model fingerprinting by creating examples close to the decision boundary to define the fingerprint. The goal is to frame the decision boundary of a model and detect any variation in it. Subsequent works based on these ideas continue to use adversarial examples [178, 140, 284, 172]. Lukas *et al.* [140] introduced the concept of conferrable examples, which are adversarial examples that transfer only to variations of the targeted model. AFA [284] and TAFA [172] use dropout as a cost-effective surrogate of model variants when generating adversarial examples for fingerprinting purposes.

**All the methods discussed so far use adversarial examples for creating a model's fingerprint. However, generating adversarial examples with specific properties can be a challenging task. In addressing this challenge, this thesis presents an innovative approach called FBI [147], which is introduced in Chapter 7. It stands out as the only method that utilizes unmodified images. It estimates the mutual information to measure the statistical dependence of two models' predictions.**

### 2.3.2 Proactive defenses

Proactive defenses are primarily aimed at reducing the effectiveness of attacks, making them more difficult to execute. While these defenses may not completely make attacks impossible, they increase the number of queries and time required to successfully steal the model. Proactive defenses encompass a wide range of methods such as:

- Hard-label output: It is a simple yet highly effective approach. As discussed in Section 2.2.2, by only revealing the predicted label without any additional information, it significantly increases the number of queries required for efficient model extraction. This increase makes it considerably time-consuming for attackers to steal a model;
- Input perturbation: To enhance the difficulty of model stealing, one effective approach is to add perturbations to the input, which can obscure the model’s true attention and make the extraction process more challenging [252]. For instance, Guiga *et al.* [74] use noise on unimportant pixels identified by the Gradient-weighted Class Activation Mapping (Grad-CAM) method to protect image models.
- Output perturbation: It is another technique that introduces noise to the output of the model. While many methods in this category add noise to logits or probabilities [168], it does not have an impact on decision-based black-box methods. Implementing output perturbation for decision-based black-box attacks in a way that does not significantly affect accuracy requires a clever approach. For instance, Kariyappa *et al.* [110] only randomize the model’s output when the query lies outside the original data distribution. This ensures that the output remains reliable within the known data distribution while adding uncertainty for queries that fall outside it.

## 2.4 Conclusion

This chapter provided a brief overview of model confidentiality. While all components of a model may be stolen, the majority of the literature focuses on models that output probabilities. However, even in such cases, most attacks are expensive in terms of queries and time. Additionally, side-channel attacks are another option but require hardware access.

It appears challenging to believe that stealing a model is possible in a realistic scenario without any prior knowledge about it. However, it is worth noting that part of the knowledge may already be available. Many models used in production are based on models trained on large publicly available databases, which are then fine-tuned for specific tasks. The real accessible

challenge may lie in identifying the correct architecture among all the existing ones.





# ADVERSARIAL EXAMPLES

---

## 3.1 Introduction

### 3.1.1 Illustration

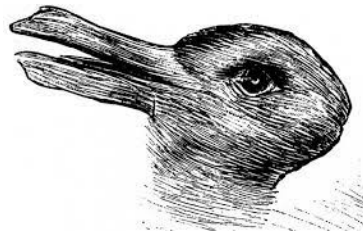


Figure 3.1 – Adversarial example for human perception.

As discussed in Section 1.2.1, neural networks have demonstrated impressive performance on various classification tasks, surpassing human accuracy in some cases [82]. However, these high-performing networks are also highly sensitive to *adversarial examples*.

The existence of adversarial examples, introduced by Szegedy [221] in 2013, is a counter-intuitive feature of neural networks. Despite the high accuracy of DNNs on classification tasks, a small imperceptible *perturbation* added to an image can cause a change in the predicted label. These so-called *adversarial examples* can be found in various types of data, including images [262, 23], audio [165, 222] and text [274, 159, 286]. They pose a security issue for the deployment of neural networks in real-world applications, particularly in safety-critical domains such as autonomous driving [260] or medical diagnosis [89, 50].

Misclassifying an input with a perceptible perturbation is indeed a known concept. For instance, optical illusions are a well-known example of this phenomenon and can be considered adversarial examples of human perception. In Figure 3.1, the interpretation of the image changes. It plays with the human perception of juggling between a rabbit and a duck. The characteristic of adversarial examples is the same but for DNNs. They are imperceptible to the

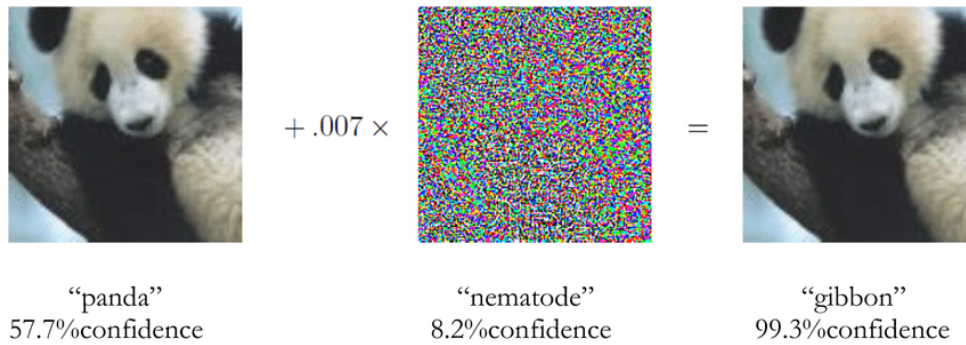


Figure 3.2 – Adversarial example of a panda misclassified as a gibbon [221].

human eye while causing misclassification by DNNs. This is demonstrated by the famous example of a panda image being misclassified as a gibbon, as shown in Figure 3.2. Notably, DNNs are generally robust to random perturbations but vulnerable to carefully crafted adversarial perturbations [58, 221].

With the wide range of models imagined, different DNNs exhibit varying degrees of sensitivity to adversarial examples. The ability of a model to withstand adversarial examples is referred to as its *robustness*. Investigating and understanding this concept can contribute to enhancing model robustness. Before delving further, let us explore the mathematical definition of adversarial examples.

### 3.1.2 Definition

An adversarial example  $\mathbf{x}_{\text{adv}}$  can be defined as a perturbed input sample that is designed to deceive a machine learning model  $f$ . An adversarial perturbation  $\delta$  is added to the original input sample  $\mathbf{x}_o$  of label  $y$  to cause misclassification:

$$\mathbf{x}_{\text{adv}} = \mathbf{x}_o + \delta \quad \text{s.t.} \quad \text{cl}(\mathbf{x}_{\text{adv}}) \neq y \quad (3.1)$$

Adding a large random noise to the image may work but it is not very informative or interesting. The goal is to find the optimal adversarial  $\mathbf{x}_{\text{adv}}^*$  with the minimal perturbation  $\delta^*$  leading to misclassification.

The perturbation is typically constrained within a certain norm or distance metric such as the  $\ell_p$  norm. Three choices of the  $\ell_p$  norms are retrieved in the literature. The  $\ell_0$  norm measures the number of pixels modified. The  $\ell_2$  norm measures the Euclidean distance between the original image and the adversarial example. The  $\ell_\infty$  norm measures the maximum difference between

the original image and the adversarial example.

This thesis focuses on the  $\ell_2$  norm which is the most common norm used in the adversarial literature as far as images are concerned [23, 156]. It corresponds to the straight-line distance between two points, these two points representing images in a high dimensional space. This geometric interpretation aligns well with human perception of image similarity. The term of *distortion*, denoted  $\text{dist}$ , is used to describe the magnitude of the perturbation. The distortion is defined as the root mean square error:

$$\text{dist}(\mathbf{x}_{\text{adv}}, \mathbf{x}_o) := \|\mathbf{x}_{\text{adv}} - \mathbf{x}_o\|_2 / \sqrt{D}, \quad (3.2)$$

where  $D$  is the number of pixels.

This is easily interpretable: if  $\mathbf{x}_{\text{adv},i} = \mathbf{x}_{o,i} \pm \epsilon$ , for all pixel  $i$  in the integer interval  $\llbracket 1, D \rrbracket$ , then  $\text{dist}(\mathbf{x}_{\text{adv}}, \mathbf{x}_o) = \epsilon$ . It is easily translated into a PNSR as image processing professionals do:  $\text{PSNR} = 48.13 - 20 \log_{10}(\text{dist}(\mathbf{x}_{\text{adv}}, \mathbf{x}_o))$  dB. Adversarial perturbations usually spread all over the image and have a small amplitude like in invisible watermarking. This is a case where measures based on  $\ell_2$  norm remain good indicators of the quality. A perceptual similarity is obviously better, but more complex and less interpretable.

The goal is to find the optimal adversarial  $\mathbf{x}_{\text{adv}}^*$ :

$$\mathbf{x}_{\text{adv}}^* = \arg \min_{\mathbf{x}_{\text{adv}}} \text{dist}(\mathbf{x}_{\text{adv}}, \mathbf{x}_o). \quad (3.3)$$

This definition holds for *untargeted* attacks, where the goal is to cause misclassification, whatever the arrival label. In the case of *targeted* attacks, the goal is to cause misclassification to a specific class  $y_t$ :

$$\mathbf{x}_{\text{adv}}^* = \arg \min_{\mathbf{x}_{\text{adv}} \in \{\mathbb{R}^D : \text{cl}(\mathbf{x}_{\text{adv}}) = y_t\}} \text{dist}(\mathbf{x}_{\text{adv}}, \mathbf{x}_o). \quad (3.4)$$

### 3.1.3 Quality of Evasion Attacks

To assess an evasion attack, a set of  $n$  images is considered. These images are represented as  $\mathcal{X} = \{\mathbf{x}_o^1, \mathbf{x}_o^2, \dots, \mathbf{x}_o^n\}$  with corresponding labels  $\{\mathbf{y}^1, \mathbf{y}^2, \dots, \mathbf{y}^n\}$ . The attack generates  $n$  adversarial examples denoted as  $\{\mathbf{x}_{\text{adv}}^1, \mathbf{x}_{\text{adv}}^2, \dots, \mathbf{x}_{\text{adv}}^n\}$ .

Subsequently, various metrics are employed to gauge the effectiveness of the attack. The most commonly used metrics include the *attack success rate* (ASR), the *mean distortion* (MD), and the *operating characteristic* (OC).

**Attack Success Rate (ASR) under Distortion Constraint.** When conducting an evasion attack, the goal is to achieve a certain level of success. Typically, this is quantified by imposing a specific distortion constraint often denoted as  $\epsilon$ . The ASR is a metric that calculates the probability of adversarial examples being misclassified by the target model  $f$  while staying under the threshold  $\epsilon$ . This can be formulated as:

$$\text{ASR}(\epsilon) := \mathbb{P}(\text{dist}(\mathbf{x}_{\text{adv}}, \mathbf{x}_o) < \epsilon). \quad (3.5)$$

In other words,  $\text{ASR}(\epsilon)$  represents the fraction of adversarial examples successfully misclassified within a given distortion budget  $\epsilon$ . This is evaluated by using the Iverson bracket over the set of adversarial examples:

$$\text{ASR}(\epsilon) := n^{-1} \sum_{i=1}^n [\text{dist}(\mathbf{x}_{\text{adv}}^i, \mathbf{x}_o^i) < \epsilon]. \quad (3.6)$$

Usually, the threshold  $\epsilon$  is predefined, and the ASR is often referred to simply as ASR. This metric is commonly applied in the context of white-box attacks [280, 142, 22, 156] with relatively modest thresholds, frequently set as  $\ell_\infty$  lower than  $\epsilon = 8/255$  if  $\mathbf{x} \in \llbracket 0, 1 \rrbracket^D$  (if  $\mathbf{x} \in \llbracket 0, 255 \rrbracket^D$ , it means  $\epsilon = 8$ ). When dealing with black-box setups, this measure is also employed, particularly in cases involving transferable attacks [265, 52, 244].

**Mean Distortion under Success Constraint.** The concept of Mean Distortion pertains to the average magnitude of perturbations required to induce misclassification through an attack. For this score, the ASR is usually supposed to be 100%. This metric is calculated as follows:

$$\overline{\text{dist}} := n^{-1} \sum_{i=1}^n \text{dist}(\mathbf{x}_{\text{adv}}^i, \mathbf{x}_o^i). \quad (3.7)$$

**Operating Characteristic.** The operating characteristic provides insights to gain a comprehensive understanding of the interplay between the attack and the model. This characteristic establishes a relationship between the distortion  $D$  and the probability of attack success  $\text{ASR}(D)$ , defined by the equation:

$$\text{ASR}(D) := \mathbb{P}(\text{dist}(\mathbf{x}_{\text{adv}}, \mathbf{x}_o) < D). \quad (3.8)$$

It may also be seen as the cumulative distribution function of the distortion. Compared to the ASR and the mean distortion, the operating characteristic is a more general metric that does not

require a predefined threshold on the success or the distortion. It is worth noting that, unlike the prevalent practice of measuring ASR for a specific distortion level within the literature [38], the present thesis considers the entire spectrum of  $D$  values. Empirically, this operating characteristic is computed over the set of adversarial examples using the following formula:

$$\text{ASR}(D) := n^{-1} \sum_i^n [\text{dist}(\mathbf{x}_{\text{adv}}^i, \mathbf{x}_o^i) < D]. \quad (3.9)$$

## 3.2 Existence of Adversarial Examples

Understanding adversarial examples is a critical challenge for promoting the widespread adoption of deep learning in security-critical applications. However, the scientific community remains divided on this subject because of the high dimensionality in which DNNs work. Several theories have been put forth to explain the existence of adversarial examples. We focus on one particular hypothesis gaining acceptance and extensively utilized in the literature to develop new attack and defense strategies. Readers are invited to read the survey of [77] for a more complete overview of the different theories.

**Initial Idea.** When adversarial examples were first introduced, the initial hypothesis by Szegedy *et al.* [221] regarding their existence was based on the idea of low-probability *pockets* in the classification regions. These *pockets* are difficult to find, which helps explain why models are robust to random noise. Indeed, it is observed that the neighborhood of adversarial examples possesses unique characteristics compared to benign images. Adversarial examples are found to be situated close to decision boundaries in all directions and are surrounded by the class of their original image [83].

Initially, Szegedy *et al.* [221] suggested that these pockets are a result of the high non-linearity present in deep networks. However, further investigations into adversarial examples within the data manifold have led to additional explanations [77].

**Data Manifold Hypothesis.** When we perceive images with our human vision, we are exposed to a vast array of shapes, colors, and objects, leading us to believe that the number of possible natural images is infinite. However, as mentioned in Section 1.3, this is not the case. The high-dimensional data, which serves as input to the model [101], actually lies in a lower-dimensional smooth manifold. The majority of the space  $\llbracket 0, 1 \rrbracket^D$  is out of this data manifold.

Adversarial examples are predominantly found in this empty space, outside the data manifold [216, 199, 227]. They exploit the fact that the model is not adequately trained in this region [204], resulting in the creation of inaccurate decision boundaries. Essentially, the vulnerabilities in the model's behavior arise from the training process itself, leaving ambiguous or "grey" areas in the input space, close to the data manifold, which adversarial examples can exploit to mislead the model's predictions.

**Piece of Evidence.** Based on this line of reasoning, it can be deduced that the availability of more diverse data for training contributes to the model's ability to effectively generalize and develop robustness against adversarial examples. This idea is substantiated by research findings [68, 203], which have demonstrated that when the model is trained on diverse and abundant data, its robustness is significantly improved.

Non-linearities within the model's architecture play a crucial role in enabling effective learning from limited training data [203, 239]. However, this characteristic can also be seen as a form of overfitting, where the model learns to perform exceptionally well on the training data but may struggle with unseen or perturbed inputs.

On the other hand, models that exhibit increased robustness against adversarial examples tend to display a low curvature [157]. These characteristics indicate the model's ability to generalize well and avoid overfitting the training data. By maintaining a smoother decision boundary, the model becomes less susceptible to being misled by small perturbations, making it more resilient to adversarial examples.

### 3.3 Properties of Adversarial Examples

Adversarial perturbations can be intentionally designed with specific properties to evaluate the robustness of a model or to expose potential vulnerabilities. These properties serve as important indicators. In this section, we discuss and explore the key properties associated with adversarial perturbation which are summarized in Figure 3.3.

**Universal Adversarial Perturbations (UAPs)** are input-agnostic perturbations. In other words, they can be applied to a wide range of input samples to deceive a machine-learning model. Unlike traditional adversarial examples that are specific to individual input samples, universal adversarial examples are designed to generalize across different inputs, potentially fooling the

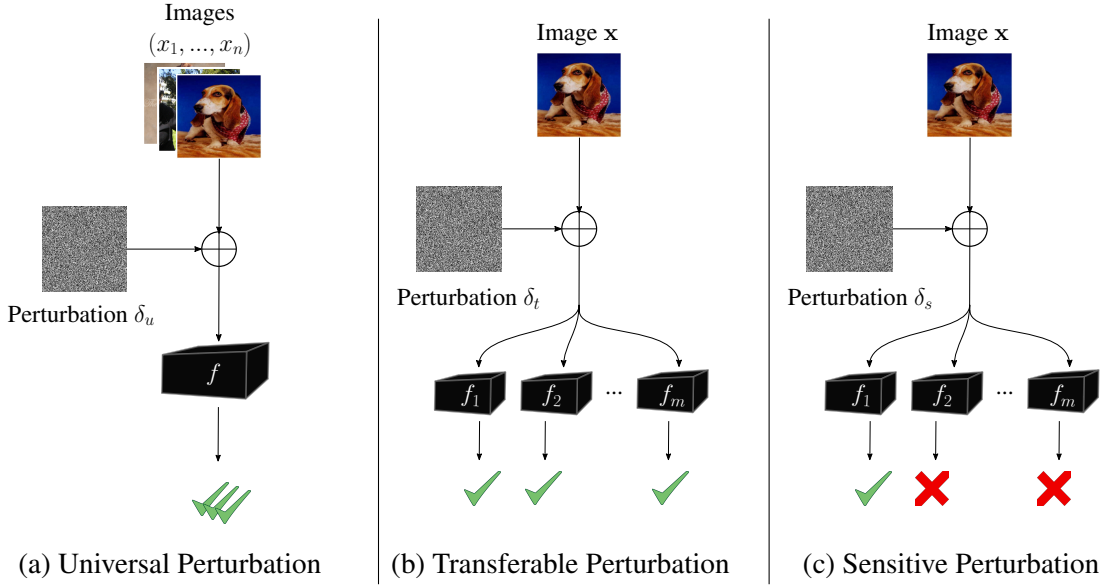


Figure 3.3 – Diagram of the different properties of a single adversarial perturbation.

model consistently. With the notations in Section 3.1.3, the UAP  $\delta_a$  leads to incorrect predictions for all images in the set:

$$\mathbf{y}^i \neq f(\mathbf{x}_o^i + \delta_a); \quad \forall \mathbf{x}_o^i \in \mathcal{X}. \quad (3.10)$$

The concept of UAP has been introduced by Moosavi-Dezfooli [155] on images but has been extended to other fields such as audio [160], text [148] and even image retrieval [130]. It can be easily obtained on a subset of images as shown in Figure 3.4. The universal perturbation is crafted iteratively. At each iteration, one image is selected and the perturbation is updated to fool the model on this image plus the current universal perturbation.

UAPs have significant implications for the security and robustness of machine learning models. If it exists, it means that an attacker could generate a single perturbation that can consistently fool the model regardless of the input [277].

**Transferable Adversarial Perturbations (TAPs)** have been discovered soon after the revelation of adversarial examples [72]. They refer to perturbations that can be crafted on one machine learning model, known as the source model  $f_s$ , and successfully transfer their adversarial properties to other models  $f_1, f_2, \dots, f_m$  performing on the same task. This TAP  $\delta_t$  can



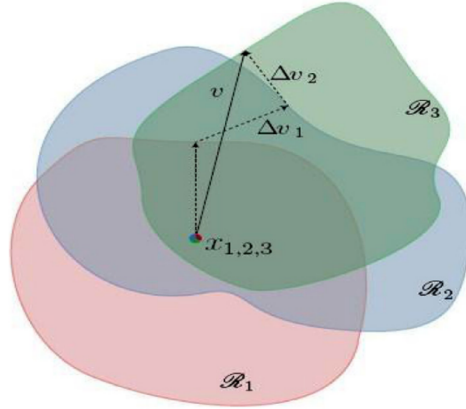


Figure 3.4 – Process of finding universal adversarial perturbations for three images from [155]. Each color represents the classification region of an image.

be expressed as:

$$f_j(\mathbf{x}_o + \delta_t) \neq \mathbf{y}; \quad \forall f_j \in \{f_1, f_2, \dots, f_m\}. \quad (3.11)$$

TAPs are an intriguing phenomenon in the field of adversarial machine learning. It demonstrates that certain adversarial perturbations can exploit vulnerabilities or common weaknesses across different models, leading to consistent misclassification or incorrect predictions. This property has led to the development of a whole category of attacks [250, 136, 94, 265, 52, 96], which is discussed in Section 3.4.1.

**Sensitive perturbations** , as the name suggests, are specifically forged to deceive a specific machine learning model. Unlike TAPs, these perturbations are crafted to exploit the unique vulnerabilities or weaknesses of a particular model, resulting in consistent misclassification or incorrect predictions for that specific model. By keeping the notation in Figure 3.3, the model-unique perturbation  $\delta_s$  can be expressed as:

$$f_s(\mathbf{x}_o + \delta_s) \neq y; f_j(\mathbf{x}_o + \delta_s) = y; \quad \forall f_j \in \{f_1, f_2, \dots, f_m\}. \quad (3.12)$$

This property is not as attractive for attackers compared to transferability or universality, as it is limited to a specific model. However, it holds significance in defense strategies. Specifically, this property becomes relevant when the models under consideration are slight modifications of the target model  $f$ . It has been utilized under the term *Sensitive Examples* [85] to establish the intellectual property of a model and to ensure that the model remains unaltered.

## 3.4 Adversarial Attacks

The discovery of adversarial examples has posed challenges in forging them. Various attacks (*i.e.* processes to craft adversarial examples) have been developed to generate these perturbations. Many of these attacks are gradient-based, utilizing the gradient of the loss function to determine the direction in which to modify the input and create an adversarial example [280, 181, 22, 156, 142, 72, 193]. While these gradient-based attacks are powerful, they often rely on unrealistic assumptions and may not be applicable in real-world scenarios. In this work, we focus on exploring attack methods under realistic assumptions.

### 3.4.1 Transfer-Based Attacks

The threat analysis related to adversarial examples usually considers two scenarios: the white box assumption where the attacker knows the internals of the target model, and the black box assumption where he does not but has limited access to the target.

White box attacks are now performing very well in terms of Attack Success Rate and distortion. We distinguish two kinds of attacks: 1) attacks constrained by a distortion budget like PGD [142] whose performance is measured by the ASR, and 2) attacks yielding almost surely an adversarial example like DeepFool [156] and CW [22] whose performance is measured by the distortion. The trend is to make them speed efficient as well with fewer computations of the gradient of the neural network, like FMN [181] and BP [280, 15].

Transferable attacks leverage the transferability of adversarial examples, which are crafted to deceive one machine learning model, intending to transfer this adversarial property to a different model performing the same task (as discussed in Figure 3.3). To elaborate on this, the assumption underlying transferable attacks is that the attacker possesses knowledge of another model, referred to as the *source* model, which is trained for the same classification problem as the target model. The attacker then conducts a white-box attack on the source model, hoping that the resulting adversarial example also deceives the target model.

Nevertheless, in practice, this direct transfer of adversarial examples from the source to the target model often yields poor results. The reason for this lies in the specificity of the adversarial perturbation generated for the source model. Adversarial perturbations are highly tailored to the specific characteristics and decision boundaries of the source model, making them less effective when directly applied to the target model. This is where transferable attacks stand.

Transferable-based attacks increase the *Attack Success Rate* (ASR) by avoiding the overfitting of examples on the source. Input transformations have been purposed by DI [265], TI [52],

and `Admix` [249]. Papers [250, 136, 94] stabilized the gradient while [254, 93, 282, 290] focus on the importance of intermediate features. The combination of several methods has also been investigated [292, 96]. For instance, `TAIG` [96] uses input transformation, integrated gradients, and attention reduction.

Ensemble-model attacks assume that the attacker leverages not a single source but several source models. Paper [250] averages the logits of several sources and run a white box attack on this aggregation of models, whereas [267] reduces model variance. The main obstacle to these methods is the computational complexity. To counteract this drawback, some methods design a specific ensemble of sources like ghost networks [133] or pruned networks [244]. This amounts to creating several sources without the need to compute the gradient of each model.

**The biggest assumption of these transferable attacks is that they own a model and expect that this model has good average transferability with the target. But it may be too optimistic. The contribution `FiT` in Part III focuses on this property.**

### 3.4.2 Model Surrogate-Based Attacks

These attacks are similar to transfer-based attacks as they leverage the transferability property to move the adversarial example from one model to another. However, the key distinction lies in the ownership of the models. In this case, the attacker does not have access to any existing model. He uses model extraction techniques (see Section 2.2.2) to create his surrogate model to gain a white-box access to the target model, allowing for more effective attacks. An introduction to model extraction attacks is provided in Section 2.2.2. This section focuses on techniques specifically oriented towards adversarial examples. However, it is worth noting that the methods in this section [289, 248, 219] also fall under the category of model extraction attacks.

Nevertheless, there exists a nuanced distinction in their objectives. The attacks outlined in Section 2.2.2 were designed to obtain a model that matches the victim model’s accuracy. Conversely, model surrogate-based attack [289, 248, 219] have a different focus: to acquire a model with high transferability to the victim model, specifically to facilitate evasion attacks. This distinction is shown in the choice of the metrics (accuracy for the former and Attack Success Rate or perturbation distortion for the latter), and it’s further highlighted by the heightened emphasis on the decision boundary within this section.

Early days black-box attacks used a surrogate model that mimics the targeted model. Initially, real data was employed to create the surrogate model [231, 174]. But, recent research has highlighted the challenges of accessing large databases of real data. For instance, it can be difficult for tasks like speech recognition to obtain a vast amount of data. To address this limitation, a new class of attacks called *data-free* attacks has emerged. One of the pioneering approaches in data-free attacks was introduced by `Dast` [289], where they utilized Generative Adversarial Networks (GANs) to generate synthetic data. The GAN’s generator produces images from noise, which are then fed to the target model to obtain the output as the ground truth for training the substitute model. The generator is further used to create new samples to explore the differences between the surrogate and the victim model, covering all possible labels. This approach has been further refined [248, 219], which aimed to find more informative data points and ensure that the decision boundaries of the two networks are close. In terms of query efficiency, `Dast` [289] required a massive number of 20,000,000 queries on Microsoft Azure to attack a MNIST model in a hard-label setup.

A more accessible approach has been proposed by [283], which adopts a two-stage method to reduce the number of required queries. In the first stage, no queries are made to the black box. Instead, the generator generates images solely using the substitute model, which provides the ground truth and the gradient. The generator’s loss function includes a term to maximize the information entropy in the pseudo-labels given by the surrogate, resulting in data diversity with an equal generation of samples for each class. Once correct images are generated, the second stage involves training the surrogate model. This two-stage approach significantly improves the number of queries required. Under the decision-based setup, *only* 20,000 queries are needed on the MNIST dataset and 250,000 queries on CIFAR-10.

The adversarial examples found with these attacks have low distortion. However, while a considerable amount of queries is spent for training the surrogate, not a single adversarial example is forged.

### 3.4.3 Query-Based Attack

In query-based attacks, the attacker does not have access to a model initially. He iteratively queries the black-box to minimize the perturbation’s distortion. These attacks often leverage the low curvature of the decision boundary near the data point [187, 23, 126]. The attack process typically begins with a large random noise as an initial perturbation, sufficient to be adversarial. Subsequently, a line search is employed to approach the decision boundary of the target model.

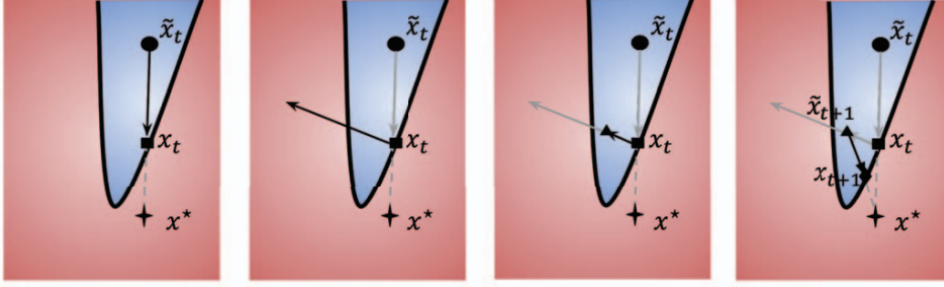


Figure 3.5 – Gradient Estimation for query-based attacks by HSJA [23].

Once in proximity to the boundary, most attacks [187, 23, 126, 25, 235, 285, 139] attempt to estimate the gradient of the boundary  $\mathcal{S}$ , which is not directly accessible to the attacker. With a Monte-Carlo estimation and by treating the decision boundary as a hyperplane, they approximate the direction of the gradient as follows:

$$\widetilde{\nabla}\mathcal{S}(\mathbf{x}, \mu) = \frac{1}{K} \sum_{k=1}^K f(\mathbf{x} + \mu \times u_b) \times u_b, \quad (3.13)$$

where  $\mu$  is a parameter controlling the range of the estimation,  $u_b$  is a random direction sampled from a standard normal distribution, and  $K$  is the number of samples. Then, a step is made in the direction  $\widetilde{\nabla}\mathcal{S}$  as depicted in Figure 3.5.

Improving the query efficiency is an important area of research [187, 24, 126]. Attackers employ various strategies, such as:

- Reducing the dimensionality of the search space: If we consider ImageNet input space, the adversarial stands in an input space of  $\approx 150,000$  dimensions. For our problem, it's like finding a needle in a haystack with the eyes closed. Works tried to reduce the dimension by focusing on important input features. In the context of images, a common approach is to select some frequencies [126, 187, 251]. Low-frequency components, imperceptible to human perception, can significantly impact model predictions [243, 208]. Using the *Discrete Cosine Transform* (DCT), the works [126, 187] specifically target low-frequency perturbations to explore this phenomenon.
- Leveraging query-history: The information gathered from previous queries, whether they were successful or failed, improves the perturbation generation process [211, 129, 251]. While traditional attacks use random directions from a standard normal distribution to update the adversarial, AHA [129] introduces a more efficient approach by using

the query history as a prior. The mean of the Gaussian distribution used to generate the direction follows the last direction if it was successful or its opposite if it failed.

- Leveraging geometrical properties of the input-space: In Section 1.3, we explore the specific properties of the decision boundary. The attackers leverage them to simplify the problem and reduce the number of queries [28, 251]. For instance, in [251], a triangle is constructed using the original image  $\mathbf{x}_o$ , the current adversarial example at iteration  $t$   $\mathbf{x}_{\text{adv},t}$ , and the next adversarial example  $\mathbf{x}_{\text{adv},t+1}$ , where two points are adversarial. By leveraging geometric properties, they formulate an optimization problem to find the optimal angle of the triangle to obtain the new adversarial point  $\mathbf{x}_{\text{adv},t+1}$ .
- Improving initialization: The initial step for any query-based attack involves obtaining the first adversarial point, and a common approach is a random initialization, resulting in a distant starting point from the original image. For instance, [211] utilizes a transfer-based attack to initialize the first noise.

All the ingredients used nowadays were already present in this literature dating back to 1997 [36, 137]: surjection onto the boundary with binary search, estimation of the gradient at a ‘sensitive’ point lying on the boundary, dimension reduction. The new HopSkipJumpAttack [23] is indeed very similar to the old Blind Newton Sensitivity Attack [34] introduced in Section 2.3.1. The last work on this subject by this community [55] surprisingly does not use any gradient estimate but random directions; like the very first decision-based attack [18].

**The state-of-the-art attacks use gradient estimation. As it will be shown in Figure 4.1 in Chapter 4, a large number of queries are used for estimating the gradient direction in a high dimensional space. During this gradient estimation process, no progress is made in reducing the perturbation distortion. The contribution SurFree [146] developed Chapter 4 introduced during this thesis avoids this period of stagnation with a geometric approach, providing a faster attack. It demonstrates how a prior work can be successfully adapted to the context of deep learning highlighting that certain challenges faced today might have been addressed in the past in another field.**

### 3.4.4 Comparison of Numbers of Queries Required

When comparing attacks within the same category, the number of queries needed to craft an adversarial example is a common metric. The more queries are required, the more time the defender has to detect the attack, and it becomes more expensive for the attacker as well. Let’s

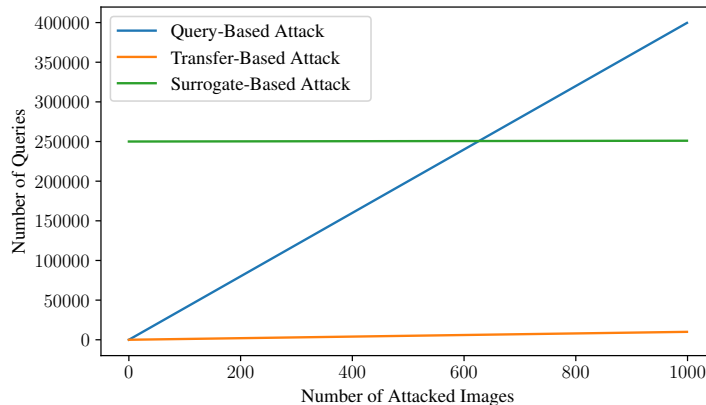


Figure 3.6 – Estimation of the required number of queries as a function of the number of attacked images using CIFAR-10 for transfer-based attacks, surrogate-based attacks, and query-based attacks.

compare the average number of queries to attack an image on each category as a function of the number of attacked images with a 100% ASR. Unfortunately, no prior work on surrogate-based attacks or even model extraction has learned a surrogate on the ImageNet dataset. Even when CIFAR-10 is tested, the number of queries required to train the surrogate may not be provided. The CIFAR-10 dataset is considered for the analysis.

Based on the results from state-of-the-art attack [210], query-based attacks achieve good results after approximately 400 queries on CIFAR-10. For the analysis, we consider this value for the number of queries required by query-based attacks to attack a single image.

When using transferable attacks, by assuming a model possessed by the attacker, the number of queries required is notably reduced. In a few cases, a single query may be enough to transfer the adversarial example from the source model to the target model. However, to ensure a 100% attack success rate (ASR) on the target model, attackers may use a few additional queries to position the adversarial example on the decision boundary of the target model. We assume that 10 queries are adequate to achieve a 100% ASR.

The most effective surrogate-based attack methods typically require at least 250,000 queries to successfully clone the model in the black-box [283]. Once the model is cloned, it is assumed that the decision boundary of the surrogate model closely resembles that of the target model. As a result, any adversarial example crafted on the surrogate model is likely to be adversarial on the target model as well.

Figure 3.6 presents a estimated number of queries required to attack multiple images on CIFAR-10 for each category of attacks based on the number of queries selected above. Transfer-

based attacks are the most effective, but they require that the attacker possesses a model on the same task. When the attacker has no prior information, surrogate-based attacks become interesting compared to query-based attacks after approximately 600 attacked images. Before this point, query-based attacks outperform surrogate-based attacks in terms of required queries. Unfortunately, the literature does not provide the number of queries required to train the surrogate model on ImageNet. However, it is expected to increase with the model’s complexity. Therefore, the gap between query-based attacks and surrogate-based attacks may be even more significant on ImageNet.

It is worth noting that this short analysis does not consider distortion, as it was purposely left out. Even though surrogate-based attacks may be more efficient in terms of distortion, all methods obtained distortion within a similar range of values.

## 3.5 Defenses

The adversarial examples involve an ongoing game between the attacker and the defender. The attacker continuously develops new innovative attack techniques to find the most effective and efficient adversarial perturbations. On the other hand, the defender attempts to find ways to prevent the attacker from finding such perturbations. This adversarial relationship drives the advancement of both attack and defense methods in the field of adversarial machine learning with always a step ahead for the attacker.

In the previous section, we explored various attack methods used to craft adversarial examples under realistic scenarios. In this section, we shift our focus to defense methods. We distinguish three main types of defenses against realistic attacks: pre-processing, detection, and training. Other defenses exist such as obfuscation [7, 47] which consist in masking the parameters and the gradients of the model. However, these methods did not show efficiency or are almost useless against evasion attacks.

### 3.5.1 Pre-preprocessing defenses

These defense mechanisms aim to enhance the model’s robustness by applying transformations to the input before it is fed to the model. A diverse range of input transformations have been proposed, including image cropping and rescaling, bit depth reduction [268], JPEG compression [207, 75, 41, 54, 117], histogram equalization[185, 177], and more. However, some of these defenses have been shown to be inefficient or not effective as soon as the attacker knows



the defense [16, 7]. Additionally, some transformations may lead to a drop in model accuracy and require retraining to maintain performance [278, 75].

Randomization has emerged as a promising strategy to enhance the robustness of deep learning models against evasion attacks [75, 264]. The randomness induces instability in the decision boundary. A strategy has been proposed by BaRT [185] to address the limitations of individual pre-processing defenses. It takes a comprehensive approach by randomly combining 25 different transformations. This multi-defense strategy aims to enhance overall robustness effectively by leveraging a diverse set of techniques.

However, it is important to note that while randomization significantly increases the cost of decision-based black-box attacks, it may not truly increase the robustness of the model compared to other defenses [67]. Evasion attacks are still possible despite the presence of randomization. One common way to bypass randomization is by using *Expectation Over Transformation* (EOT). EOT computes the "correct output" of the model by averaging the model's output over all possible transformations [7]. It means that the attacker must query a single input multiple times and then aggregate the gradients for reliable outputs. This technique has been used to break several randomized defenses [75, 264], as demonstrated by Athalye *et al.* [7].

The effectiveness of randomization as a defense strategy against the creation of adversarial examples was demonstrated a long time ago. As far back as 1998, this approach had already proven to be successful in countering oracle attacks in the context of watermarking [138]. In the work [138], the authors suggested introducing random perturbations to the input data as a means to thwart attackers from pinpointing the decision boundary.

**This thesis studies the effect of randomization through randomized smoothing, a technique commonly employed to certify the robustness. It involves aggregating the model's predictions on a set of random perturbations added to the input. In Chapter 6 with the contribution RS, its effectiveness is illustrated in mitigating decision-based black box attacks [145].**

### 3.5.2 Detection

When discussing detection in decision-based black-box scenarios, there are two main strategies. The first strategy, applicable to any evasion attack, involves detecting adversarial examples themselves. The second strategy, specific to query-based attacks, focuses on detecting the presence of the attacker. This section is an overview of detection methods. See Ahmed Aldah-

dooh [3] for a more complete survey.

**Adversarial Detection** Defenders have various methods to detect adversarial inputs. One approach is based on analyzing the properties of the decision boundary. For example, He *et al.* [83] utilize neighborhood analysis, considering the distance to decision boundaries and the number of different classes nearby, to distinguish between benign and adversarial examples. Another strategy involves measuring the consistency of the model’s output when the input is modified. For example, Li *et al.* [131] use context inconsistency by removing parts of the images to detect adversarial examples. Another approach known as `Feature Squeezing`[268] reduces the color bit depth of the input (although any transformation discussed in Section 3.5.1 could be used).

However, a majority of detectors concentrate on the data manifold [209, 101, 216, 152, 198, 37, 104]. Adversarial examples are assumed to lie in lower probability regions, away from the data manifold, and this property is exploited by many detectors. It amounts to measuring the distance of the input from the manifold. For instance, [101] identifies the nearest neighbors in the data manifold to detect adversarial examples, while [216] employs generative models to model the training data distribution, and [152] uses a generative model to recreate the input and detect if it is adversarial or not.

**Stateful defenses** Evasion attacks under a realistic scenario often query inputs that are extremely close, for instance, to estimate the gradient of the decision boundary or to perform a binary search. One way to overcome this issue is to use a stateful defense [61, 128, 26]. These defenses operate by tracking a history of incoming queries, and rejecting those that are suspiciously similar. To do it, they need a similarity engine to compare the incoming query and to store a history of the queries. Chen *et al.* [26] firstly propose to encode the query with a deep similarity encoder and then performs a *k-nearest neighbors* (kNN) algorithm in the encoded space with the previously seen examples. The high complexity of this technique is addressed by Li *et al.* [128] which replace the similarity engine with a hash-based approach.

This defense has previously been explored in earlier literature within a different domain [9, 230]. As previously mentioned in Section 1.5, evasion attacks resemble oracle attacks, and analogous strategies have already been advanced in this context. For example, Mauro Barni *et al.* [9] imagined a detection mechanism to counteract an attack. They concentrated on the presumption that the majority of oracle attacks aim to find examples located near the decision boundary. They also suggested a method to identify the line search employed by the attacker to position

themselves on the decision boundary. This assumption remains valid in recent works on evasion attacks [23, 24]. Therefore, the defense mechanism would remain effective.

### 3.5.3 Data Augmentation

A potential explanation for the existence of adversarial examples is that the model's training data might not fully cover the input space, leading to overfitting (as discussed in Section 3.2). To address the issue of overfitting, a straightforward solution is to augment the training data. This approach, known as *data regularization* or *data augmentation*, aims to prevent overfitting by providing the model with a more diverse and representative training dataset.

Adversarial training is the most famous of these methods for improving the robustness where the training dataset is augmented with adversarial examples [203]. Considered the most powerful defense against adversarial examples, adversarial training has been early proposed by Szegedy *et al.* [221] and Ian Goodfellow *et al.* [72] but has been successfully applied 5 years later by Madry *et al.* [143]. During the training process, the model is exposed to both clean examples from the original dataset and adversarial examples generated using various attacks.

Unfortunately, it is not a free ride. Two main drawbacks need to be considered. Firstly, the process involves higher complexity compared to traditional training methods due to the computation of adversarial examples at each model update. Researchers have made efforts to address this issue [259, 288, 49, 205]. For instance, Zheng *et al.* [288] introduce a strategy to limit the training by accumulating adversarial perturbations through epochs, leveraging the transferability of adversarial perturbations over time. A similar idea is used by [205] on the gradients. Another challenge lies in the selection of the training data to generate adversarial examples. Dolatabadi *et al.* [49] argue that constructing adversarial examples for each individual data point in the training set can be inefficient and costly. Instead, they propose a *coreset selection* approach to choose a weighted subset of training data that can approximate the full gradient. This helps to reduce the computational burden of adversarial training while still achieving meaningful robustness for the model.

The second drawback of adversarial training is the loss of accuracy on clean examples. This trade-off between standard accuracy and robustness is well-documented and can be considered as an instance of the "No Free Lunch" theorem in machine learning [234]. For example, when a model is trained on the CIFAR-10 dataset using adversarial training, its standard accuracy may drop from 93.6% to 87.3% [186].

Adversarial training is not the only way to augment the training set. Other augmentation methods [123, 239, 281] are efficient in increasing model robustness. For instance, MixUp [281]

and Manifold MixUp [239] are two methods that augment the training set with linear combinations of examples. Two images  $\mathbf{x}_i$  and  $\mathbf{x}_j$  of labels  $y_i$  and  $y_j$  are mixed to create a new training image  $\tilde{\mathbf{x}}$  with label  $\tilde{y}$  such as:

$$\begin{aligned}\tilde{\mathbf{x}} &= \lambda\mathbf{x}_i + (1 - \lambda)\mathbf{x}_j \\ \tilde{y} &= \lambda 1_{y_i} + (1 - \lambda)1_{y_j},\end{aligned}$$

with  $1_{y_i}$  and  $1_{y_j}$  the one-hot encoding of the labels  $y_i$  and  $y_j$  and  $\lambda \in [0, 1]$ .

Data augmentation, particularly through adversarial training, has emerged as a promising approach to enhance the model’s robustness. Despite its drawbacks, adversarial training remains the most effective defense against adversarial examples.

## 3.6 Conclusion

This section offered a brief introduction to adversarial examples to understand their properties, their creation, and how to protect against them. This section remains a short overview of a wide subject.

The interplay between the defender and the attacker is far from futile; in fact, it is crucial in the development of DNNs. Adversarial examples provide a unique opportunity to gain insights into the vulnerabilities of DNNs. While concerns about confidentiality, integrity, and availability are paramount, understanding these weaknesses can lead to more effective and secure real-world applications. Real-world data often differ from idealistic datasets, introducing distribution shifts and noisy inputs that were not encountered during training. As we have observed, our understanding of adversarial examples is still far from complete.

During this thesis, special attention has been given to the topic of adversarial examples, considering its predominance in the security of machine learning. The surrogate-based attacks have been left aside. Their huge cost (computational and in queries) has led us to look at query-based and transferable attacks that offer more realistic attacks on a large number of different data types. Part II is dedicated to discussing the contributions made during this thesis to the field of adversarial examples.



PART II

# **Contribution on Adversarial Examples**

---



# SURFREE [146]: A FAST SURROGATE-FREE BLACK BOX ATTACK

---

## 4.1 Introduction

It is striking that black box attacks mostly use substitution to replace information that are missing. Early black box attacks used a surrogate model (trained from a huge number of input and output pairs) mimicking the targeted model [173, 174] to perform white-box attacks (See Section 3.4.1). Almost all recent score-based attacks resort to gradient estimation to compensate for the lack of back-propagation, which is the key instrument of any white-box attack [25, 235, 99, 285]. Works on score-based attacks managed to reduce the query amount from millions of requests [98] to less than a thousand with most recent approaches [285]. The number of queries became a key point in the research of new attacks. Surprisingly, this impressive decrease has not reached comparable levels in the hard-label setup.

This chapter first highlights that the recent attacks in that setup, HSJA [23], QEBA [126] and GeoDA [187] all perform costly gradient surrogate estimations. SurFree [146] proposes to bypass these, by instead focusing on careful trials along diverse directions, guided by precise indications of geometrical properties of the classifier decision boundaries. We motivate this geometric approach before performing a head-to-head comparison with previous attacks with the number of queries as a first-class citizen. SurFree [146] exhibit a faster distortion decay under low query amounts (a few hundred to a thousand).

This argument should challenge any substitution mechanisms. They all consume a fair amount of queries and it is not clear whether they are worth the gain in terms of distortion. Especially, many techniques trade some query amount for an accurate gradient estimate giving birth to good perturbation directions [187, 23]. During this step, the adversarial is not updated and the distortion stalls as Figure 4.1 shows.

This chapter considers the query amount as a central criterion. It presents a *fast* black box decision-based attack, named SurFree [146], motivated by practical applications in which a



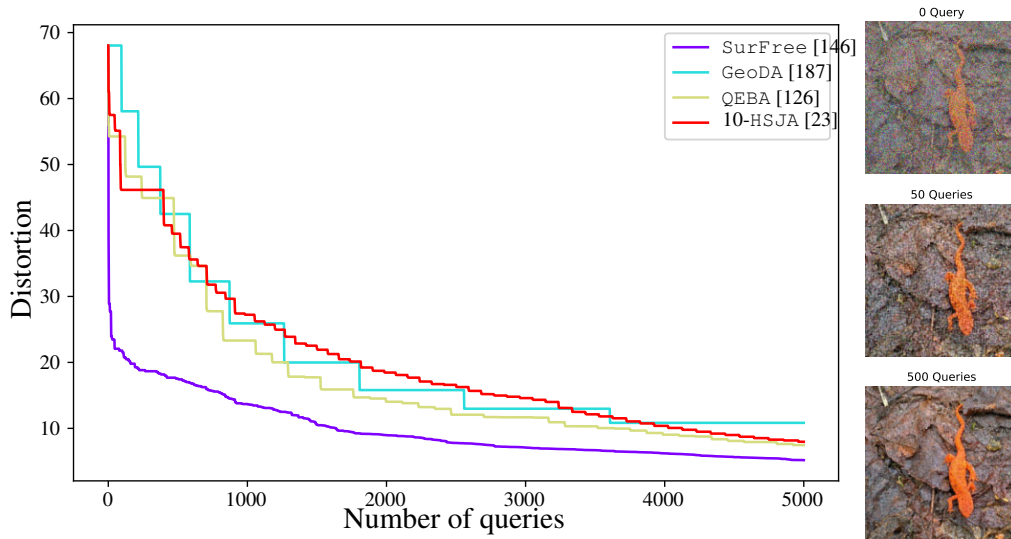


Figure 4.1 – The perturbation distortion ( $\ell_2$  norm) vs. the number of queries for image ‘lizard’. The estimation of a gradient surrogate results in plateaus of distortion for GeoDA [187], QEBA [126], and HSJA [23]. SurFree [146] avoids this waste of queries with a random walk and geometric approximation.

low amount of queries is key. Fast means that it outperforms the state of the art when it comes to the distortion of adversarial under a low query budget (as exemplified in Figure 4.1 with the purple curve).

The main contributions of this chapter are:

- SurFree [146], a black box decision-based attack not using any substitution mechanism: no surrogate of the target model, no score reconstruction, and no estimation of a gradient. It is inspired by the early works [55, 18].
- a geometrical mechanism to get the biggest distortion decrease for a given direction to be explored under the assumption of a hyperplane boundary [60].
- a head-to-head comparison of the recent approaches with distortion as a function of query number.

Experimental results show that SurFree [146] overcomes state of the art on the query amount factor (a thousand queries), while remaining competitive with unlimited queries (normal scenario for competitors).

*All the notations unique to this chapter are listed in Table 7.11 at the conclusion of this manuscript. Additionally, for general notation, please refer to Table 7.9.*

## 4.2 Problem statement

The attacker does not know the function  $f$  and can only observe the decision  $\text{cl}(\mathbf{x})$  for any image  $\mathbf{x}$ . From an original well-classified image  $\mathbf{x}_o$ , the attack is untargeted as it looks for an image  $\mathbf{x}_{\text{adv}}$  close to  $\mathbf{x}_o$  and s.t.  $\text{cl}(\mathbf{x}_{\text{adv}}) \neq \text{cl}(\mathbf{x}_o)$ . This defines the outside region  $\mathcal{O} := \{\mathbf{x} \in \mathbb{R}^D : \text{cl}(\mathbf{x}) \neq \text{cl}(\mathbf{x}_o)\}$  and the optimal adversarial image:

$$\mathbf{x}_{\text{adv}}^* = \arg \min_{\mathbf{x} \in \mathcal{O}} \|\mathbf{x} - \mathbf{x}_o\|. \quad (4.1)$$

This is a hard problem and the attack is indeed an efficient algorithm for finding an approximate solution.

We assume that when knowing a point  $\mathbf{x} \in \mathcal{O}$ , it is possible to find a point  $\mathbf{x}_b$  on the segment  $\overline{\mathbf{x}_o \mathbf{x}}$  that lies on the boundary denoted by  $\partial\mathcal{O}$ . This is usually done by a line search in the literature [23, 126, 187]. There has been experimental evidence that the boundary is a rather smooth low curvature surface for DNNs [59]. This justifies that the boundary is often approximated by a hyperplane locally around a boundary point: in other words, locally around  $\mathbf{x}_b \in \partial\mathcal{O}$ , there exists  $\mathbf{n} \in \mathbb{R}^D$ ,  $\|\mathbf{n}\| = 1$  s.t.  $\mathbf{x} \in \mathcal{O}$  if  $\mathbf{x}^\top \mathbf{n} \geq \mathbf{x}_b^\top \mathbf{n}$ .

## 4.3 Our Approach

The study of the recent attacks [23, 187, 126] under the query budget viewpoint, reveals the presence of plateaus (see Figure 4.1). These are due to the construction of a surrogate for gradients and appear to be particularly costly. Moreover, the budget allocated to gradient estimate in [23] does not impact the speed of convergence: fewer queries give less accurate gradient estimates yielding a smaller distortion decrease but at a higher rate. Our rationale is to set this query budget to its extreme value, *i.e.* zero. We thus trade this budget for more directions investigated with the hope that their multiplication allows for a faster distortion decrease. We now develop this idea.

### 4.3.1 Basic idea

Let us assume that we know a point on the boundary:  $\mathbf{x}_b \in \partial\mathcal{O}$ . We define  $d := \|\mathbf{x}_b - \mathbf{x}_o\|$  and  $\mathbf{u} := (\mathbf{x}_b - \mathbf{x}_o)/d$  so that  $\|\mathbf{u}\| = 1$ . We restrict the search for a closer adversarial point in a random affine plane  $\mathcal{P}$  of dimension 2. This plane  $\mathcal{P}$  contains the point  $\mathbf{x}_o$  and is spanned by vector  $\mathbf{u}$  and a random orthogonal direction  $\mathbf{t} \in \mathbb{R}^D$ ,  $\|\mathbf{t}\| = 1$ ,  $\mathbf{t}^\top \mathbf{u} = 0$ . Note that  $\mathbf{x}_b \in \mathcal{P}$ .

In polar coordinates, we consider a point in  $\mathcal{P}$  that is at a distance  $d(1 - \alpha)$  from  $\mathbf{x}_o$  and makes an angle  $\theta$  with  $\mathbf{u}$ :

$$\mathbf{z}(\alpha, \theta) = d(1 - \alpha) (\cos(\theta)\mathbf{u} + \sin(\theta)\mathbf{t}) + \mathbf{x}_o, \quad (4.2)$$

with  $\alpha \in [0, 1]$  and  $\theta \in [-\pi, \pi]$ . Note that  $\mathbf{z}(0, 0) = \mathbf{x}_b$  and  $\mathbf{z}(1, \theta) = \mathbf{x}_o, \forall \theta$ . If  $\mathbf{z}(\alpha, \theta)$  is adversarial, then the distortion decreases by  $100 \times \alpha\%$ .

This section shows how to choose  $(\alpha, \theta)$  to raise the probability of  $\mathbf{z}(\alpha, \theta)$  being adversarial. This study makes a clear cut with [55, 18] which also considers random directions.

This section assumes that the intersection  $\partial\mathcal{O} \cap \mathcal{P}$  is a line passing by  $\mathbf{x}_b$  and with normal vector  $\mathbf{n} \in \mathcal{P}$ ,  $\|\mathbf{n}\| = 1$ . Without loss of generality,  $\mathbf{n}$  is pointing outside s.t. a point  $\mathbf{z} \in \mathcal{P}$  is adversarial if  $(\mathbf{z} - \mathbf{x}_b)^\top \mathbf{n} \geq 0$ . In polar coordinates,  $\mathbf{n} := \cos(\psi)\mathbf{u} + \sin(\psi)\mathbf{t}$  with  $\psi \in (-\pi/2, \pi/2)$ .

The point  $\mathbf{z}(\alpha, \theta) \in \partial\mathcal{O} \cap \mathcal{P}$  minimizing the distance from  $\mathbf{x}_o$  is the projection of  $\mathbf{x}_o$  onto this line, obtained for  $\theta = \psi$  and  $\alpha = 1 - \cos(\psi)$ . The attacker can not create this optimal point because angle  $\psi$  is unknown. Note that

- If  $\psi = 0$ , then  $\mathbf{n} = \mathbf{u}$ ,  $\mathbf{t}^\top \mathbf{n} = 0$ ,  $(\mathbf{z}(\alpha, \theta) - \mathbf{x}_b)^\top \mathbf{n} = d((1 - \alpha)\cos(\theta) - 1) < 0$ , and  $\mathbf{z}(\alpha, \theta)$  is not adversarial. This corresponds to the case where  $\partial\mathcal{O} \cap \mathcal{P}$  is a tangent line of the circle of center  $\mathbf{x}_o$  and radius  $d$ . This implies that  $\mathbf{x}_b$  is already optimum because it is the projection of  $\mathbf{x}_o$  onto  $\partial\mathcal{O} \cap \mathcal{P}$ .
- If  $\theta = 0$  and  $\alpha > 0$ , then  $(\mathbf{z}(\alpha, 0) - \mathbf{x}_b)^\top \mathbf{n} = \alpha(\mathbf{x}_o - \mathbf{x}_b)^\top \mathbf{n} < 0$  because  $\mathbf{x}_o$  is not adversarial. Therefore,  $\mathbf{z}(\alpha, 0)$  is not adversarial.

For  $\theta \neq 0$ , calculation shows that  $\mathbf{z}(\alpha, \theta)$  is adversarial if

$$g_\alpha(\theta) := \left| \frac{1 - (1 - \alpha)\cos(\theta)}{(1 - \alpha)\sin(\theta)} \right| \leq \tan(\psi)\text{sign}(\theta). \quad (4.3)$$

Point  $\mathbf{z}(\alpha, \theta)$  might be adversarial only if  $\psi$  and  $\theta$  share the same sign s.t. the rhs (4.3) is positive. In this case, the surprise is that (4.3) separates parameters  $(\alpha, \theta)$  that the attacker controls from the unknown angle  $\psi$ .

Minimizing  $g_\alpha(\theta)$  raises the chances that (4.3) holds. Its derivative cancels for  $\theta = \theta^*(\alpha) := \pm \arccos(1 - \alpha)$  (according to the sign of  $\psi$ ) so that

$$g_\alpha(\theta^*(\alpha)) = \frac{\sqrt{1 - (1 - \alpha)^2}}{1 - \alpha} = |\tan(\theta^*(\alpha))|. \quad (4.4)$$

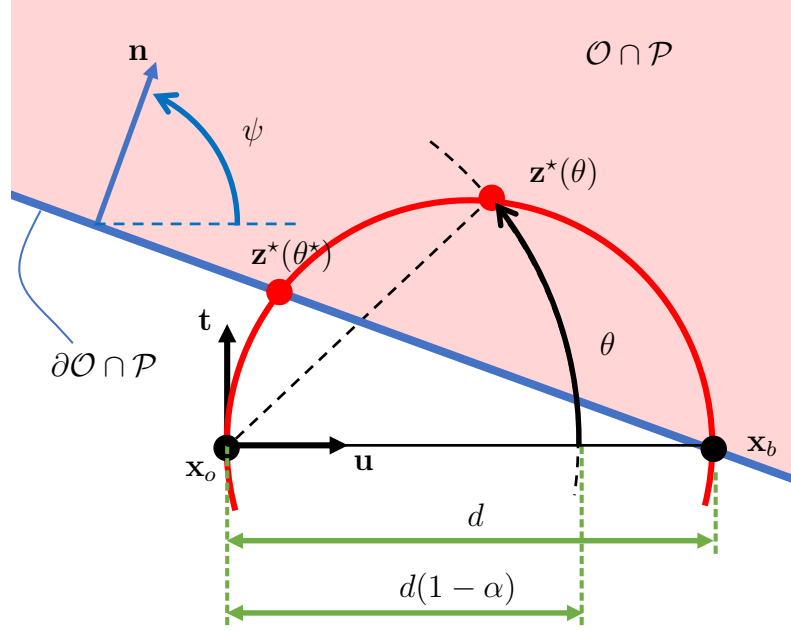


Figure 4.2 – The geometrical configuration of the problem in  $\mathcal{P}$ .

This quantity is an increasing function of  $\alpha$  ranging from 0 ( $\alpha = 0$ ) to  $+\infty$  ( $\alpha \rightarrow 1$ ). From now on, we denote by  $\mathbf{z}^*(\theta) := \mathbf{z}(1 - \cos(\theta), \theta)$  a point created with this coupling.

**Property 1** Consider the mid-point  $\mathbf{z} = (\mathbf{x}_o + \mathbf{x}_b)/2$ . The locus of the points  $\mathbf{z}^*(\theta) \in \mathcal{P}$  is the circle of center  $\mathbf{z}$  and radius  $d/2$ . Indeed,  $\mathbf{z}^*(0) = \mathbf{x}_b$  and  $\mathbf{z}^*(\pm\pi/2) = \mathbf{x}_o$ .

A little algebra shows that  $\|\mathbf{z}^*(\theta) - \mathbf{z}\| = d/2, \forall \theta \in [-\pi/2, \pi/2]$ . This circle is depicted in red in Figure 4.2.

**Property 2** If  $\mathbf{z}^*(\theta)$  is adversarial, then so is  $\mathbf{z}^*(\phi)$  for  $\phi \in [0, \theta]$ . Conversely, if  $\mathbf{z}^*(\theta)$  is not adversarial, then so is  $\mathbf{z}^*(\phi)$  for  $\phi \in [\theta, \text{sign}(\theta).\pi/2]$ .

This is due to the monotonicity of function  $\alpha \rightarrow g_\alpha(\theta^*(\alpha))$ .

**Property 3**  $\theta^* = \psi$  is the angle yielding a maximum distortion decrease of  $\alpha = 1 - \cos(\psi)$ . The point  $\mathbf{z}^*(\theta^*)$  is indeed the projection of  $\mathbf{x}_o$  on the boundary line  $\partial\mathcal{O} \cap \mathcal{P}$ :  $\mathbf{z}^*(\theta^*) = d \cos(\psi)\mathbf{n} + \mathbf{x}_o$ .

This is shown by injecting (4.4) in (4.3).

### 4.3.2 Iterations over orthonormal directions

This section assumes that the boundary  $\partial\mathcal{O}$  is an affine hyperplane passing through  $\mathbf{x}_{b,1}$  in  $\mathbb{R}^D$ , with normal vector  $\mathbf{N}$ . We consider a random basis of  $\text{span}(\mathbf{x}_{b,1} - \mathbf{x}_o)^\perp$  composed of  $D - 1$  vectors  $\{\mathbf{t}_i\}_{i=1}^{D-1}$ . The normal vector is decomposed in spherical coordinates:

$$\begin{aligned} \mathbf{N} &= \sin(\psi_{D-1})\mathbf{t}_{D-1} + \cos(\psi_{D-1})\sin(\psi_{D-2})\mathbf{t}_{D-2} + \\ &\dots + \cos(\psi_{D-1})\dots\cos(\psi_2)\mathbf{n}_1, \end{aligned} \quad (4.5)$$

where  $\mathbf{n}_1 := \sin(\psi_1)\mathbf{t}_1 + \cos(\psi_1)\mathbf{u}_1$  is the  $\ell_2$  normalized projection of  $\mathbf{N}$  onto hyperplane  $\mathcal{P}_1$  spanned by  $\mathbf{t}_1$  and  $\mathbf{u}_1 := (\mathbf{x}_{b,1} - \mathbf{x}_o)/d$ . Note that  $\mathbf{N}^\top \mathbf{u}_1 = \cos(\psi_{D-1})\dots\cos(\psi_1)$ . Then Prop. 3 finds  $\mathbf{x}_{b,2} := \mathbf{z}^*(\theta^*) \in \mathcal{O} \cap \mathcal{P}_1$  and defines  $\mathbf{u}_2 := (\mathbf{x}_{b,2} - \mathbf{x}_o)/d \cos(\psi_1) = \mathbf{n}_1$ . We iterate on  $\mathcal{P}_2$  spanned by  $(\mathbf{t}_2, \mathbf{u}_2)$  to get  $\mathbf{N}^\top \mathbf{u}_2 = \cos(\psi_{D-1})\dots\cos(\psi_2) \geq \mathbf{N}^\top \mathbf{u}_1$ .

**Property 4** *Iterating this process converges to the adversarial point with minimal distortion.*

Iterations increase the scalar product between  $\mathbf{N}$  and  $(\mathbf{x}_{b,k} - \mathbf{x}_o) \propto \mathbf{u}_k$  given by:

$$\mathbf{N}^\top \mathbf{u}_k = \prod_{i=1}^{D-k} \cos(\psi_{D-i}). \quad (4.6)$$

At the end,  $\mathbf{x}_{b,D} \in \mathcal{O}$  and  $\mathbf{x}_{b,D} - \mathbf{x}_o$  is colinear with  $\mathbf{N}$ , thus pointing to the projection of  $\mathbf{x}_o$  to the hyperplane boundary.

A clever strategy browses the directions according to the decreasing order of their angles  $(|\psi_k|)_k$  (biggest distortion decreases first). This is out of reach for the attacker oblivious to  $\mathbf{N}$  and not willing to spend queries for its estimate.

### 4.3.3 Convex boundary

Our procedure can be seen as a coordinate descent on a random basis. If the boundary  $\partial\mathcal{O}$  is not a hyperplane but a smooth and convex surface, then cycling over the vectors  $\{\mathbf{t}_i\}_{i=1}^{D-1}$  multiple times ensures convergence to a local minimum [161]. On one hand, this reference shows that the rate of convergence of the random coordinate descent (on expectation) is essentially the same as the *worst-case* rate of the standard gradient descent (when it is available). On the other hand, estimating the gradient in the black box setting costs more queries than the coordinate descent of Section 4.3.1. These conflicting arguments deserve investigation.

## 4.4 The SurFree [146] attack

This section presents the attack based on the ideas explained in Section 4.3. One iteration of SurFree [146] is summarized in pseudo-code Algorithm 1.

### 4.4.1 The algorithm

**Initialisation.** The algorithm needs an initial point  $\mathbf{x}_{b,1} \in \partial\mathcal{O}$ . It first generates a point  $\mathbf{y}_0 \in \mathcal{O}$ . As done in [187, 126],  $\mathbf{y}_0$  is one image from the targeted class (targeted attack) or a noisy version of  $\mathbf{x}_o$  (untargeted attack). Defining  $\mathbf{y}_\lambda = \lambda\mathbf{x}_o + (1 - \lambda)\mathbf{y}_0$ , a binary search over  $\lambda \in (0, 1)$  results in  $\mathbf{x}_{b,1}$  adversarial and close to the boundary.

**New direction.** At iteration  $k$ , the point  $\mathbf{x}_{b,k} \in \mathcal{O}$  and close to  $\partial\mathcal{O}$  defines  $\mathbf{u}_k \propto \mathbf{x}_{b,k} - \mathbf{x}_o$ ,  $\|\mathbf{u}_k\| = 1$ . Line 3 generates pseudo-randomly  $\mathbf{t}_k \sim \mathcal{T}$  (see Section 4.4.2). A Gram-Schmidt procedure makes it orthogonal to  $\mathbf{u}_k$  and to the  $L$  (at most) last directions  $\mathcal{V}_{k-1} := \{\mathbf{t}_j\}_{j=\max(k-L,1)}^{k-1}$ , producing the new direction  $\mathbf{t}_k$  in line 4.

**Sign Search.** The algorithm considers points  $\mathbf{z}(\alpha, \theta)$  as defined in (4.2) with  $\mathbf{u} = \mathbf{u}_k$ ,  $\mathbf{t} = \mathbf{t}_k$ ,  $d_k := \|\mathbf{x}_{b,k} - \mathbf{x}_o\|$ , and the coupling  $\cos(\theta) = 1 - \alpha$ . The sign of  $\theta$  depends on the sign of unknown  $\psi$  (see Section 4.3.1). Hence, we test  $T$  angles starting with the biggest amplitudes, alternating + and - sign, as stored in the vector  $\theta_{\max} \cdot \boldsymbol{\tau}$  with:

$$\boldsymbol{\tau} := (1, -1, (T-1)/T, -(T-1)/T, \dots, 1/T, -1/T). \quad (4.7)$$

The search stops as soon as an adversarial image is found. If this fails, line 17 decreases  $\theta_{\max}$ , direction  $\mathbf{t}$  is given up (line 18), and another direction is generated.

**Binary Search.** When the sign search finds an adversarial image at  $\theta = \theta_{\max}t/T$ , the binary search (line 12) refines the angle  $\theta$  over the interval  $\theta_{\max}[t, t + \text{sign}(t)]/T$  within  $\ell$  steps. The result is  $\theta^*$  and  $\mathbf{z}^*(\theta^*)$  is the new boundary point  $\mathbf{x}_{b,k+1}$  provoking a distortion decrease  $\alpha^* = 1 - \cos(\theta^*)$ .

### 4.4.2 Distribution of the directions

The algorithm is a random process as it draws directions from distribution  $\mathcal{T}$  according to Algorithm 2. This has two roles: dimension reduction and adaptivity to the content of  $\mathbf{x}_o$ .

Dimension reduction is implemented with the help of a reversible image transformation (DCT  $8 \times 8$ , or full frame DCT in Table 4.1). Line 3 selects a fraction  $\rho$  of the transform coefficients, typically in the low-frequency subband. We draw  $\rho D$  samples uniformly distributed

over  $\{-1, 0, 1\}$ , the other transform coefficients being set to 0. The inverse transform yields the direction  $\mathbf{t}$  in the pixel domain.

Adaptivity to the visual content makes the perturbation less perceptible thanks to the masking effect well known in watermarking [55]. It shapes the adversarial perturbation like the visual content of  $\mathbf{x}_o$ . The following is a simple implementation of this principle: denote the  $i$ -th transform coefficient of image  $\mathbf{x}_o$  by  $X_{o,i}$ . Line 5 modulates the amplitude of a random variables by  $A(|X_{o,i}|)$ , where  $A : \mathbb{R}^+ \rightarrow \mathbb{R}^+$  is a non-decreasing function. The goal is to shape the power distribution of the perturbation as one of the images.

### 4.4.3 Interpolation

Section 4.3 motivated our design assuming the boundary is a hyperplane. This extra interpolation is an *option* of *SurFree* [146] inspired by the watermarking attack [55], which tackles convex surfaces with small curvature as in Figure 4.3.

A given iteration starts with  $\mathbf{x}_{b,k} \in \partial\mathcal{O}$  at angle  $\theta = 0$  and distance  $d$ . The binary search in line 12 gives the angle  $\theta^*$  of a boundary point at distance  $d \cos(\theta^*)$ . This option finds a third point on the boundary at angle  $\theta^*/2$  thanks to a binary search between  $\mathbf{x}_o$  and  $\mathbf{z}^*(\theta^*/2)$ . This point, depicted in blue in Figure 4.3, is at distance  $\delta \leq d \cos(\theta^*/2)$ .

Thanks to these three boundary points resp. at angle 0,  $\theta^*/2$ , and  $\theta^*$ , we interpolate the mapping from angle to distance (of the surjection of  $\mathbf{z}(\alpha, \theta)$  onto the boundary) by a second order polynomial and find its minimum at:

$$\hat{\theta} = \frac{\theta^* 4\delta - d(\cos(\theta^*) + 3)}{4 2\delta - d(\cos(\theta^*) + 1)}. \quad (4.8)$$

This option concludes by a binary search finding the point on the boundary between  $\mathbf{x}_o$  and  $\mathbf{z}^*(\hat{\theta})$ . The new point  $\mathbf{x}_{b,k+1}$  is the closest point we found on the boundary.

## 4.5 Experimental Work

We first specify the experimental setup and the parameters of our approach. We then perform an ablation study on *SurFree* [146] (Section 4.5.2), for it allows to precise gains on the two considered metrics. Section 4.5.3 performs a head-to-head comparison of all the competing approaches.

**Algorithm 1** One iteration of SurFree [146]

---

**Require:** Original image  $\mathbf{x}_o$ , boundary point  $\mathbf{x}_{b,k} \in \partial\mathcal{O}$ , previous directions  $\mathcal{V}_{k-1} := \{\mathbf{t}_j\}_{j=\max(k-L,1)}^{k-1}$

**Ensure:** Output  $\mathbf{x}_{b,k+1} \in \partial\mathcal{O}$ ,  $\mathcal{V}_k$

- 1: **New direction**
- 2:  $\mathbf{u}_k = \eta(\mathbf{x}_{b,k} - \mathbf{x}_o)$  ▷  $\eta(\mathbf{x}) := \mathbf{x}/\|\mathbf{x}\|$
- 3:  $\mathbf{t}_k \sim \mathcal{T}$  ▷ Algorithm 2
- 4:  $\mathbf{t}_k = \eta\left(\text{proj}_{\text{span}(\mathcal{V}_{k-1} \cup \mathbf{u}_k)^\perp}(\mathbf{t}_k)\right)$  ▷ Gram-Schmidt
- 5:  $\mathcal{V}_k = \mathcal{V}_{k-1} \cup \{\mathbf{t}_k\}$
- 6: **Sign Search**
- 7:  $j = 1$ ,  $\boldsymbol{\tau} = (T, -T, (T-1), -(T-1), \dots, 1, -1)/T$
- 8: **while**  $\mathbf{z}^*(\theta_{\max} \cdot \tau_j) \notin \mathcal{O} \wedge j \leq 2T$  **do**
- 9:      $j \leftarrow j + 1$
- 10: **if**  $j < 2T$  **then**
- 11:     **Binary Search**
- 12:      $\theta^* = \text{BS}(\theta_{\max} \cdot \tau_j; \theta_{\max}(\tau_j + \text{sign}(\tau_j)/T))$
- 13:      $\theta_{\max} \leftarrow \theta_{\max}/(1 - \kappa)$
- 14:     **Return**  $\mathbf{x}_{b,k+1} = \mathbf{z}^*(\theta^*)$
- 15:     **or** Interpolation Section 4.4.3
- 16: **else** ▷ Sign Search failed
- 17:      $\theta_{\max} \leftarrow \theta_{\max} \times (1 - \kappa)$  ▷ Geometric decay
- 18:     Go to line 3 ▷ Give up

---

**Algorithm 2** Draw direction  $\mathbf{t} \sim \mathcal{T}$ 


---

**Require:** Original image  $\mathbf{x}_o$ , frequency subband  $\mathcal{F}$  s.t.  $|\mathcal{F}| = \rho D$ ,  $A(\cdot)$  shaping function

**Ensure:** A random direction  $\mathbf{t}$  perceptually shaped as  $\mathbf{x}_o$

- 1:  $\mathbf{X}_o = \text{DCT}(\mathbf{x}_o)$
- 2: **for**  $j = 1 : n$  **do**
- 3:     **if**  $j \in \mathcal{F}$  **then**
- 4:          $r \sim \mathcal{U}_{\{-1,0,1\}}$  ▷  $r \in \{-1, 0, +1\}$
- 5:          $T_j = A(|X_{o,j}|) \times r$
- 6:     **else**
- 7:          $T_j = 0$
- 8: **Return**  $\mathbf{t} = \eta(\text{DCT}^{-1}(\mathbf{T}))$  ▷  $\eta(\mathbf{x}) := \mathbf{x}/\|\mathbf{x}\|$

---



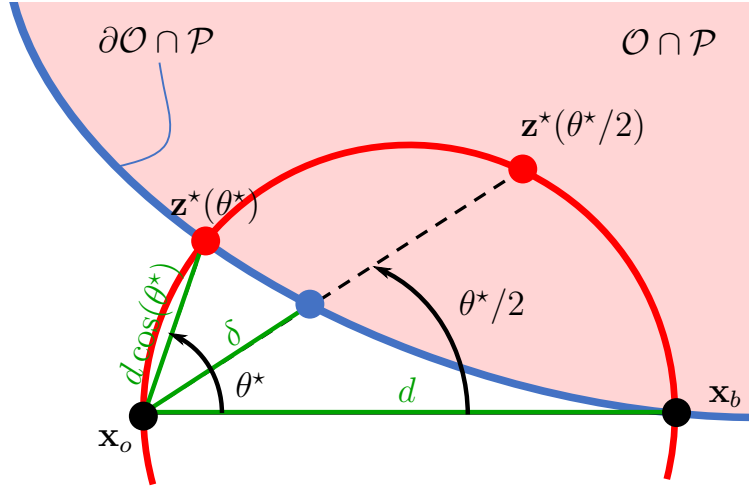


Figure 4.3 – Interpolation mechanism to refine the boundary point.

### 4.5.1 Datasets and Experimental Setup

**Datasets** For MNIST, we use a pre-trained CNN network that is composed of 2 convolutional layers and 2 fully connected Layers. Its accuracy is 99.14%. A subset of 100 *correctly* classified images has been randomly chosen to perform the ablation study. Our attack generates directions on the pixel domain without any dimension reduction.

The ImageNet dataset is tackled by a pre-trained ResNet18, made available for the PyTorch environment [176]. Its top-1 accuracy is 0.6976. We randomly selected  $n = 350$  *correctly* classified images from the ILSVRC2012’s validation set with size  $D = 3 \times 224 \times 224$ .

**Setup and Code** We now detail the specific parameters of *SurFree* [146], for both MNIST and ImageNet. We set empirically the following values in Alg. 1:  $T = 3$ ,  $L = 100$ ,  $\theta_{\max} = 30$ ,  $\kappa = 0.02$ , at most  $\ell = 10$  steps for the binary search (with an early stop if the range is lower than 1% of  $d$ ). We develop *SurFree* [146] on top of the FoolBox library.

**Evaluation Metrics** The two core evaluation metrics are the amount of queries, and the resulting distortion on the attacked image defined. The distortion is measured with the  $\ell_2$  norm and the mean distortion defined in Section 3.1.3. The scores defined in Section 3.1.3 are updated to take into account the number of queries. For a given  $\mathbf{x}_o$ , it is the smallest distortion obtained over the sequence of queries  $(\mathbf{q}_j)_{j=1}^K$  that happen to be adversarial:

$$\text{dist}(K, \mathbf{x}_o) := \min_{1 \leq j \leq K : \text{cl}(\mathbf{q}_j) \neq \text{cl}(\mathbf{x}_o)} \|\mathbf{q}_j - \mathbf{x}_o\|_2 \quad (4.9)$$

The mean over  $n$  original images gives a characteristic of the attack efficiency revealing its capacity to find an adversary close to the original image and especially its speed.

$$\text{dist}(K) := \frac{1}{n} \sum_{i=1}^n \text{dist}(K, \mathbf{x}_{o,i}) \quad (4.10)$$

We define the success rate as the probability of getting a distortion lower than a target  $\text{dist}_t$  within a query budget  $K$ :

$$\text{ASR}(\text{dist}_t, K) := \frac{|\{i \in \llbracket 1, n \rrbracket : \text{dist}(K, \mathbf{x}_{o,i}) \leq \text{dist}_t\}|}{n} \quad (4.11)$$

## 4.5.2 Ablation Studies

**Impact of the components - MNIST** This first ablation evaluates how the hyperplane hypothesis [60] meets practical experimentation, and how the interpolation mechanism of Section 4.4.3 is able to compensate this hypothesis. To this end, four variants of our attack are tested in Figure 4.4 and 4.5: `SignSearch` stops at line 10 of Alg. 1 whereas `SurFree` [146] is the regular attack, ‘+Interpolation’ uses the option 4.4.3.

Our attack is highly random due to the generation of directions. This may yield unstable results with adversarial images of scattered distortion. Figure 4.4 shows the distortion decrease averaged over 100 images and Figure 4.5 the standard deviation for one image attacked 20 times.

This outlines the trade-off between the complexity of one iteration in terms of query number and the gain in the distortion decrease. The Interpolation option may yield a substantial decrease depending on the direction. This explains its large standard deviation. Yet, its costs (2 more binary searches) slow down the speed. `SignSearch` is less costly and offers competitive distortions only at the beginning. `SurFree` [146] strikes the right trade-off both in terms of averaged distortion and standard deviation. Compared to `SignSearch`, it always exhausts the explored direction giving the best gain under the hyperplane boundary assumption. The first important insight is that this hypothesis seems to be good enough to ensure a rapid decay.

The ablation study also tested different values for some parameters of `SurFree` [146]. The value of  $\kappa$  has no significant impact provided that  $\kappa > 0$ . Parameter  $T$  does not benefit from higher value because of the finer search in line 12.

**Impact of the direction generation domain - ImageNet** The literature reports that black box attacks have difficulty in handling large images like ImageNet. Attacks become slow because

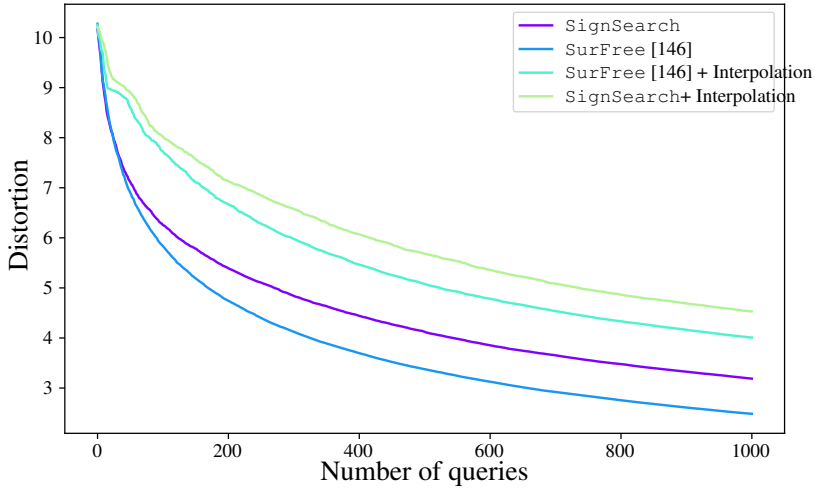


Figure 4.4 – Ablation study on SurFree [146]. Mean distortion  $d(K)$  (4.10) vs. number  $K$  of queries on MNIST.

the space is too large to be explored efficiently. All competing attacks resort to a dimension reduction, typically by leveraging a full DCT transform [126, 187]. Yet, dimension reduction lowers the degrees of freedom for the attacker: the closest adversarial as defined in (4.1) has a bigger distortion under this constraint. The distortion supposedly converges faster but to a bigger limit.

SurFree [146] is no exception. Table 4.1 shows that the distortion in the full pixel domain is bigger within the first thousand queries. For the same query budget, constraining the perturbation to lie in a smaller low-frequency subspace defined with the full DCT as in [126, 187] is beneficial. Yet, this frequency reduction has to be controlled, at the risk of suppressing too many frequencies and obtaining a more important distortion: distortion reported for a reduction of  $\rho = 25\%$  are always larger than those for 50%.

We now question the type of DCT transform. Indeed, while the DCT full frame is widely acclaimed, we prefer the block-based DCT as used in JPEG. It gives a better space-frequency localization trade-off. Table 4.1 shows that it does change the distortions a lot. The 4 last rows of Table 4.1 focus on the adaptivity to the visual content of the original image (see Section 4.4.2). Amplitude function  $A(x) = x$  concentrates the perturbation power too much on some high amplitude coefficients when the original image has sharp edges.  $\tanh(x)$  is a good compromise between the constant and the identity functions. It offers an early distortion drop and reaches similar levels to  $A(x) = cst$  in the long run. Our design is driven by the small query budget requirement so we choose  $\tanh$  and  $\rho = 50\%$  on  $\text{DCT}_{8 \times 8}$ .

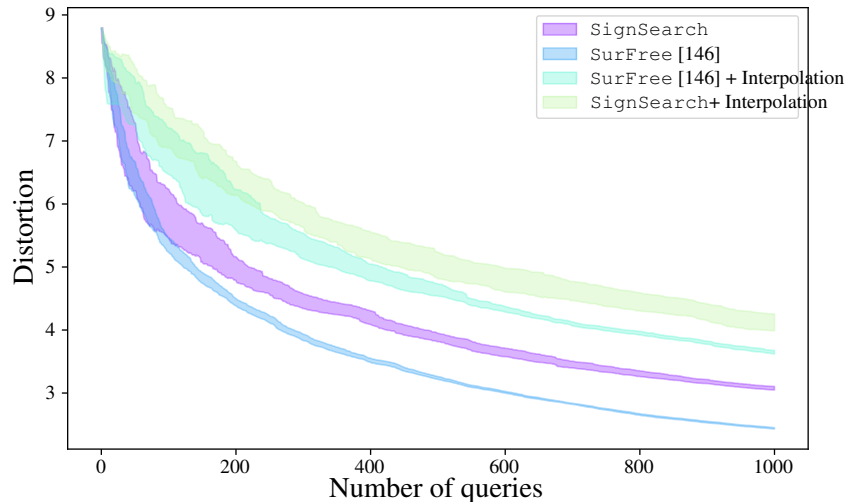


Figure 4.5 – Ablation study on SurFree [146]. The deviation of the distortion over 20 runs of SurFree [146] on one MNIST image.

### 4.5.3 Benchmarking

We compare to recent algorithms considered as state-of-the-art decision-based black box attacks: HSJA [23], GeoDA [187] and QEBA [126]. These 3 algorithms leverage gradient surrogates. The benchmark does not include older attacks like OPT [27] and BA [18] because they have proven less efficient than the three above-mentioned references.

We use the author’s code for these algorithms: HSJA [23] is integrated into the FoolBox library [190, 191]. For GeoDA [187] and QEBA [126], we pull implementations from their respective GitHub repositories<sup>1 2</sup> with default parameters. For GeoDA [187], the number of queries devoted to the gradient estimates follow a geometric progression of common ratio  $\lambda^{-2/3}$  with  $\lambda = 0.6$ , and the dimension reduction focuses on 5,625 coefficients of the full DCT transform. Concerning QEBA [126],  $\rho = 25\%$  dimension reduction on low frequency full DCT coefficients. HSJA [23] works on the pixel domain, the number of queries devoted to gradient estimates scales as  $N_0\sqrt{j}$  with  $j$  the iteration number. We tested two versions with  $N_0 \in \{10, 100\}$ , which is directly observable with the larger plateaus on Figure 4.6.

A very important point is that all attacks are initialized with the same first adversarial example to avoid favouring a competitor by giving it an easier initialization.

1. QEBA: <https://github.com/AI-secure/QEBA>  
 2. GeoDA: <https://github.com/thisisalirah/GeoDA>

| Space               | Shaping $A(x)$   | Dim. Reduc. $\rho$ | $K = 100$    | $K = 1000$  |
|---------------------|------------------|--------------------|--------------|-------------|
| Pixel               | –                | 100%               | 27.23        | 17.20       |
| DCT <sub>full</sub> | <i>cst</i>       | 50%                | 26.50        | 15.35       |
| DCT <sub>full</sub> | <i>cst</i>       | 25%                | 32.08        | 21.23       |
| DCT <sub>8×8</sub>  | <i>cst</i>       | 50%                | 19.49        | 10.69       |
| DCT <sub>8×8</sub>  | <i>cst</i>       | 25%                | 18.26        | <b>9.93</b> |
| DCT <sub>8×8</sub>  | <b>x</b>         | 50%                | 20.11        | 11.96       |
| DCT <sub>8×8</sub>  | <b>x</b>         | 25%                | 20.29        | 12.22       |
| DCT <sub>8×8</sub>  | tanh( <b>x</b> ) | 50%                | <b>17.38</b> | 10.22       |
| DCT <sub>8×8</sub>  | tanh( <b>x</b> ) | 25%                | 18.20        | 10.61       |

Table 4.1 – Mean distortion  $\text{dist}(K)$  when random directions are generated with different subspaces and shaping (ImageNet).

| target<br>dist <sub>t</sub> | $K = 500$ queries |             |            |               | $K = 1,000$ queries |             |            |               | $K = 2,000$ queries |             |            |               |
|-----------------------------|-------------------|-------------|------------|---------------|---------------------|-------------|------------|---------------|---------------------|-------------|------------|---------------|
|                             | HSJA [23]         | GeoDA [187] | QEBA [126] | SurFree [146] | HSJA [23]           | GeoDA [187] | QEBA [126] | SurFree [146] | HSJA [23]           | GeoDA [187] | QEBA [126] | SurFree [146] |
| 30                          | 0.56              | 0.79        | 0.71       | <b>0.90</b>   | 0.88                | 0.93        | 0.88       | <b>0.96</b>   | 0.98                | 0.96        | 0.97       | <b>0.99</b>   |
| 10                          | 0.13              | 0.25        | 0.32       | <b>0.44</b>   | 0.23                | 0.52        | 0.46       | <b>0.57</b>   | 0.40                | 0.70        | 0.69       | <b>0.73</b>   |
| 5                           | 0.07              | 0.14        | 0.17       | <b>0.23</b>   | 0.09                | 0.21        | 0.30       | <b>0.31</b>   | 0.13                | 0.39        | 0.47       | <b>0.50</b>   |

Table 4.2 – Success rate  $\text{ASR}(\text{dist}_t, K)$  for achieving a targeted distortion  $\text{dist}_t$  under a limited query budget  $K$  (ImageNet).

**Performance evaluation: distortion vs. queries** Figure 4.6 displays the distortion of the perturbation ( $\ell_2$  norm) versus the number of queries. *SurFree* [146] presents a smooth curve, resulting from averaging over 350 images. Even with this averaging, the other attacks still show large plateaus (as highlighted in Figure 4.1 for one image) because gradient estimates are scheduled at the same instants for any image. Note that these plateaus are not shown in the papers because the distortion is seen as a function of the iteration number, not the query number. The two most recent attacks, *QEBA* [126] and *GeoDA* [187] indeed beat *HSJA* [23] as reported in the corresponding papers. *SurFree* [146] dives significantly faster than all attacks to lower distortions (notably from 1 to 750 queries), while *QEBA* [126] prevails at around 3,750 queries. Note that *SurFree* [146] is also first with DCT full but for a shorter period ( $\approx 800$  queries). For completeness, here are the scores at 10,000 queries: 2.09 (*QEBA* [126]) < 2.72 (*SurFree* [146]) < 3.48 (*HSJA\_10*) < 4.63 (*GeoDA* [187]). Although a small query budget drives its design, *SurFree* [146] is not off in the long run. Similar results are observed for MNIST (pixel domain, without dimension reduction) where *SurFree* [146] is ahead up to  $\approx 5,000$  queries.

**Performance evaluation: Success rate** We now consider three query budgets,  $K \in \{500, 1,000, 2,000\}$ , which are rather low with regards to the state-of-the-art.

| attack        | $K = 100$         | $K = 500$         | $K = 1000$        | $K = 100$        | $K = 500$       | $K = 1000$      | $K = 100$            | $K = 500$            | $K = 1000$        |
|---------------|-------------------|-------------------|-------------------|------------------|-----------------|-----------------|----------------------|----------------------|-------------------|
| SurFree [146] |                   |                   |                   |                  |                 |                 |                      |                      |                   |
|               | amer. dipper- 2.6 | amer. dipper- 1.3 | amer. dipper- 0.9 | stone wall- 14.9 | stone wall- 8.7 | stone wall- 5.4 | cliff dwelling- 21.9 | cliff dwelling- 18.4 | triceratops- 13.5 |
| QEBA [126]    |                   |                   |                   |                  |                 |                 |                      |                      |                   |
|               | stingray- 60.6    | stingray- 33.7    | stingray- 20.8    | stone wall- 25.2 | stone wall- 4.8 | stone wall- 2.6 | wombat- 58.3         | wombat- 24.3         | wombat- 13.6      |
| GeoDA [187]   |                   |                   |                   |                  |                 |                 |                      |                      |                   |
|               | brambling- 18.9   | brambling- 9.7    | brambling- 5.8    | stone wall- 15.8 | megalith- 4.5   | megalith- 2.6   | armadillo- 49.4      | tusker- 31.3         | tusker- 18.9      |

Table 4.3 – Visual trajectories for an easy (chickadee), a medium (king penguin), and a difficult image (warthog) - predicted label and distortion

Table 4.2 details how the success rate  $asr(\text{dist}_t, K)$  varies for some setup  $(\text{dist}_t, K)$  (4.11). Figure 4.7 shows the success rate  $ASR(\text{dist}_t, 500)$  increase with  $\text{dist}_t$ . GeoDA [187] is superior to QEBA [126] for large target distortions only. Both schemes outperform HSJA [23]. SurFree [146] remains the best attack for any target distortion up to this 2,000 query budget.

Finally, Table 4.3 displays the visual trajectories of three attacked images witnessed as easy, medium, and difficult to attack for SurFree [146]. While all three attacks affect differently the images, SurFree [146] gives relatively less annoying artefacts. We also note a drawback of QEBA [126]: the adversarial often keeps the label of the random starting point (*e.g.* stingray), hence sometimes converging to a local minimum which is far from the optimal solution (4.1).

## 4.6 Conclusion

The performance of black box decision-based attacks reveals important gaps when it comes to the required amount of queries. Core to the three state-of-the-art approaches these papers consider is the estimation of gradients. This step is particularly costly, with regards to our novel geometrical attack SurFree [146]. The trial of multiple directions together with a simple mechanism getting the best distortion to decrease along a given direction allows a fast convergence to qualitative adversarial, within an order of hundreds of queries solely. This sets a new stage for future works.

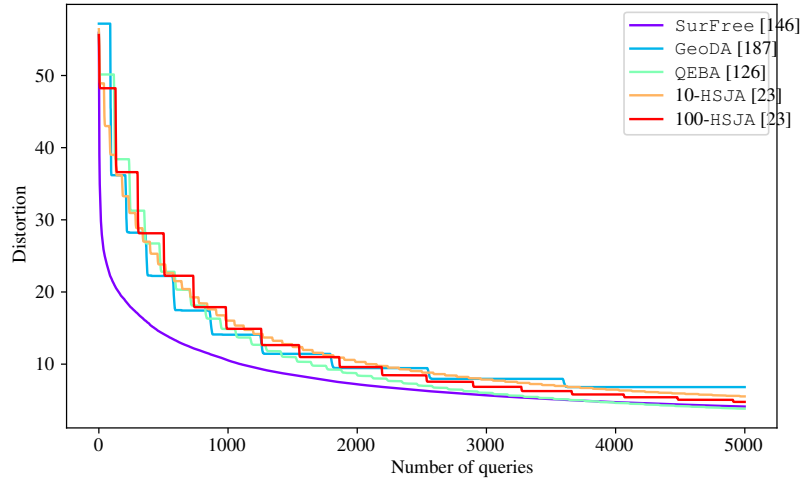


Figure 4.6 – Benchmark on ImageNet. The amount of queries  $K$  ( $x$ -axis) w.r.t. mean distortion  $\text{dist}(K)$  ( $y$ -axis).

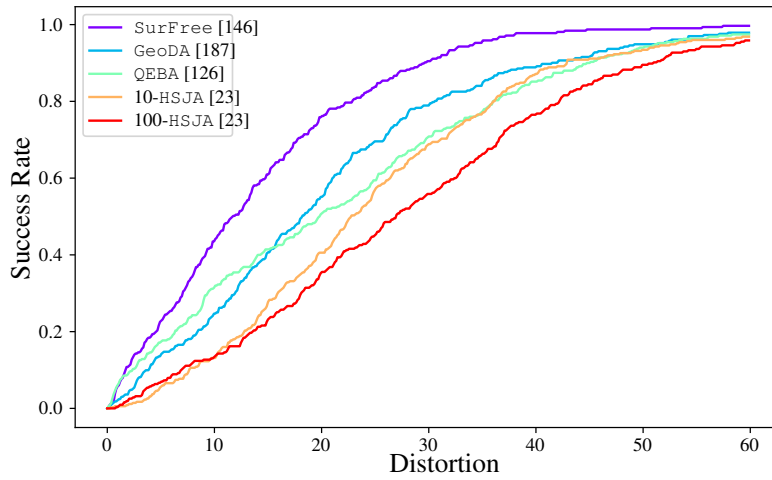


Figure 4.7 – Success rate  $\text{ASR}(\text{dist}_t, K)$  (4.11) vs. target distortion  $\text{dist}_t$  with  $K = 500$  queries over ImageNet.

# HOW TO CHOOSE YOUR BEST ALLIES FOR A TRANSFERABLE ATTACK?

---

## 5.1 Introduction

Transferability is usually measured by the Attack Success Rate (ASR), *i.e.*, the probability that the adversarial example crafted for the source model also deludes the target model. We argue that this measure leads to an unfair evaluation of transferability. In the context of adversarial examples, it is not just a matter of discovering data that is not well classified, but rather identifying the perturbation that can fool a classifier with minimal distortion. This principle should also apply to transferable attacks.

For illustration purposes, let us consider two models, one is robust in the sense that the necessary amount of adversarial perturbation is large, whereas the other model is weak. If the attacker uses the robust model as the source to attack the weak target network, the ASR of the transferable attack will certainly be big. It does not mean that this is the right choice. The ASR is high because the robust source model needs large perturbation to be deluded, which will fool any weaker model. The ASR alone does not reflect the overshooting in distortion. The converse, using the weak to attack the robust, would yield a low ASR. To summarize, the ASR fails to capture the relevance of the *perturbation direction* given by the source for attacking the target.

The first contribution of this chapter is to put distortion back into the picture. Section 5.2 evaluates transferability by comparing the distortion of a transferable attack to the ones of two reference attacks: On one hand, the strongest attack, *i.e.*, the white box attack directly applied on the target model; on the other hand, the weakest attack, *i.e.*, the black box attack.

The second contribution shows the great variability of the performance of transferable attacks. Figure 5.1 summarizes this observation by plotting the ASR as a function of the distortion (the experimental protocol is explained in Section 5.3.1). Naturally, the black box attack needs much more distortion than the white box attack. For instance, the white box attack yields an ASR of 50% with a distortion of 0.19, whereas the black box attack needs a distortion of 9.7.



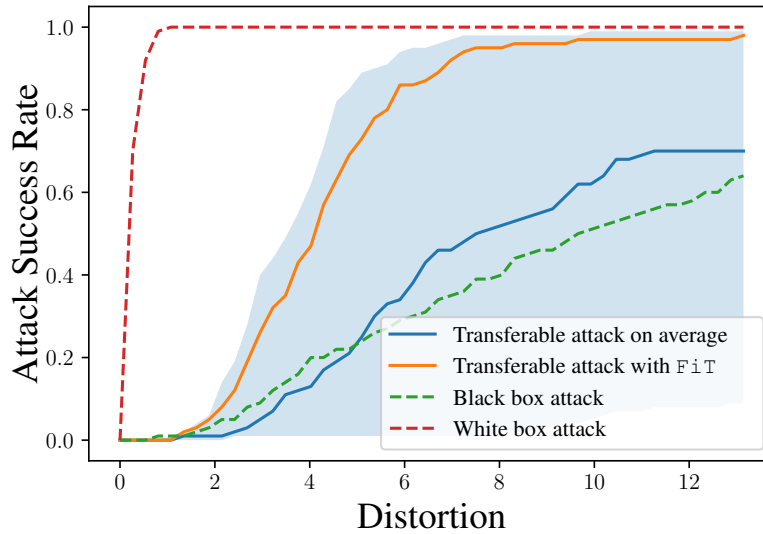


Figure 5.1 – Evaluation of transferability by comparing the Attack Success Rate vs. distortion trade-off of a white box, transferable, and black box attacks against model  $\text{CoatLite}_{\text{small}}$  (See Section 5.3.1 for details). The blue area is the range of trade-offs operated by a transferable attack with random source models. A transferable attack may be worse than a black box attack without a good source selection (like  $\text{FiT}$ ).

The surprise is that if the attacker resorts to a transferable attack and picks a source model at random, there is almost a 50% chance that the attack performs even worse than the black box attack. Section 5.3 outlines a triad of factors: input, source model, and attack.

All this literature uses the ASR as the figure of merit for a given distortion while this chapter draws the full operating characteristic of ASR vs. distortion. Moreover, the performance of the transferability is compared neither to the white box nor the decision-based black box attack.

This observation challenges the prevalent notion that adversarial examples transfer easily between models, and highlights the need to carefully choose the source model to attack a target. Under the assumption that the attacker has indeed several candidate models, our third contribution, named  $\text{FiT}$  in Section 5.4, provides an affordable measure for model selection, allowing the attacker to choose a good source model with only a few queries to the target.

*All the notations unique to this chapter are listed in Table 7.13 at the conclusion of this manuscript. Additionally, for general notation, please refer to Table 7.9.*

## 5.2 Methodology

This chapter introduces a new way to gauge transferability. The main motivation is that transferability should indicate whether a source  $s$  is useful for attacking a target  $t$ , independently from the inherent robustness of  $t$ .

### 5.2.1 Notations

This chapter follows the general notation in Section 1.2.3. For a given input  $x$  of class  $y$ , the adversarial example  $x_a$  is the result of an attack (be it white box, black box, or by transferability) such that

$$\arg \max_{1 \leq k \leq C} f_k(x_a) \neq \arg \max_{1 \leq k \leq C} f_k(x) = y. \quad (5.1)$$

In the case of transferable attacks, the attacker uses a source model  $f_s$  to build a transferable adversarial example against the target model  $f_t$ . This chapter considers a collection of  $m$  source models denoted by  $\mathcal{F}_s = \{f_s^1, \dots, f_s^m\}$ , and a set of  $n$  inputs  $\mathcal{X}$ .

### 5.2.2 Measurement

### 5.2.3 Transferability

Given a white box and a black box attacks, The proposed methodology first computes the operating characteristic  $P_t^{\text{wb}}(D)$  of a state-of-the-art white box attack directly applied to the target model, and the operating characteristic  $P_t^{\text{bb}}(D)$  of a state-of-the-art black box attack. It then measures where the operating characteristic of the transferable attack from the source  $s$  to the target  $t$  lies in between the two characteristics as follows:

$$T_{s,t} := \frac{\int_0^\infty P_{s,t}(u) - P_t^{\text{bb}}(u) du}{\int_0^\infty P_t^{\text{wb}}(u) - P_t^{\text{bb}}(u) du}. \quad (5.2)$$

The proposed score is calculated as the ratio of the areas between the different operational characteristics, which are defined as Cumulative Distribution Functions (CDFs). The numerator is therefore close to the 1-Wasserstein distance [43] between the ASR function of the distortions of the transferable attack and that of the black box attack, where the absolute value is removed to obtain a signed score. If the transferable attack performs as well as the white box attack (resp. as bad as the black box attack), then  $T_{s,t} = 1$  (resp.  $T_{s,t} = 0$ ). Figure 5.1 shows that the numerator of (5.2) can indeed be negative as transferability can be even worse than the black box if the

source  $s$  is not well chosen. Therefore, zero is not a lower bound for  $T_{s,t}$ . In the denominator, the 1-Wasserstein distance is obtained exactly between the white box and the black box because the white box consistently generates adversarial examples with lower distortion.

## 5.2.4 Practical implementation

**Attacks.** We only consider state-of-the-art attacks which almost surely deliver an adversarial example. As said in Section 3.4.1, this is usually the case for black box attacks, but we are limited to using white box attacks that are *not* subject to a distortion budget. As for transferability, the attack uses the publicly available model  $s$  and input  $x$  to craft an adversarial direction  $u_{x,s}$  of Euclidean norm  $\sqrt{N}$ . The use of  $\sqrt{N}$  for the norm is for the sake of simplicity in notation. We assume that there is an oracle giving the minimal distortion along this direction to fool the target  $t$ . In other words,  $x_a = x + d u_{x,s}$ , with

$$d = \min\{\delta : \arg \max_k f_{t,k}(x + \delta u_{x,s}) \neq y\}. \quad (5.3)$$

Note that  $\text{dist}(x_a, x) = d$ . This definition favours transferability as its best. In practice, such an oracle does not exist but the attacker finds a good estimate of (5.3) thanks to a line search within a few queries to the target.

**Transferability.** We run a given attack over a collection  $\mathcal{X}$  of  $n$  inputs correctly classified by the target model. We compute the distortions  $d(j) = \text{dist}(x_{a,j}, x_j)$  with  $x_j \in \mathcal{X}$  and sort them so that  $d(1) \leq d(2) \leq \dots \leq d(n)$ . We set  $d(0) = 0$  and  $d(n+1) = \infty$  to properly define the empirical operating characteristic by the following step function:

$$\hat{P}(D) := j/n \quad \forall D \in [d(j), d(j+1)). \quad (5.4)$$

It is then easier to estimate the integrals appearing in (5.2), which are indeed areas under two curves, by a Lebesgue sum rather than a Riemann sum. This gives:

$$\hat{T}_{s,t} := \frac{\sum_{j=1}^n d_{s,t}(j) - d_t^{\text{bb}}(j)}{\sum_{j=1}^n d_t^{\text{wb}}(j) - d_t^{\text{bb}}(j)}, \quad (5.5)$$

where the distortions  $(d_t^{\text{bb}}(j))_j$  (resp.  $(d_t^{\text{wb}}(j))_j$ ) resulting from the black box (resp. white box) attack against model  $t$  are also sorted in increasing order.

## 5.3 Triad of transferability: data, model, attack

This section is an experimental investigation of the factors affecting transferability.

### 5.3.1 Experimental setup

The study assesses transferable attacks on a total of 48 models, with 47 of them sourced from the Timm library [258] and one very robust, namely `ResNet50AdvTrain`, obtained from the GitHub repository<sup>1</sup>. Section C.1 lists all the models. The experiments utilize 100 images from the validation set of ILSVRC’12, all of which are correctly classified by all models under consideration.

The study considers three transferable attacks - DI [265], TAIG [96], and DWP [244]- each using a different approach to improve transferability (see Section 3.4.1). These attacks are selected as the best in their categories in [287]. They all share an  $\epsilon$  parameter to control the maximum perturbation added per pixel. The effect of the parameter  $\epsilon$  indeed happens to be negligible in our protocol, as demonstrated in Section C.2.1. We choose  $\epsilon = 8$ .

To measure transferability (5.5), we need state-of-the-art black box and white box attacks. Certain methods may exhibit a preference towards one model over the other, necessitating the use of multiple attacks. Our study employs four white box attacks (BP [280, 15], DeepFool [156], I-FGSM [117], and PGD [142]) and three black box attacks (SurFree [146], RayS [24], and GeoDA [187]). All black box attacks are run with 2,000 queries, which has been determined to be sufficient for achieving convergence. For distortion-constrained white box attack, the  $\epsilon$  parameter is set to 4. We record the smallest perturbation distortion over the black box (white box) for each image and draw the operating characteristic (5.4) as appearing in green (resp. red) dashed line in Figure 5.1.

As for the transferable attack, for a given target model, the attacker has access to a subset of all the other models whose architecture differs from the target. This amounts to an average of 45 models out of 48. For instance, in Figure 5.1, `CoatLitesmall` being the target, we exclude all other `CoatLite` models from being a source. The light blue area delimits the operating characteristics of transferable attack DI [265] using as the source one of the 45 remaining models.

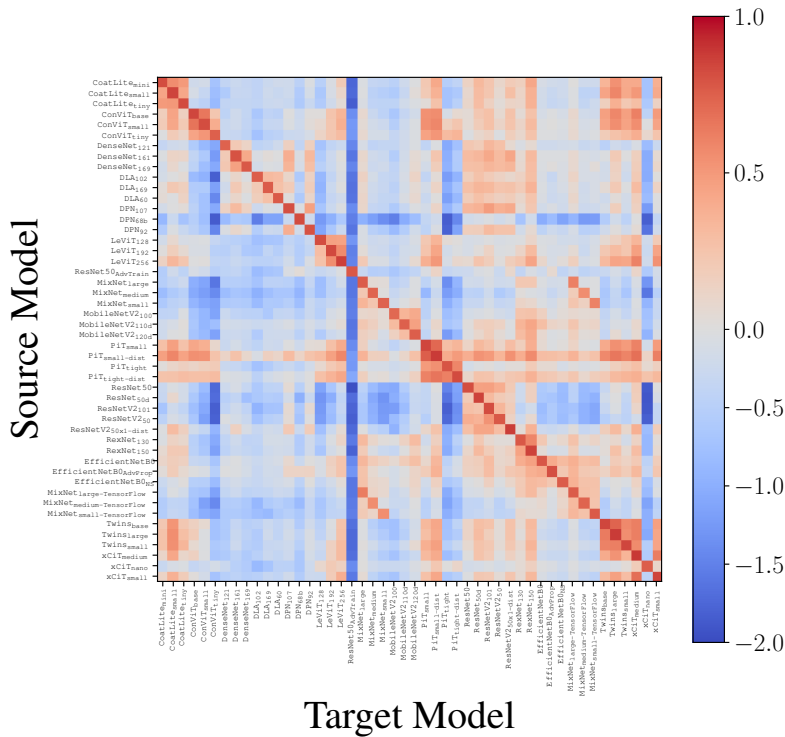


Figure 5.2 – Transferability score  $\hat{T}_{s,t}$  matrix of 48 sources and 48 targets listed in App. C.1 with attack DI [265].

### 5.3.2 Model dependence

**Large transferability variation.** Figure 5.2 shows the matrix  $\hat{T}_{s,t}$ , where  $s$  and  $t$  are any of the  $48^2$  pairs, for attack DI [265]. Not all models possess the same transferability capabilities. At first look, the figure contains more blue than red cells which means that transferability takes a negative value more often.

Some rare models, like  $\text{PiT}_{\text{small-dist}}$ , exhibit good transferability towards any target. On the contrary,  $\text{DPN68b}$  is always a bad source. On the other hand,  $\text{ResNet50}_{\text{AdvTrain}}$  is a very difficult target (note however that its accuracy is low), followed by  $\text{PiT}_{\text{light}}$  and  $\text{ConViT}_{\text{tiny}}$ , whereas the family of  $\text{ResNet}$  are a fairly easy target.

Prior works have shown that models with architectures similar to the target transfer better [261, 180]. The red squares appearing in the matrix confirm this (models are ordered according to architecture family). For instance,  $\text{MobileNetV2}$  and  $\text{ResNet}$  architectures exhibit transferability close to 0.7. However, this is not an absolute truth.  $\text{EfficientNet-B0}$  has better transferability against  $\text{MixNet}_{\text{large}}$  than  $\text{MixNet}_{\text{medium}}$ , even though they have sim-

1. <https://github.com/MadryLab/robustness>

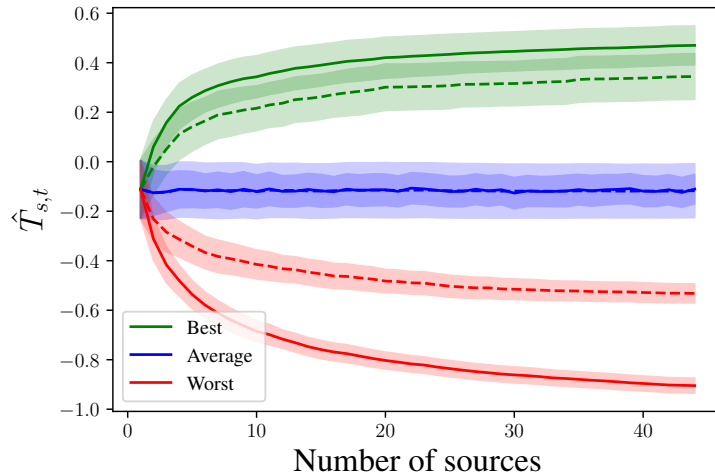


Figure 5.3 –  $\hat{T}_{s,t}$  function of the number of available sources and attack DI [265]. The best or worst model selected per image (solid line) or on average (dashed line).

ilar architectures. ConViT transfers exceptionally well to Twins although their architectures are very different. Similar observations hold for the other attacks but with lower transferability scores (see Appendix C.3).

From now on, we exclude models whose architecture is similar to the target.

**Impact on the attack.** Selecting the right source among several available models is critical for achieving high transferability. Figure 5.3 displays the transferability score (5.5) for the best and the worst choices (shown in dotted lines) as the number of source candidates increases. In this evaluation, a single model is selected as the source for building all adversarial examples. On average, the transferability is lower than zero, regardless of the attack method. This means that transferable attacks perform worst than black box attacks on average. If the attacker knows how to select the best source model,  $\hat{T}_{s,t}$  quickly converges to a maximum which is positive but below 0.5. This means that transferability at its best performs closer to a black box attack rather than a white box attack. These remarks highly mitigate the threat of transferability and put emphasis on the crucial selection of the best source model. As far as we know, these facts are not reported in the adversarial examples literature.

### 5.3.3 Image dependence

The difficulty of transferring adversarial examples from a source to a target varies significantly depending on the input  $x$ . This is illustrated in Figure 5.4, which shows the distribution

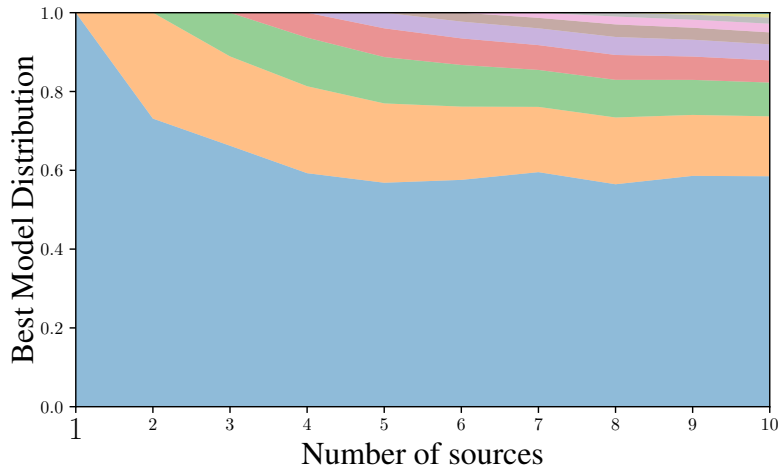


Figure 5.4 – Distribution of best image-model selection function of the number of sources. Adversarial examples obtained with DI [265].

of the best-performing model for each image based on the available sources. Even with a large number of sources available, there is typically one source that performs significantly better than the others, but only over 60% of the images on average. However, this superiority decreases rapidly as the number of available sources increases. It means that a better source exists for 40% of the images on average.

Supposing that the attacker knows the best source of each input, Figure 5.3 shows in solid line that the transferability converges to a value close to 0.5. The performance of the transferable attack lies halfway between the ones of the white box and black box attacks. This highlights the importance of selecting an appropriate source model for a given target and input.

### 5.3.4 Attack dependence

**Transferable attack.** Figure 5.5 compares transferable attacks with a 2D histogram of the distortion pair for two attacks. This is computed over all inputs in  $\mathcal{X}$  and all pairs of source and target models. The results show that DWP [244] exhibits poor transferability compared to DI [265] and TAIG [96], which produce adversarial perturbations with similar distortion. Additionally, regardless of the attack complexity or method, the challenging images remain consistent. If one attack requires a high distortion for a given input / source / target, other attacks are likely to encounter similar difficulties.

**Traditional white box attack.** We now compare the methods designed for transferability with a naive approach. A white box attack (ie. not specific to transferability) is executed on

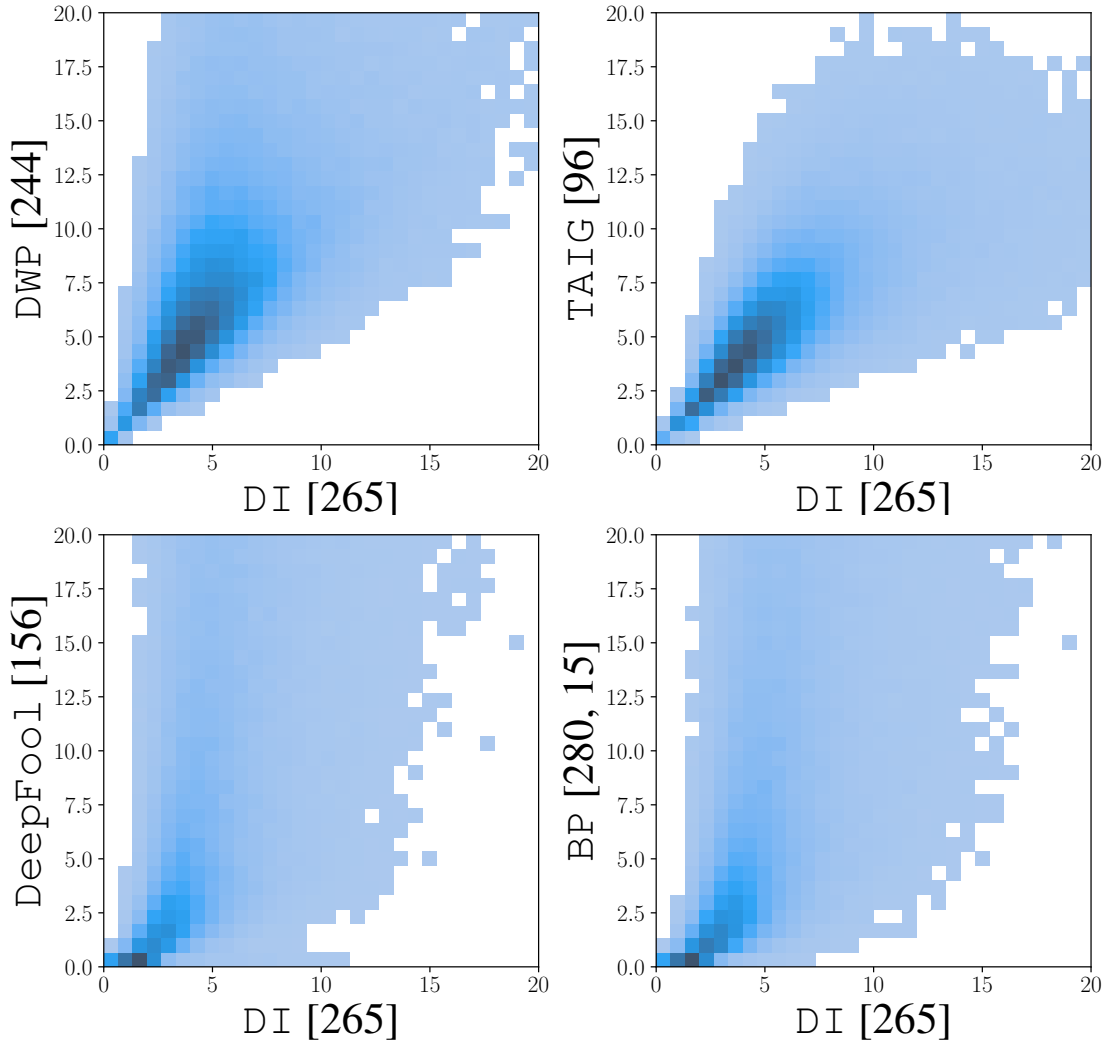


Figure 5.5 – 2D Histogram of the minimum distortion for transferable and white box attacks.

the source, and then the direction found serves as  $u_{x,s}$  to place the adversarial examples on the target boundary with (5.3). The best  $\hat{T}_{s,t}$  score is 0.27 for BP [280, 15] compared to 0.52 for DI [265]. This confirms the superiority of the recent methods designed for transferability. However, even though DI [265] is on average better, Figure 5.5 shows that when the necessary distortion is small, traditional white box attacks like DeepFool [156] and BP [280, 15] indeed beats DI [265]. On the other hand, DI [265] performs better for inputs requiring more distortion.



## 5.4 How to choose the best source

Section 5.3 outlines that the choice of the source is of utmost importance thanks to the transferability measure defined in Section 5.2. This section investigates whether the attacker can guess which model is the best source.

We now propose a procedure which combines the dependences with respect to the source and target models (Section 5.3.2), and the input (Section 5.3.3). Model dependence is measured by evaluating the similarity between the source and target models, which is denoted as  $\text{ModSim}$ . On the other hand, the image dependence is measured by the quality of the adversarial example, denoted as  $\text{TransQ}$ . Both metrics are combined into the following score:

$$\text{FiT}(s, t, \mathbf{x}) := \text{ModSim}(s, t) \times \text{TransQ}(s, \mathbf{x}). \quad (5.6)$$

These indicators should be easy to compute. We especially pay attention to the number of queries to the target. This score opens the door to a new strategy for the attacker which first selects the best source among the available models

$$s^*(t, \mathbf{x}) = \arg \max_{\sigma \in \mathcal{F}_s} \text{FiT}(\sigma, t, \mathbf{x}), \quad (5.7)$$

and then crafts the adversarial direction  $u_{\mathbf{x}, s^*(t, \mathbf{x})}$ .

### 5.4.1 Criterion $\text{ModSim}(s, t)$

Section 5.3.2 highlights the correlation between the transferability and the similarity between the source and target models. Gauging model similarity has been previously studied in the context of fingerprinting as a defense to protect intellectual property (see Section 2.3.1). This chapter uses fingerprinting methods as an attack that leaks information about the target.

We consider the fingerprinting method [147] because it works in a decision-based setup since the target is a black box in our application. Querying two models  $s$  and  $t$  with few natural images, it computes a distance  $\text{Dist}(s, t) \in [0, 1]$  by comparing their outputs. Since we look for a similarity, we set  $\text{ModSim}(s, t) = 1 - \text{Dist}(s, t)$ . However, this provides a symmetrical similarity, ie.  $\text{ModSim}(s, t) = \text{ModSim}(t, s)$ , while transferability is not (see Figure 5.2). This shows that this criterion alone is not sufficient.

### 5.4.2 Criterion $\text{TransQ}(s, x)$

This criterion evaluates the general transferability of a given adversarial example crafted by a source. Our idea is to leverage the assumption that the attacker has a set of models  $\mathcal{F}_s$ . Consequently, we can evaluate the transferability thanks to the other models of this set, without querying the target.

For a given input  $\mathbf{x}$ , the source  $s$  provides the adversarial direction  $u_{\mathbf{x},s}$  and we compute the distortion  $d_{s,\sigma}$  necessary to delude classifier  $\sigma \in \mathcal{F}_s$  with (5.3). We then aggregate these distortions into a single score with two flavours:

$$\text{TransQ}^{(1)}(s, \mathbf{x}) := \left( \frac{1}{|\mathcal{F}_s|} \sum_{\sigma \in \mathcal{F}_s} d_{s,\sigma} \right)^{-1}. \quad (5.8)$$

A good source for input  $\mathbf{x}$  gives birth to lower distortions, so that  $\text{TransQ}^{(1)}(s, \mathbf{x})$  is large.

$$\text{TransQ}^{(2)}(s, \mathbf{x}) := \frac{\sum_{\sigma \in \mathcal{F}_s} d_{s,\sigma} - d_{\sigma}^{\text{bb}}}{\sum_{\sigma \in \mathcal{F}_s} d_{\sigma}^{\text{wb}} - d_{\sigma}^{\text{bb}}}. \quad (5.9)$$

This measure is similar to (5.5) except that it is computed over the set of models instead of a set of inputs.

### 5.4.3 Results

The experimental setup is the same as in Section 5.3.1. We select the fingerprinting method FBI [147] with 200 benign natural images to compute the criterion  $\text{ModSim}(s, t)$ . It implies that the attacker first makes 200 queries to the target in a preliminary step before forging any adversarial example. Appendix C.4.1 shows that more images improve the fingerprinting accuracy hence the model selection, but the results converge after 200 images.

This section is structured as follows: we use `FiT` to select the best model for single-model attacks, and then we employ it to identify the best subset of models for ensemble-model attacks and different combinations of attacks.

#### Single-model attacks

**Criterion**  $\text{ModSim}(s, t)$ . Table 5.1 indicates that architectural similarity is a reliable measure of transferability between two models. It can drive the selection of a good source giving birth to a transferable attack outperforming the black box attack since  $\hat{T}_{s,t}$  is larger than 0 (except

Table 5.1 – Transferability  $\hat{T}_{s,t}$  for DI [265], TAIG [96] and DWP [244] for single and ensemble-model attacks.

| Category                                    | Selection Method |                             | DI [265]    | TAIG [96]   | DWP [244]   |
|---|------------------|-----------------------------|-------------|-------------|-------------|
| Single-model attack                         | Best             |                             | 0.52        | 0.46        | 0.34        |
|   | Random           |                             | -0.16       | -0.12       | -0.72       |
|   | ModSim           | FBI [147]                   | 0.18        | 0.12        | -0.39       |
|   | TransQ           | <i>ASR</i>                  | -0.21       | -0.24       | -1.14       |
|   |                  | TransQ <sup>(1)</sup> (5.8) | 0.38        | 0.24        | 0.10        |
|   |                  | TransQ <sup>(2)</sup> (5.9) | 0.37        | 0.23        | 0.08        |
|   | FiT              | TransQ <sup>(1)</sup> (5.8) | <b>0.40</b> | <b>0.27</b> | <b>0.12</b> |
| TransQ <sup>(2)</sup> (5.9)                 |                  | 0.39                        | 0.25        | 0.10        |             |
| Ensemble-model attack<br>with three sources | Best             |                             | 0.72        | 0.62        | 0.46        |
|   | Random           |                             | 0.43        | 0.40        | 0.02        |
|   | ModSim           | FBI [147]                   | 0.59        | 0.49        | 0.05        |
|   | TransQ           | <i>ASR</i>                  | 0.53        | 0.45        | 0.06        |
|   |                  | TransQ <sup>(1)</sup> (5.8) | 0.62        | 0.54        | 0.33        |
|   |                  | TransQ <sup>(2)</sup> (5.9) | 0.61        | 0.54        | 0.33        |
|   | FiT              | TransQ <sup>(1)</sup> (5.8) | <b>0.64</b> | 0.55        | 0.35        |
| TransQ <sup>(2)</sup> (5.9)                 |                  | <b>0.64</b>                 | <b>0.57</b> | <b>0.36</b> |             |

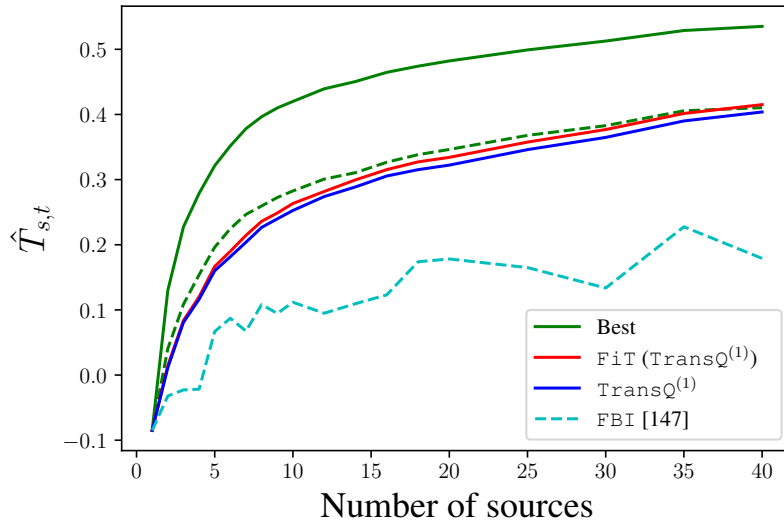


Figure 5.6 –  $\hat{T}_{s,t}$  as a function of the number of available sources for several selection methods. Dotted lines refer to the selection of a unique model for all images and solid lines refer to a model selected per image. Attack is DI [265].




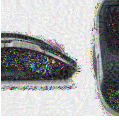
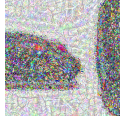




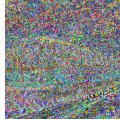
for DWP [244]). Yet, the results remain low compared to the best results obtained. As discussed in Sec 5.3.2, similarity may not suffice because it implies the selection of a unique source for a given target. Better results are achieved when adapting the source to the input.

**Criterion**  $\text{TransQ}(s, x)$ . Improving transferability without querying the target model in a preliminary step is possible thanks to  $\text{TransQ}(s, x)$ . Adversarial examples that exhibit good transferability on multiple models are more likely to also deceive the unknown targeted model. Figure 5.6 shows that a significant improvement in transferability is achieved even with only a few available models. Table 5.1 confirms this observation for the two other attack methods. This strategy is indeed better than the selection based on model similarity.

**Score**  $\text{FiT}(s, t, x)$ . Combining both criteria together as in (5.6) leads to a slight improvement in transferability compared to  $\text{TransQ}(s, x)$  alone. Figure 5.6 confirms this holds over a wide range of numbers of available sources. For single-model attacks,  $\text{TransQ}^{(1)}$  gives slightly better results than  $\text{TransQ}^{(2)}$ .

**Visual results** Table 5.2 visually demonstrates the impact of different model selection methods on the quality of adversarial examples. Even when  $\text{FiT}$  is not accurate, the resulting adversarial examples are still close to the best ones obtained from the source models, and the perturbation remains imperceptible. However, random selection generates noisy perturbations, and the worst-case scenario destroys the image entirely.

Table 5.2 – Visual impact of the source selection with DI [265] attacking  $\text{ConViT}_{\text{base}}$  (first row) and  $\text{DPN}_{92}$  (second row).

|            | Original  | Best  | FiT (TransQ <sup>(1)</sup> )  | Random   | Worst   |
|------------|---|---|---|--|---|
|            |  |  |  |  |  |
| label      | mouse   | oil filter  | purse   | purse  | jigsaw puzzle   |
| source     |   | $\text{PiT}_{\text{small-dist}}$  | $\text{ConViT}_{\text{base}}$   | $\text{MobileNetV2}_{110d}$  | $\text{DenseNet}_{121}$   |
| distortion |   | 5.62  | 8.13  | 35.9   | 87.8  |
|            |  |  |  |  |  |
| label      | tram  | elec. locomotive  | elec. locomotive  | racing car   | jigsaw puzzle   |
| source     |   | $\text{ResNetV2}_{50x1\text{-dist}}$  | $\text{PiT}_{\text{small-dist}}$  | $\text{DLA}_{60}$  | $\text{MixNet}_{\text{medium}}$   |
| distortion |   | 6.94  | 7.50  | 33.3   | 128.7   |

## Ensemble-model attacks

**Importance of model selection in ensemble-model attack.** Ensemble attacks remain an under-studied area due to the significant computational resources required for evaluating attacks. Consequently, these attacks have only been evaluated with a limited number of sources. Section C.4.2 shows that increasing the number of sources is not necessarily beneficial. The FiT score provides a scalable solution: we select a small subset of sources based on their FiT scores (top-3) from the bigger set of available sources. In a way, it is better to put quality above quantity. For example, when running an ensemble-model attack against  $\text{xCiT}_{\text{nano}}$ , selecting only the three best models using the FiT measure can lead to a significant improvement in transferability compared to using a larger set of models. In our experiments, an ensemble-model of 20 random models achieved a  $\hat{T}_{s,t}$  of 0.47, while an ensemble-model of only three models selected with FiT was able to achieve a  $\hat{T}_{s,t}$  of 0.56 against the same target.

**Performance of ensemble-model attacks.** Table. 5.1 shows that ensemble-model transferability surpasses that of single-model attacks. While the average results over a random selection of 3 sources increase, they remain closer to the black box results than the white box ones (transferability lower than 0.5). Choosing the top-3 sources returned by our scores for leading the ensemble-model attack yields better performance, comparable to the best possible results obtainable with ensemble-model attacks. Notably, the DI [265] and TAIG [96] methods approach the performance of white box results (transferability greater than 0.5).

### White box vs. transferable attacks

Our last result is that the selection of a single source with the `FiT` score does not make traditional white box attacks transferable. Table 5.3 shows that `BP` [280, 15], `DeepFool` [156], `PGD` [142], and `I-FGSM` [117] yield negative transferability values. More precisely, `BP` [280, 15], `I-FGSM` [117], and `PGD` [142] perform better with `FiT` than the transferable attacks without any selection mechanism (ie. on average with a random source). However, Section 5.3.4 highlights that the traditional white box attacks may be competitive for some inputs demanding low adversarial perturbation distortion. It is valuable to add them to the options of the attacker and let the `FiT` score decide the preferable option (source and attack method). This provides our best transferability score for a single model attack, close to 0.5.

## 5.5 Conclusion

Transferability is a crucial feature of adversarial examples as it allows a single perturbation to deceive multiple models. However, solely relying on Attack Success Rate (ASR) to measure transferability overlooks the degree of distortion needed to fool a model. This chapter introduces a novel approach to assess transferability by comparing it to the distortion of two reference attacks: white box and black box attacks. We show that transferable attacks can perform worse than black box attacks without an appropriate selection of the source model, highlighting the need to choose the best source model to target a specific model.

The proposed solution, named `FiT`, allows the attacker to choose one of the best source models with minimal queries to the target. Our experiments demonstrate that the proposed solution performs well in multiple attack scenarios.

This study has highlighted the differences in transferability between images for the same source model and their particularity for this specific network. Further research could focus on addressing this issue and investigating its underlying causes, with the hope of designing an even better selection mechanism able to spot the best source model.

Table 5.3 – Transferability  $\hat{T}_{s,t}$  for white box and transferable attacks (single-model). The FiT score uses  $\text{TransQ}^{(1)}$ .

| Attack       |                | Best | FiT         |
|--------------|----------------|------|-------------|
| White Box    | BP [280, 15]   | 0.27 | -0.06       |
|              | DeepFool [156] | 0.03 | -0.34       |
|              | PGD [142]      | 0.29 | -0.01       |
|              | I-FGSM [117]   | 0.29 | -0.03       |
| Transferable | DI [265]       | 0.52 | 0.40        |
|              | TAIG [96]      | 0.46 | 0.27        |
|              | DWP [244]      | 0.34 | 0.12        |
| All Attacks  |                | 0.65 | <b>0.48</b> |

# RANDOMIZED SMOOTHING UNDER ATTACK: HOW GOOD IS IT IN PRACTICE?

---

## 6.1 Introduction

After the contributions on adversarial attacks and the forge of adversarial examples, let's delve into the defense.

To defend against adversarial examples, many techniques have been proposed as shown in Section 3.5. A novel proposal is to *certify* the robustness of classifiers, within particularly randomized smoothing [165, 122, 125, 33]. Certification is a general and model-agnostic paradigm, which can be applied without additional retraining. Its advantage is to theoretically certify a level of robustness to attacks, with a correctness guarantee for the elected label in some radius around inputs sent to the classifier.

Randomized smoothing is without a doubt an important advance to approach the robustness of classifiers. Nevertheless, its application as a defense (and not only as a theoretical guarantee) comes with blind spots: *i*) The exact certified robustness is impossible to compute due to the dimensionality of the input space handled by current classifiers. Monte Carlo methods are used to estimate this certified robustness. There is a lack of understanding of the interplay between the theoretical certification for a radius that is fully spanned, and the practice where a limited amount of samples is key to tractability. In addition, since this defense is randomized in essence, the classic definition of an adversarial [221] is not applicable anymore. The defender lacks a definition of an adversary in the case of her randomized defense (since no attack trials is 100% adversarial). *ii*) The amount of samples required by this sampling approach is unclear and varying in the papers: between 100 [197, 124] and 100 000 [33, 105]. No results to date have shown the importance of this quantity on the effectiveness of attacks. *iii*) Finally, although this defense is in principle applicable without retraining, it is yet recommended [33] to mitigate



the accuracy drop involved. Indeed, the larger the noise radius certified, the more robust the classifier, at the price of an important accuracy drop that can be limited by retraining on noisy data. The relation of the radius with the final accuracy and the effectiveness of the attacks is also unclear.

This chapter tackles these three issues, in a dedicated attempt to consider randomized smoothing as a practical defense. We first confront theory and practice for certification and defense in the context of randomized smoothing. We then evaluate the practical robustness of this defense with regard to the impact of the Monte Carlo sample sizes and the noise variance parameter. This study highlights the effectiveness of randomized smoothing in defeating state-of-the-art black-box attacks with much smaller parameters that are suggested in the papers limited to considering it as a mere theoretical certification only.

*All the notations unique to this chapter are listed in Table 7.12 at the conclusion of this manuscript. Additionally, for general notation, please refer to Table 7.9.*

## 6.2 Related Work - Randomized smoothing (RS)

Introduced by Lecuyer *et al.* [122], RS is a model-agnostic method to obtain a certified local robustness of a model. It guarantees a correct prediction within a certain radius around a given input. In other words, it certifies that no adversarial example lies at a distance smaller than this radius. The beauty of this literature is that evasion attacks are no longer considered since the robustness is formally guaranteed. In practice, RS is merely a Monte Carlo simulation requiring a large number of calls to the model. It produces a lower bound of robustness and spoils accuracy. Papers [125, 33, 79] found stronger bounds while [122, 197] improved the noise tolerance of classifiers. None of them considers attacking RS.

## 6.3 Randomized smoothing: from theory to practice

This section summarizes RS certification for  $\ell_2$  norm robustness, focusing on the differences between theory and practice.

### 6.3.1 A primer on random smoothing

For the sake of simplicity, consider a trained binary classifier  $f : \mathbb{R}^D \rightarrow \{0, 1\}$ . RS defines a new classifier  $g_\sigma$  as follows:

$$g_\sigma(\mathbf{x}) = \arg \max_{\mathbf{y} \in \{0,1\}} \mathbb{P}[f(\mathbf{x} + \sigma \mathbf{N}) = \mathbf{y}], \quad \mathbf{N} \sim \mathcal{N}(0, I). \quad (6.1)$$

The main advantage of  $g_\sigma$  is that its robustness is certified. Assume that a genie reveals the value of the two probabilities  $\pi_0(\mathbf{x}) := \mathbb{P}[f(\mathbf{x} + \sigma \mathbf{N}) = 0]$  and  $\pi_1(\mathbf{x}) := 1 - \pi_0(\mathbf{x})$ , then  $\mathbf{x}$  is classified according to (6.1) with a certified robustness:

$$R(\mathbf{x}, \sigma) = \sigma \Phi^{-1}(\pi_{g_\sigma(\mathbf{x})}(\mathbf{x})). \quad (6.2)$$

where  $\Phi$  is cumulative distribution function of the standard normal distribution  $\mathcal{N}(0, I)$ . All points at a distance from  $\mathbf{x}$  lower than  $R(\mathbf{x}, \sigma)$  are classified in the same way. Note that despite the term ‘randomized smoothing’,  $g_\sigma$  is indeed a deterministic classifier. Its frontier  $\partial g_\sigma$  is the locus of the points s.t.  $\pi_0(\mathbf{x}) = 1/2$ . For instance, if the base classifier  $f$  is linear, then  $g_\sigma = f$ .

In practice, there is no genie and the defender uses a Monte Carlo simulation over  $n_{\text{mc}}$  random i.i.d. samples  $\mathbf{n}_i$  distributed as  $\mathbf{N}$  yielding  $n_{\text{mc}}$  decisions  $y_i$  with  $i$  in  $\llbracket 1, n_{\text{mc}} \rrbracket$ .

The final predicted class is an aggregation of these  $n_{\text{mc}}$  ‘micro’-decisions such as the majority vote. They also give a confidence interval  $\underline{\pi}_0(\mathbf{x}) < \pi_0(\mathbf{x})$  up to a given confidence level. This defines the classifier  $g_{\sigma, n_{\text{mc}}}$ , a practical implementation of the ideal  $g_\sigma$  function. The robustness is assessed up to the confidence level using (6.2) with  $\underline{\pi}_0$ , which yields  $\underline{R}(\mathbf{x}, \sigma) < R(\mathbf{x}, \sigma)$ . Maximizing the certified robustness around a given point  $\mathbf{x}$  with a high confidence level requires large  $n_{\text{mc}}$  and  $\sigma$  [122, 33, 197].

### 6.3.2 A critical point of view

The main argument of RS is the following: Leading evasion attacks to gauge the security of a classifier is no longer needed since its robustness is certified. This has to be clarified:  $\underline{R}(\mathbf{x}, \sigma)$  certifies the robustness of the theoretical classifier  $g_\sigma$  which does not exist in practice. The practical classifier  $g_{\sigma, n_{\text{mc}}}$  behaves as  $g_\sigma$  only when  $n_{\text{mc}} \rightarrow \infty$ .

More importantly,  $g_{\sigma, n_{\text{mc}}}$  is not a deterministic function. For  $\mathbf{x} \in \partial g_\sigma$ ,  $g_{\sigma, n_{\text{mc}}}(\mathbf{x})$  acts as a random variable from one call to another since  $\pi_0(\mathbf{x}) = \pi_1(\mathbf{x}) = 1/2$  even for large  $n_{\text{mc}}$ . This challenges the concept of frontiers hence the definition of adversarial examples.

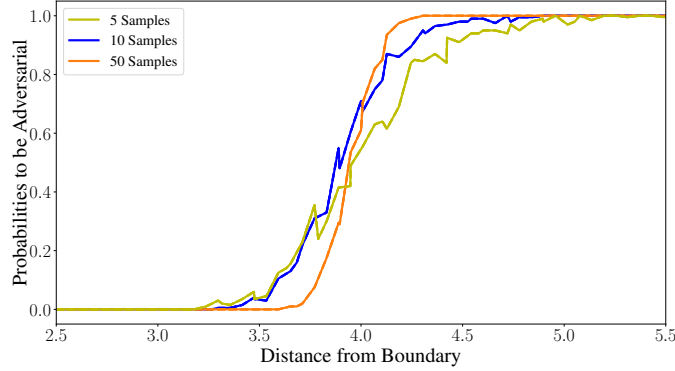


Figure 6.1 – Probability of being adversarial by following the direction  $\mathbf{x}_{\text{adv}} - \mathbf{x}_o$ , Image  $\mathbf{x}_o$  attacked with BP [280, 15] to get the best adversarial  $\mathbf{x}_{\text{adv}}$  on the boundary of ResNet 50.

### 6.3.3 Pushing the frontiers

Consider a point  $\mathbf{x}$  s.t.  $f(\mathbf{x}) = 1$  and at a distance  $\delta = \beta\sigma$  from the frontier  $\partial f$  of the base classifier. The so-called SORM in statistical reliability engineering approximates

$$\pi_0(\mathbf{x}) \approx \Phi(-\beta) \prod_{i=1}^{d-1} \frac{1}{\sqrt{1 + \beta\kappa_i}}, \quad (6.3)$$

where  $\{\kappa_i\}$  are the signed principal curvatures of the surface  $\partial f$ . If flat, all the curvatures equal 0, and  $\mathbf{x}$  lies on the boundary  $\partial g_\sigma$  of the ideal RS classifier if  $\pi_0(\mathbf{x}) = 1/2$  implying  $\delta = 0$ . If  $\partial f$  is convex toward  $\mathbf{x}$ , the curvatures are all negatives, the second term gets larger and compensates  $\Phi(-\beta)$  so that  $\pi_0(\mathbf{x}) = 1/2$  for some  $\beta > 0$ . This shows that the frontier  $\partial g_\sigma$  is closer than  $\partial f$  when lying in a convex region, and thus further away when sitting in a concave region. If the original images lie in concave regions, then RS pushes the frontier and thus increases the norm of the adversarial perturbation. In Figure 6.1, a white-box attack against model  $f$  first finds an adversarial  $\mathbf{x}_{\text{adv}} \in \partial f$ . We see that going along the direction  $\mathbf{x}_{\text{adv}} - \mathbf{x}_o$ , we cross the frontier  $\partial g_\sigma$  (i.e.  $\pi_0(\mathbf{x}) = 0.5$ ) after  $\mathbf{x}_{\text{adv}} \in \partial f$  for a  $\mathbf{x}$  s.t.  $\|\mathbf{x} - \mathbf{x}_{\text{adv}}\| \approx 4.0$ . This is also illustrated in Figure 6.2 on a 2D cut of  $\mathbb{R}^d$ .

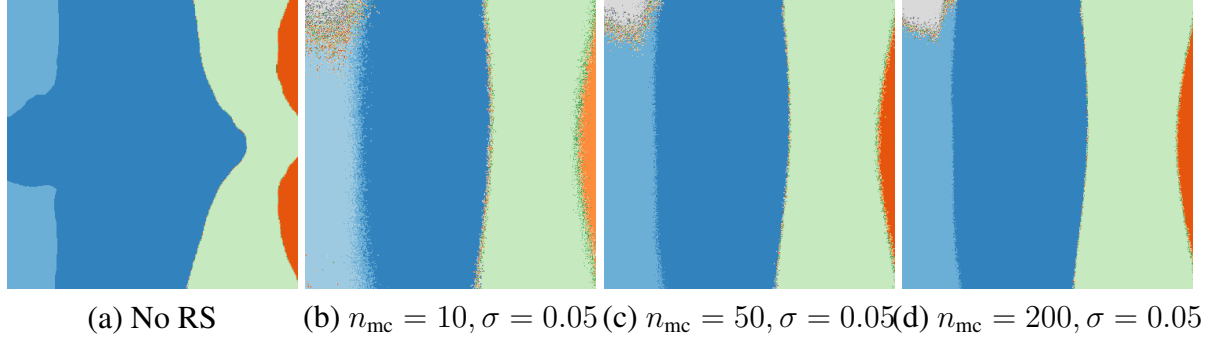


Figure 6.2 – 2D slice in the image space of ResNet50 with and without RS. Each point is an image, his color represents the elected label. Centered on a real image, 2 random directions are taken.

### 6.3.4 Adversarial example with confidence level

We propose a new definition for untargeted attack: an adversarial example of  $\mathbf{x}_o$  of level  $P_a \in [0, 1]$  is a point  $\mathbf{x}_{\text{adv}}$  s.t.

$$\mathbb{P}[g_{\sigma, n_{\text{mc}}}(\mathbf{x}_{\text{adv}}) \neq g_{\sigma}(\mathbf{x}_o)] \geq P_a. \quad (6.4)$$

If the attacker is satisfied with a level  $P_a = 1/2$ , then the closest adversarial example lies on the frontier of  $g_{\sigma}$  at a distance bigger than  $R(\mathbf{x}, \sigma) > \underline{R}(\mathbf{x}, \sigma)$ . We believe that attackers are requiring stronger guarantee  $P_a > 1/2$ , hence the closest adversarial example is at an even bigger distance. Eq. (6.4) requires that  $\sum y_i \sim \mathcal{B}(n_{\text{mc}}, 1 - \pi_0(\mathbf{x}_{\text{adv}}))$  takes a value greater than  $n_{\text{mc}}/2$  (due to the majority vote) with a probability larger than  $P_a$ . This holds for:

$$\pi_0(\mathbf{x}_{\text{adv}}) < 1 - I_{P_a}^{-1}(\tilde{n}, \tilde{n}) < 1/2, \quad (6.5)$$

with  $\tilde{n} = 1 + \lfloor n_{\text{mc}}/2 \rfloor$  and  $I_p^{-1}(a, b)$  is the inverse incomplete beta function. Applying (6.2) onto  $\mathbf{x}_o$  and  $\mathbf{x}_{\text{adv}}$ , it comes that

$$\begin{aligned} \|\mathbf{x}_o - \mathbf{x}_{\text{adv}}\| &= \|\mathbf{x}_o - \mathbf{x}_b\| + \|\mathbf{x}_b - \mathbf{x}_{\text{adv}}\| \\ &\geq R(\mathbf{x}_o, \sigma) + \sigma \Phi^{-1}(I_{P_a}^{-1}(\tilde{n}, \tilde{n})), \end{aligned} \quad (6.6)$$

where  $\mathbf{x}_b \in [\mathbf{x}_o, \mathbf{x}_{\text{adv}}] \cap \partial g_{\sigma}$ . To conclude, the robustness  $\underline{R}(\mathbf{x}_o, \sigma)$  certified by the practical implementation of RS is even less tight in practice.

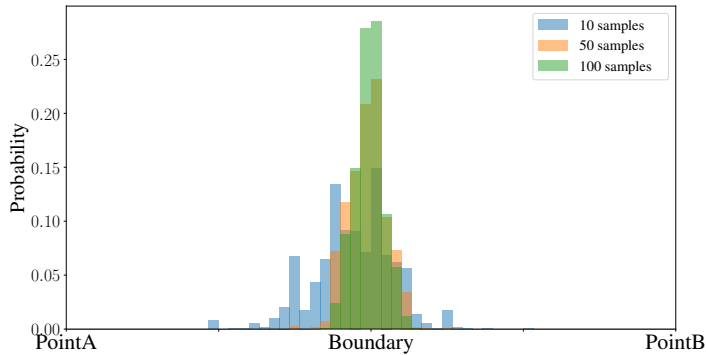


Figure 6.3 – Distribution of the output of a binary search with RS.

### 6.3.5 Jeopardizing black box attacks

Black box attacks usually make two assumptions. First, from a point outside the class region, a binary search can find a point  $\mathbf{x}_b$  right on the boundary within a controlled accuracy. However, RS classifier  $g_{\sigma, n_{mc}}$  is random in practice especially when  $n$  is small and this jeopardizes the binary search. Figure 6.3 shows the distribution of the result of the binary search. It concentrates around  $\partial g_{\sigma}$  only when  $n$  is large.

Second, the boundary is smooth so that it is possible to estimate the normal vector of the tangent hyperplane locally around  $\mathbf{x}_b$  on the boundary. This is usually done by bombarding the classifier with noisy versions of  $\mathbf{x}_b$  and observing its outputs. Yet, RS randomizes the immediate neighborhood of boundaries, as seen in Figure 6.2. This indeed does not spoil the estimation. We notice that normal vector estimations for  $\mathbf{x}_b \in \partial g_{\sigma}$  with and without RS correlate very well, provided that the noise variance used for the estimate is larger than the variance  $\sigma^2$  of RS. A large  $\sigma^2$  may spoil the estimation but it is detrimental to the natural accuracy of  $g_{\sigma, n_{mc}}$ . Yet, the estimation is true of poor quality due to the violation of the first assumption: the binary search may yield a point  $\mathbf{x}_b$  not exactly on the boundary and this biases the estimate. For instance, we notice that HSJA [23] may crash because all the noisy versions of  $\mathbf{x}_b$  give the same output.

## 6.4 Black Box Attacks vs. RS

### 6.4.1 Experimental Setup

We attack the classifier models with 200 random images from the ILSVRC2012’s validation set with size  $D = 3 \times 224 \times 224$ .

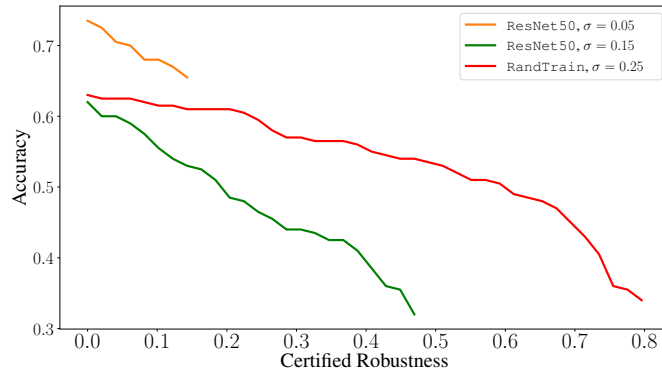


Figure 6.4 – Certification for ResNet50 and RandTrain.

**Classifiers** The base classifier is ResNet50 [80]. RS is performed with two noise standard deviations:  $\sigma = 0.05$  gives an acceptable drop of the accuracy of 3%, whereas  $\sigma = 0.15$  yields larger certified robustness value but with a loss of 12% of accuracy (see Figure 6.4).

Paper [33] proposes to re-train the model with noisy data to use a bigger  $\sigma$  without sacrificing too much accuracy. This new model is called RandTrain. With  $\sigma = 0.25$ , the accuracy loss is also around 12% but it delivers larger certified robustness (see Figure 6.4).

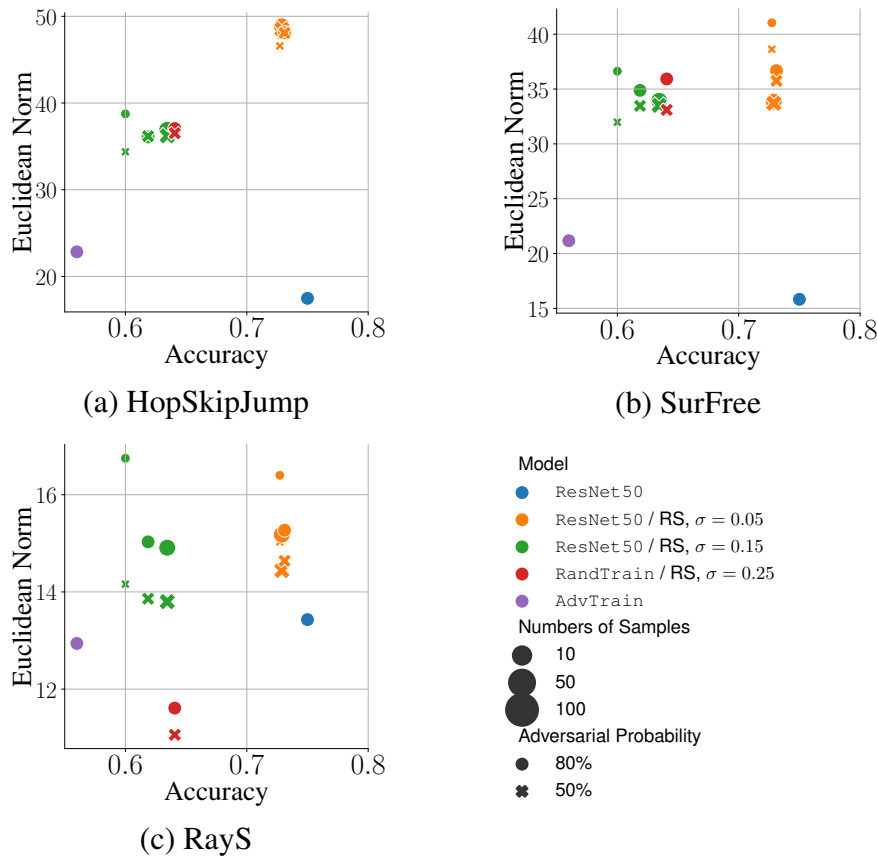
We compare RS to the adversarially trained ResNet50 from [142] that we denote AdvTrain.

**Black-box attacks** Three state-of-the-art attacks are considered. RayS [24], SurFree [146], and HSJA [23] achieve good results within 1,000 calls to the classifier, but we use up to 2,000 queries to be sure they reach their full potential.

**Protocol** The distortion is measured as the Euclidean norm of the adversarial perturbation in the domain  $[0, 1]^D$ . To assess that a point  $\mathbf{x}_{\text{adv}}$  complies with (6.4), the attacker needs to query  $\ell = O(1/P_a)$  times the classifier  $g_{\sigma, n_{\text{mc}}}$ . We speed up the simulation by considering that (6.4) holds if  $\lceil n_{\text{mc}} P_a \rceil$  micro-decisions are not correct.

## 6.4.2 Evaluation Results

**Certified robustness vs. practice** The gap between theory and practice is salient when considering Figure 6.4 and Figure 6.5: Figure 6.5 reports distortions at least 30 times larger than the certified robustness in Figure 6.4.


 Figure 6.5 – Average adversarial  $\ell_2$  distortion vs. Accuracy

**New definition of adversarial needed** Attacks are not disturbed by level  $P_a$  (6.4). Being 80% adversarial forces to move away a little (Figure 6.1) especially for small  $n_{mc}$ . The robustness is only slightly better.

**A small amount of noise is enough** Regardless of the attack, a large  $\sigma$  does not robustify the network whereas it spoils its accuracy. This is true even if the network has learned to handle noise: RandTrain has the same robustness and the same accuracy as the vanilla ResNet50 with RS  $\sigma = 0.15$ . The situation is even worse against RayS [24]: noticeably RandTrain is less robust than ResNet50 without RS.

**A small number of samples is enough** A big number of samples is key to getting ‘large’ certified robustness. Figure 6.5 shows another reality. The robustness against all 3 attacks is better with fewer samples. This confirms explanations in Section 6.3.5. Fewer samples make the prediction at the boundary more random which jeopardizes more black box attacks.

Binary search is the only tool common to the 3 attacks. A point exactly on the boundary is crucial for HSJA [23] since it estimates the gradient. SurFree [146] does not do that but relies on the smoothness of the boundary. As for RayS [24], the binary search improves the distortion but it is not crucial for the convergence. It explains why RayS [24] is not as impacted as SurFree [146] and HSJA [23].

## 6.5 Conclusion

Certification with randomized smoothing is an important advance to apprehend the robustness of classifiers. Yet it was not considered as a practical defense; this chapter chose this angle to reveal its real robustness facing state-of-the-art black-box attacks. We *i*) illustrated formally the gap between a theoretical certification and a practical defense, and redefined what is an adversarial that faces a randomized defense. We found that the recommendations made in order to have larger certified bounds are often antagonistic with the concrete actions for obtaining a robust and accurate classifier in practice: *ii*) a low amount of samples is enough to fuzzy the frontiers; this is key to bother black box attacks, and *iii*) a high noise variance does not robustify the classifier much, while it make accuracy drop; a small variance is enough.





PART III

# **Contribution on Model Confidentiality**

---



# FINGERPRINTING CLASSIFIERS WITH BENIGN INPUTS

---

## 7.1 Introduction

Fingerprinting classifiers aims at deriving a signature uniquely identifying a machine learning model, like the human fingerprint's minutiae in biometry. This is essentially a black-box problem: the classifier to be identified is in a black-box in the sense that one can just make some queries and observe the resulting model outputs. For instance, this is the case when the model is embedded in a chip, or accessible via an API.

The main application that related works [245, 19, 178, 172, 284] target is the proof of ownership. An accurate deep neural network is a valuable industrial asset due to the know-how for training it, the difficulty of gathering a well-annotated training dataset, and the required computational resources to learn its parameters. In this context, the entity identifying a black-box wants to *detect* whether it is not a stolen model of her.

Another at least critical application is information gain. For instance, an attacker willing to delude the classifier first gains some knowledge about the remote model, or a company wants to determine which model is in use in a competitor's production system. This aspect has been left aside as of today, and we tackle it under the notion of the fingerprinting *identification* task.

For clarity, we name Alice the entity willing to identify the model that Bob has embedded in the black-box.

### 7.1.1 Challenges

We hereafter name a *model* a reference architecture, together with its set of hyperparameters tuned by its designers. When any of these components are modified, we coin the resulting model a *variant*. The biggest difficulty is that there exist plenty of ways to modify a model while maintaining its intrinsic good accuracy. These procedures simplify a network (quantization of

the weights and/or activations, pruning, see *e.g.* [78]), or make it more robust (preprocessing of the input, adversarial re-training [66]). These mechanisms were not a priori designed to make fingerprinting harder but they leave room for Bob to tamper with the fingerprint of a model. We assume that Alice also knows some of these procedures. Yet, they are often defined by many parameters and among them scalars so that there is virtually an infinity of variants. Like in biometry, the fingerprint should be discriminative enough to be unique per model but also sufficiently robust to identify a variant.

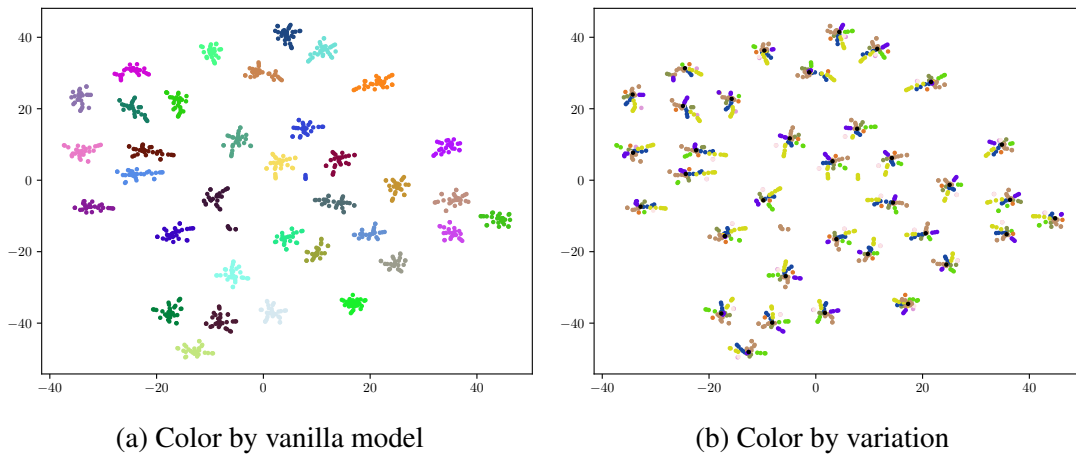


Figure 7.1 – A t-SNE representation of the pairwise distances of 1081 different models: 10 types of variation applied on 35 off-the-shelves vanilla models for ImageNet with different parameters (listed in App. B.1).

The approaches in the literature have two common pillars. They use the boundaries in the input space drawn by a classifier as the fingerprint, *i.e.* the unique signature identifying the model [19, 172, 284]. Two neural networks sharing the same architecture, the same training set and procedure are different because the training is stochastic (like the Stochastic Gradient Descent). This causes their boundaries in the input space not to overlap fully. Most of the papers in the literature are looking for discriminative deviations of these boundaries. Second, the key task is detection: Alice makes a guess about the model in the black-box and then she sends specific queries to test whether her hypothesis holds [19, 85, 172, 284].

## 7.1.2 Our Rationale and Contributions

Our work differs from related works on two key aspects: *i)* we do not forge any specific input but use regular benign inputs, and *ii)* we directly identify models using their intrinsic classification behavior.

We thoroughly investigate the use of benign inputs for fingerprinting models contrary to the previous works crafting specific inputs. We thus do not need to probe the input space to discover the decision boundaries. Benign inputs constitute a certain advantage, as it removes the need of often complex crafting procedures. It is less prone to defenses being implemented on Bob’s side (*e.g.* rejection based on the distance to the decision frontier [152]).

The second salient observation is the restriction of previous works on the detection task. The more general possibility to identify a model or a family of models inside the black-box remains unstudied.

In a nutshell, when Bob has picked a model among a set of networks known by Alice, then our solution is essentially deterministic: Alice has to find a sequence of inputs of minimum length to identify the black-box. We apply a greedy algorithm that carefully selects the input to iteratively narrow down the set of suspects, *i.e.* candidate models, until it becomes a singleton. Approximation theory tells that this is suboptimal but we report that in practice many networks are indeed identified within less than three input queries.

When Bob has made a variant of a model, its output may not match the output of any known model by Alice. We then use C.E. Shannon’s information theory to measure the statistical similarity between the outputs of two models. This approach is common in the field of Information Forensics and Security, especially in biometry [270, 42], PUF [12], content identification [273, 240], or traitor tracing [65, 95].

As an appetizer, Figure 7.1 depicts the t-SNE representation from the pairwise distances within a set of 35 vanilla models and their variants. The model families are well clustered in the sense that variants are closer to their original network than any other model. Alice may not identify precisely the variant of the model but at least she can accurately identify its family, *i.e.* infer which was the original vanilla model and even which kind of variation Bob applied.

Our contribution is fourfold.

1. We demonstrate that the mere use of benign images is enough to accomplish high success rates for fingerprinting modern classification models. This is to be opposed to the computationally demanding task of crafting inputs for that same goal.
2. The fingerprinting detection task, introduced by state-of-the-art works, is complemented with the introduction of the identification task. We frame the latter as an information theoretical problem.
3. We present a distance based on the empirical Mutual Information, gauging how close two models are. This distance permits generalizing the notion of modifications (also coined as *attacks*) on models through the concept of model families and variants.

4. We perform extensive experimentation by considering more than 1,000 classification models on ImageNet. A head-to-head comparison with the two related works reports significant improvements w.r.t. accuracy in the detection task.

Section 7.2 is a threat analysis listing all the working assumptions in our work. The next two sections deal both with detection (Alice verifies her hypothesis about the black-box) and identification (Alice discovers which model is in the black-box) but under two scenarios: Section 7.3 builds on the fact that Bob has picked a model among the set known by Alice, whereas Section 7.4 assumes that the black-box may be an unknown model. Both sections contain experimental results.

*All the notations unique to this chapter are listed in Table 7.14 at the conclusion of this manuscript. Additionally, for general notation, please refer to Table 7.9.*

## 7.2 Threat Model

This section details the goals of Alice and Bob.

### 7.2.1 Bob: Keeping his Model Anonymous

#### Goals

Bob is playing first by secretly selecting a model and putting it in the black-box under scrutiny. This model can be a vanilla model or a variant of a known model. A variant is created by applying on a given vanilla model  $m$  the procedure  $V$  parametrized by  $\theta \in \Theta$  which describes the type of modification and the associated parameters. This can be thought of as an attack by Bob on the vanilla model to harden identification. We denote such a variant by  $v = V(m, \theta)$ .

The goal of Bob is to offer an accurate black-box classifier while maintaining the ‘anonymity’ of the model in use. The first requirement is that a small loss in the model performance is tolerated by Bob. If a variant does not comply with this criterion then Bob cannot consider it as an option. In classification, the performance of a model  $m$  is often gauged by the top-1 accuracy, denoted  $\eta(m)$ . We formalize this requirement as

$$\frac{\eta(m) - \eta(V(m, \theta))}{\eta(m)} < \eta, \quad (7.1)$$

where  $\eta > 0$  is the tolerance (15% in our experimental work).

We also assume that the black-box performs the same classification task. As far as we know, fingerprinting is not possible between two networks performing different tasks if only top-k output is available. Transfer learning is therefore not considered as in previous works [178, 19, 246].

## Resources

The second requirement is more subtle. We first need to limit the power of Bob. If Bob creates an accurate model *ex nihilo*, then Alice can pursue neither detection nor identification. We assume that Bob cannot train such a model from scratch because he lacks good training data, expertise in machine learning, or computing resources. This also means that Bob can retrain a model only up to a limited extent (typically using a small amount of new data). In other words, the complexity of the procedure creating  $v = V(m, \theta)$  ought to be much smaller than the effort spent at training the original model  $m$ . Our experimental work considers two kinds of procedures.

**Modification of the Input**  $v(x) = m(T(x, \theta))$ . Classifiers are robust to light input modifications. For images, the transformation  $T$  can be JPEG compression, posterizing, blurring, *etc.*. In the same spirit, *randomized smoothing* [33] consists in adding noise to the input and aggregating the predicted classes into one single output.

**Modification of the Model**  $v(x) = T(m, \theta)(x)$ . The transform  $T$  slightly changes the model weights by for instance quantization, pruning, adversarial retraining or finetuning. Some of these procedures require small retraining with few resources so as not to lose too much accuracy.

In the sequel, the model in the black-box is denoted by  $b$  and  $\mathcal{B}$  is the set of all possibilities, defined as:

$$\mathcal{B} := \{v = V(m, \theta) : m \in \mathcal{P}, \theta \in \Theta, \eta(v) > (1 - \eta)\eta(m)\}, \quad (7.2)$$

where  $\mathcal{P}$  is a set of vanilla models and  $\Theta$  a set of transformations (encompassing the identity  $v = m$ ).



## 7.2.2 Alice: Disclosing the Remote Model

### Goals

The task of Alice is to disclose which model is in the black-box. This has two flavours: detection or identification.

*Detection* (denoted by  $\text{det}$ ) means that Alice performs a hypothesis test. She first makes a hypothesis about the black-box, then makes some queries, and finally decides whether the hypothesis holds based on the outputs of the black-box. The outcome of the detection is thus binary: Alice’s hypothesis is deemed correct or not. This is the nominal use case in the related works [245, 19, 178, 172, 284].

*Identification* (denoted by  $\text{I}$ ) means that Alice has no prior about the model in the black-box. She makes queries and processes the outputs to finally make a guess. The outcome is either the name of a model she knows, or the absence of a decision if she has not enough evidence.

### Knowledge about the Black-Box

The second crucial point is her knowledge about the black-box. Alice can only detect or identify a relation to a model she knows: it means she has an implementation of this model, which she can freely test. We denote the set of models known by Alice by  $\mathcal{A}$ .

As by the very definition of a *variant*, Alice may know some of them but not all of them. For instance, some procedures  $V$  admit a real number as a parameter. Therefore, there is virtually an infinite number of variants. This leads to the convenient notion of a model *family*, we now introduce under three flavours:

- $\mathcal{F}(m)$ : This family is the set of all variants made from the original vanilla model  $m$ :

$$\mathcal{F}(m) := \{v = V(m, \theta) : \theta \in \Theta\}. \quad (7.3)$$

- $\mathcal{F}(m, \Psi)$ : This family is the set of all variants made from the original vanilla model  $m$  by a specific procedure:

$$\mathcal{F}(m, \Psi) := \{v = V(m, \theta) : \theta \in \Psi \subset \Theta\}, \quad (7.4)$$

where  $\Psi$  denotes the subset of parameters related to this specific procedure.

- $\mathcal{F}(m, \{\theta\})$ : This family is a singleton composed of a particular variant:

$$\mathcal{F}(m, \{\theta\}) := \{v = V(m, \theta)\}. \quad (7.5)$$

With these definitions in mind, detection is based on the hypothesis that the black-box belongs to a given family, while Identification looks for the family the black-box belongs to.

### Resources

A third element is the resources of Alice. We already mention the set  $\mathcal{A}$  containing some vanilla models and few variants of theirs. She also has a collection of typical inputs, *i.e.* a testing dataset. We suppose that these inputs are statistically independent and distributed as the data in the training set of the models. In the sequel, the collection of inputs is denoted  $\mathcal{X} = \{x_1, \dots, x_N\}$ .

In the end, be it for detection or identification, Alice selects some elements of  $\mathcal{X}$  for querying the black-box. We denote this by an ordered list of indices:  $q_{1:\ell} = (q_1, \dots, q_\ell) \in \llbracket N \rrbracket^\ell$ , where  $\llbracket N \rrbracket := \{1, \dots, N\}$ . This means that Alice first queries  $x_{q_1}$ , and then  $x_{q_2}$  and so forth. The outputs of the black-box are denoted as  $z_{1:\ell} = (z_{q_1}, \dots, z_{q_\ell})$ , with  $z_{q_i} = \mathbf{b}(x_{q_i})$ .

### 7.2.3 The Classifier in the Black-Box

The black-box works as any classifier. We denote the set of possible classes  $\mathcal{C}$ . The output  $z = \mathbf{b}(x)$  for input  $x$  is the first  $k$  classes ordered by their predicted probabilities (*i.e.* the top- $k$ ). It means that  $z$  is an ordered list in  $\mathcal{C}^k$ :  $z = (c_1, \dots, c_k)$ . The set  $\mathcal{Z}_k$  of possible outcomes has a size as big as  $(\mathcal{C})_k := C(C-1) \dots (C-k+1)$ . The black-box only discloses the top- $k$  classes (*i.e.* this work does not build on the associated predicted probabilities). In the experimental work, the size of  $\mathcal{C}$  is  $C = 1,000$  (ImageNet) and  $k \in \{1, 3, 5\}$  which is usual in several image classification APIs. We assume that the considered models and variants have an accuracy which is not perfect; the typical accuracy of ImageNet classic models ranges from 70% to 85%.

### 7.2.4 Summary

This chapter considers scenarios which are labeled as  $(\text{Task}, \mathcal{F}, \mathcal{A}, k)$  where  $\text{Task} \in \{\text{det}, \text{I}\}$  (Detection or Identification),  $\mathcal{F}$  is the kind of family that will be inferred by Alice,  $\mathcal{A}$  is the set of models known by Alice, and  $k$  indicates that the output of the black-box is the top- $k$  classes. There is a clear cut between the following two cases:

- Walled garden:  $\mathcal{A} = \mathcal{B}$ . We impose that the black-box is one of the networks known by Alice.
- Open world:  $\mathcal{A} \subsetneq \mathcal{B}$ . The black-box may not be a model known by Alice. This is the case when Bob uses an unknown variant, for instance.

This distinction drives the structure of the next sections because our solutions are of different nature.

## 7.3 Walled Garden

### The Black-Box is a Known Model

Under the assumption that  $\mathcal{A} = \mathcal{B}$ , Alice achieves her goal when she correctly guesses which family the black-box belongs to. The alternative is to fail to gather enough evidence to make a decision. For the sake of clarity, we explain our procedure for a given family  $\mathcal{F} \subset \mathcal{A}$ , which can be one of the three types of families presented in Section 7.2.2.

Alice has a set of models composed of some vanilla models  $\mathcal{P} = \{m_1, \dots, m_M\}$  and some variants of theirs. Alice also has the collection of benign inputs  $\mathcal{X} = (x_1, \dots, x_N)$ . Offline, she creates a database of  $|\mathcal{A}|N$  outputs  $(m(x_j))_{m \in \mathcal{A}, j \in [N]}$ .

Let  $\mathcal{D}$  be a subset of  $\mathcal{A}$ . We define by  $\mathcal{D}(x) := \{m(x) : m \in \mathcal{D}\}$  the set of labels predicted by the models in  $\mathcal{D}$  for input  $x$ . With abuse of notations,  $\mathcal{D}(x_{q_{1:\ell}})$  is the set of the concatenation of labels predicted by the models in  $\mathcal{D}$  for the entries  $(x_{q_1}, \dots, x_{q_\ell})$ . Conversely,  $\mathcal{M}(x, y, \mathcal{D}) := \{m : m \in \mathcal{D}, m(x) = y\}$  lists the models in  $\mathcal{D}$  predicting  $y$  for input  $x$ .

#### 7.3.1 Detection ( $\text{det}, \mathcal{F}, \mathcal{A} = \mathcal{B}, k$ )

Alice first makes a hypothesis about a family  $\mathcal{F}$ , and her goal is to discover whether the outcome is *positive* ( $b \in \mathcal{F}$ ) or *negative* ( $b \in \mathcal{A} \setminus \mathcal{F}$ ). We assume that Alice is convinced about her hypothesis and that she hopes for a *positive*. Our procedure thus focuses on reducing the number of models in  $\mathcal{A} \setminus \mathcal{F}$  likely to be the black-box.

Alice uses a greedy algorithm which leverages the information about the black-box retrieved from the previous queries. According to the outputs of the black-box, several models can be discarded. At step  $\ell$ ,  $(\mathcal{A} \setminus \mathcal{F})^{(\ell)}$  (resp.  $\mathcal{F}^{(\ell)}$ ) denote the subset of models in  $\mathcal{A} \setminus \mathcal{F}$  (resp.  $\mathcal{F}^{(\ell)}$ ) which agree with the previous outputs. These are candidates in the sense that they could be the black-box model.

$$(\mathcal{A} \setminus \mathcal{F})^{(\ell)} := \bigcap_{i=1}^{\ell} \mathcal{M}(x_{q_i}, b(x_{q_i}), \mathcal{A} \setminus \mathcal{F}). \quad (7.6)$$

Initially, all the models are candidates:  $(\mathcal{A} \setminus \mathcal{F})^{(0)} = \mathcal{A} \setminus \mathcal{F}$  and  $\mathcal{F}^{(0)} = \mathcal{F}$ . At step  $\ell + 1$ , the greedy algorithm sorts the inputs that have not yet been queried according to a score. This score  $s^{(\ell+1)}(x)$  reflects how much the set of candidates  $(\mathcal{A} \setminus \mathcal{F})^{(\ell)}$  reduces if input  $x$  is submitted next.

We propose the expectation of the number of candidate models outside the family after querying input  $x$  assuming that the black-box is randomly picked in  $\mathcal{F}^{(\ell)}$ . This average is weighted by the number of models in  $\mathcal{F}^{(\ell)}$  predicting a particular label:

$$s^{(\ell+1)}(x) = \sum_{y \in \mathcal{F}^{(\ell)}(x)} \left| \mathcal{M}(x, y, (\mathcal{A} \setminus \mathcal{F})^{(\ell)}) \right| \frac{|\mathcal{M}(x, y, \mathcal{F}^{(\ell)})|}{|\mathcal{F}^{(\ell)}|} \quad (7.7)$$

Alice then submits one of the inputs with the lowest score:

$$q_{\ell+1} \in \arg \min_k s^{(\ell+1)}(x_k). \quad (7.8)$$

Our procedure stops after  $L$  iterations when meeting one of the three stopping criteria:

- $(\mathcal{A} \setminus \mathcal{F})^{(L)} = \emptyset$ : The detection result is *positive*. No model outside the family responds like the black-box. The black-box is in the family since we assume it belongs to  $\mathcal{A}$ .
- $\mathcal{F}^{(L)} = \emptyset$ : The detection result is *negative*. The responses of the black-box are different from the ones of the models in the family  $\mathcal{F}$ .
- $\min_k s^{(L+1)}(x_k) = |(\mathcal{A} \setminus \mathcal{F})^{(L)}|$ : The detection failed. All the remaining models in  $(\mathcal{A} \setminus \mathcal{F})^{(L)}$  and in  $\mathcal{F}^{(L)}$  produce the same prediction no matter which input is submitted. It is therefore impossible to discern them.

### 7.3.2 Identification ( $l, \mathcal{F}, \mathcal{A} = \mathcal{B}, k$ )

Identification means that Alice makes a partition of her set of models in disjoint families:  $\mathcal{A} = \cup_{i=1}^{n_{\mathcal{F}}} \mathcal{F}_i$ . Her goal is to identify which family the black-box belongs to.

It is easy to base a verification procedure onto a detection scheme assuming there is no failure. Alice arbitrarily orders the families, and sequentially tests the hypotheses until she finds a match. The expected number of queries is given by (see the proof in App. C):

$$\mathbb{E}(L) = \frac{1}{n_{\mathcal{F}}} \sum_{j=1}^{n_{\mathcal{F}}} \mathbb{E}(L_j^{\text{pos}}) + \frac{n_{\mathcal{F}} - 1}{2n_{\mathcal{F}}} \sum_{j=1}^{n_{\mathcal{F}}} \mathbb{E}(L_j^{\text{neg}}) \quad (7.9)$$

where  $(\mathbb{E}(L_j^{\text{pos}}), \mathbb{E}(L_j^{\text{neg}}))$  are the expected number of queries necessary for taking a positive or negative decision about the *detection* of the hypothesis  $\mathcal{F}_j$ .

We propose a better approach based on a greedy algorithm similar to the detection one. Suppose that Alice has already submitted  $\ell$  queries to the black-box. By comparing the outputs of the black-box and of the models she knows, she is able to distinguish models which are not in the black-box from models likely to be in the black-box. This list of remaining models is

denoted  $\mathcal{A}^{(\ell)}$ . The goal of Alice is to reduce the set of candidates to a single family  $\mathcal{F}_i$ , not knowing in advance the model Bob placed in the black-box:

$$\exists i, \mathcal{F}_i \subset \mathcal{A}^{(\ell)}. \quad (7.10)$$

In the beginning, all the models are possibly in the black-box, *i.e.*  $\mathcal{A}^{(0)} = \mathcal{A}$ . At step  $\ell + 1$ , the greedy algorithm chooses the best input to query next knowing  $\mathcal{A}^{(\ell)}$ .

Alice may resort to the following heuristics. She supposes that the black-box is randomly chosen uniformly in the set of remaining models  $\mathcal{A}^{(\ell)}$ . For any input  $x$  not queried yet, she computes the expectation of the number of remaining families if  $x$  were selected next, *i.e.*  $|\{\mathcal{F}_i : \mathcal{F}_i \cap \mathcal{A}^{(\ell+1)} \neq \emptyset\}|$ . She randomly chooses among the inputs minimizing this figure:

$$s^{(\ell+1)}(x) = \sum_{y \in \mathcal{A}^{(\ell)}(x)} \left( \sum_{i=1}^{n_{\mathcal{F}}} \delta_{[\mathcal{M}(x,y,\mathcal{F}_i^{(\ell)}) \neq \emptyset]} \right) \frac{|\mathcal{M}(x,y,\mathcal{A}^{(\ell)})|}{|\mathcal{A}^{(\ell)}|},$$

where  $\delta_{[\mathcal{E}]}$  is the indicator function of event  $\mathcal{E}$ . The input to be submitted is sampled among the ones with the lowest score:

$$q_{\ell+1} \in \arg \min_k s^{(\ell+1)}(x_k). \quad (7.11)$$

### 7.3.3 Experimental Work

#### Detection

A first experimental work measures the number of queries needed for detection with the three types of family defined in Eq. (7.3), (7.4) and (7.5). It considers two cases: Alice's hypothesis is correct (positive case) or incorrect (negative case). The combinations are not analyzed exhaustively. For example, in the negative case for a singleton family (*i.e.* Alice is wrong to suspect that the black-box is  $\mathcal{F}(m, \{\theta\})$ ), there are  $|\mathcal{A}|(|\mathcal{A}| - 1)$  possible combinations, *i.e.* more than a million. Instead, the experiment randomly picks 1,000 positive and 1,000 negative among all these cases. Figure 7.2 shows the results obtained. As a side-product, Table 7.1 gives the percentage of inputs in  $\mathcal{X}$  answering the detection problem under the best case, *i.e.* when one unique query is sufficient.

**Few Queries are Enough** When the detection succeeds (be it a positive or negative decision about the hypothesis), at most three queries are needed, and in most cases, only one is sufficient. This holds although the greedy algorithm is known to be suboptimal. When the greedy algo-

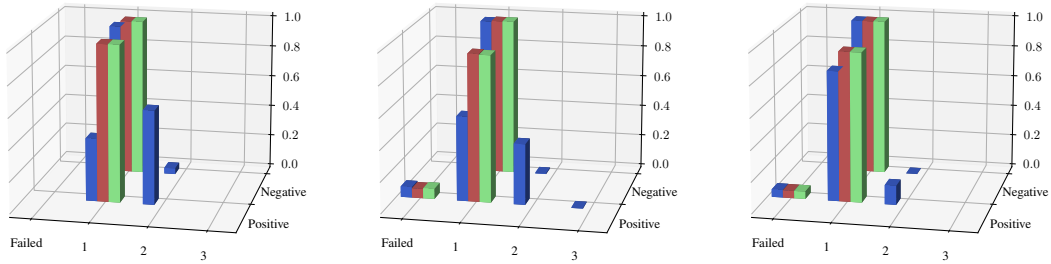


Figure 7.2 – Probability distribution of the number of queries for  $(\text{det}, \mathcal{F}, \mathcal{A} = \mathcal{B}, k)$  when the black box returns top- $k$  classes with  $k = 1$  (blue),  $k = 3$  (red) or  $k = 5$  (green). Family considered from left to right:  $\mathcal{F}(m)$  (7.3),  $\mathcal{F}(m, \Psi)$  (7.4), and  $\mathcal{F}(m, \{\theta\})$  (7.5).

rithm needs three inputs, another algorithm may only need two queries. Yet, when the greedy algorithm needs two, no algorithm can do better because the greedy would have found an unique input if existing. Positive and negative conclusions are roughly drawn within the same number of queries, although our algorithm is designed to quickly prove positive detections.

**Few Failures** The inability to detect the model in the black-box as part of the family  $\mathcal{F}$  happens when:

$$\exists m \in \mathcal{F}, \exists m' \in \mathcal{A} \setminus \mathcal{F}, \forall x \in \mathcal{X}, m(x) = m'(x). \quad (7.12)$$

The failures occur when the family corresponds to a set of variations (7.4) or an exact model (7.5). It happens that the algorithm cannot distinguish a few pairs of different variations issued from the same vanilla model. This is the only possible explanation: Otherwise, *i.e.* the indistinguishable models come from two different vanilla networks, a failure would also occur when detecting families spanned from a vanilla model (7.5), but this is not reported in Figure 7.2. In other words, Alice can always guess that the black-box is a variation of a given vanilla model, and rarely she cannot guess which variation it is exactly.

On the other hand, failures should also happen in the negative case. None is reported in Figure 7.2 because they are statistically rare. For a given family  $\mathcal{F}$ , suppose that the models  $m$  and

Table 7.1 – Percentage of inputs in  $\mathcal{X}$  concluding detection  $(\text{det}, \mathcal{F}, \mathcal{A} = \mathcal{B}, k)$  within a single query.

| Family                                       | top-1 | top-3 | top-5        |
|--|-------|-------|--------------|
| Vanilla $\mathcal{F}(m)$ (7.3)               | 0.28% | 1.2%  | <b>12.5%</b> |
| Variation $\mathcal{F}(m, \Psi)$ (7.4)       | 0.31% | 4.8%  | <b>21.0%</b> |
| Singleton $\mathcal{F}(m, \{\theta\})$ (7.5) | 0.37% | 8.3%  | <b>31.4%</b> |

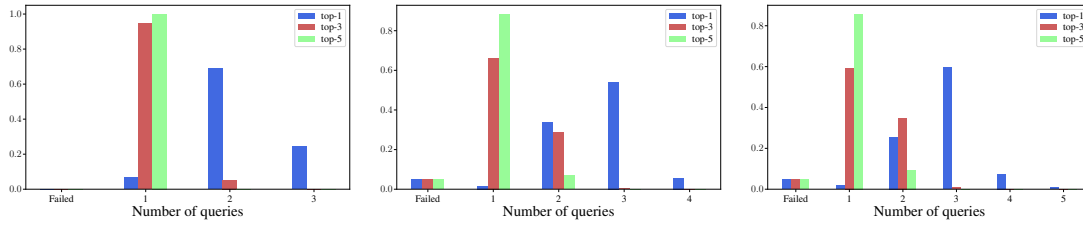


Figure 7.3 – Probability distribution of the number of queries for  $(l, \mathcal{F}, \mathcal{A} = \mathcal{B}, k)$  when the black box returns top- $k$  classes with  $k = 1$  (blue),  $k = 3$  (red) or  $k = 5$  (green). Family considered from left to right:  $\mathcal{F}(m)$  (7.3),  $\mathcal{F}(m, \Psi)$  (7.4), and  $\mathcal{F}(m, \theta)$  (7.5).

$m'$  in (7.12) are both unique. A failure happens in the positive case if Bob puts model  $m$  in the black-box. This happens with probability  $1/|\mathcal{F}|$ . A failure happens in the negative case if Bob puts model  $m'$  in the black-box. This happens with probability  $1/|\mathcal{A} \setminus \mathcal{F}| < 1/|\mathcal{F}|$ .

Experimentally, the number of queries to end up in a failure is similar to the number of queries for getting a positive outcome.

**A Bigger Top- $k$  is Better** When the output of the black-box is rich, *i.e.* top- $k$  classes with  $k > 1$ , one unique input is sufficient. Moreover, Table 7.1 shows that there are more of these unique inputs in  $\mathcal{X}$ . In this case, Alice no longer needs a large collection of benign inputs.

**A Bigger Family is Harder to Detect** Families of type (7.3) are bigger than families (7.4) which are bigger than the singleton (7.5). Ignoring the failure case, Figure 7.2 and Table 7.1 show that it is harder to detect a large family. It is more frequent that some model members take different outputs in large families. On the contrary, we observe that the variants of the same model with the same variation but with different parameters often share the same output.

## Identification

The protocol is similar to the previous one for detection. Figure 7.3 shows the results.

**Identification vs. Detection** Comparing Figure 7.2 and 7.3, two times more queries are necessary for identifying a family rather than detecting it. It is possible to identify a model quickly with at most five benign queries which are a lot less than the sequential procedure (7.9). Identification is a harder task than detection to a small extent.

The biggest difference is under the top-1 scenario where a unique query is rarely sufficient. The 35 vanilla models considered here were trained on the same dataset. They have good accu-

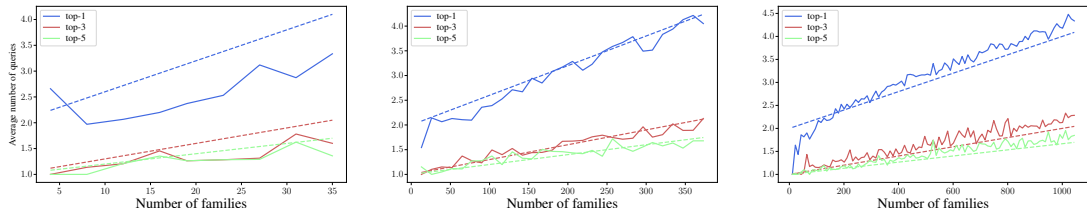


Figure 7.4 – Average number of queries as a function of the number of families  $n_{\mathcal{F}}$  for  $(1, \mathcal{F}, \mathcal{A} = \mathcal{B}, k)$  with the expectation score (7.7) and when the black-box returns top- $k$  classes with  $k = 1$  (blue),  $k = 3$  (red) or  $k = 5$  (green). The dotted lines represent the linear regressions (7.13). Family considered from left to right:  $\mathcal{F}(m)$  (7.3),  $\mathcal{F}(m, \Psi)$  (7.4), and  $\mathcal{F}(m, \{\theta\})$  (7.5).

racy ( $> 70\%$ ). If many unique inputs to identify existed, this would mean that for any of these inputs, the 35 models give 35 different top-1 predictions. Assuming that one of these models makes a correct classification, the other 34 models are wrong. If a lot of these inputs existed, this would imply models with low accuracy. In other words, these inputs are necessarily rare, or even non-existing.

**A Bigger Top- $k$  is Better** In contrast to detection, the gain of information provided by top-3 and top-5 is substantial. When the top-5 is returned, 90% of the families are identified within one query. The supervised training of the vanilla models only focuses on the top-1 s.t. it agrees with the ground truth class. For  $k > 1$ , the top- $k$  is almost specific of the model. This explains the big improvement from top-1 to top- $k$ .

**Number of Families** Figure 7.4 represents the evolution of the average number of queries to identify one out of  $n_{\mathcal{F}}$  families. The more families, the bigger the number of queries on average. But this number also depends on the size of the families and the top- $k$ . We observe that the increase is roughly linear (see dashed lines in Figure 7.4). As a rule of thumb, we observe that the expectation of the number of queries roughly follows the empirical law:

$$\mathbb{E}(L) \approx 0.002 \times \frac{\mathbb{E}(|\mathcal{F}|)n_{\mathcal{F}}}{k} + \beta(k), \quad (7.13)$$

where  $\mathbb{E}(|\mathcal{F}|)$  is the average number of elements in the family. This is a major improvement w.r.t. (7.9). For instance, for singleton family,  $\mathbb{E}(|\mathcal{F}|) = 1$  and the rate equals 0.002 under top-1, whereas the rate in (7.9) cannot be lower than 0.5 since we need at least one query to discard a hypothesis, i.e.  $\mathbb{E}(L_j^{\text{neg}}) \geq 1$ .



## 7.4 Open World: The Black-Box is an Unknown Model

This section assumes that  $\mathcal{A} \subsetneq \mathcal{B}$  because  $\mathcal{B}$  contains models or variants of models unknown by Alice.

### 7.4.1 Modeling

#### Assumptions

Our working assumptions are the following: When queried by random inputs, a variant  $V(m, \theta)$  produces outputs statistically

- independent from the outputs of a different model  $m'$ .
- dependent from the outputs of the original model  $m$ .

We consider a particular procedure for generating a variant as being like a transmission channel. The output  $Z$  of the variant  $V(m, \theta)$  is as if the output  $Y$  of the original model  $m$  were transmitted to Alice through a noisy communication channel parametrized by  $\theta$ . Like in C.E. Shannon's information theory of communication, we model this channel by the conditioned probabilities  $W_\theta(z, y) = \mathbb{P}(Z = z | Y = y), \forall (z, y) \in \mathcal{Z}_k$ .

#### Surjection

One difficulty of this context is the big size of the set  $\mathcal{Z}_k$  of outcomes under the top- $k$  assumption:  $|\mathcal{Z}_k| = (C)_k$ . It is then difficult to establish reliable statistics about the transition matrix  $W_\theta$  which is as large as  $(C)_k \times (C)_k$ .

When working with top- $k$  outputs, Alice resorts to a surjection  $S_k : \mathcal{Z}_k \mapsto \mathcal{S}_k$  with  $\mathcal{S}_k := \{0, 1, \dots, k\}$ . This greatly reduces the set of outcomes. We denote  $\tilde{z} = S_k(z)$  and  $\tilde{y} = S_k(y)$ . We choose a function  $S_k$  slightly more complex than suggested by this simple notation. Indeed, for any input  $x$ , we assume that Alice has a reference class  $c(x) \in \mathcal{C}$ . It is the ground truth class for annotated data. Otherwise, Alice computes the top-1 output of all the models she knows, and takes a majority vote to decide on  $c(x)$ . For this piece of data, a model gives  $m(x) = (c_1, \dots, c_k)$  and the surjection makes:

$$S_k(m(x)) = \begin{cases} j & \text{if } \exists j : c_j = c(x) \\ 0 & \text{otherwise.} \end{cases} \quad (7.14)$$

In words,  $S_k(m(x))$  is the rank of the reference class in the top- $k$  output or 0 if the reference class is not returned. In the end, Alice uses a transmission matrix  $(W_\theta(\tilde{z}, \tilde{y}))$  which is only  $(k+1) \times (k+1)$ .

### 7.4.2 Detection ( $\det, \mathcal{F}, \mathcal{A} \subsetneq \mathcal{B}, k$ )

For the detection task, Alice first makes the following hypothesis: The black-box is a variant of the vanilla model  $m \in \mathcal{A}$ . This variant may be the identity ( $b = m$ ), or a variant she knows, or a variant she does not know.

Contrary to the previous section, Alice randomly chooses  $L$  inputs  $(X_1, \dots, X_L) \subset \mathcal{X}$  to query the black-box and compares the observations  $(\tilde{Z}_1, \dots, \tilde{Z}_L)$  to the outputs she knows  $(\tilde{Y}_1, \dots, \tilde{Y}_L)$ , with  $\tilde{Z}_\ell := S_k(b(X_\ell))$ ,  $\tilde{Y}_\ell := S_k(m(X_\ell))$ ,  $\forall \ell \in \llbracket L \rrbracket$ . We use capital letters here to outline that these are random variables since Alice randomly chooses the inputs.

There are two difficulties: i) to gauge the distance between the outputs observed from the black-box and from model  $m$  (see Section 7.4.2) and ii) to randomly sample informative inputs from the set  $\mathcal{X}$  (see Section 7.4.2).

#### Discriminative Distance

Alice tests two hypothesis:

- $\mathcal{H}_1$ : The black-box is a variant of model  $m$ . There is a dependence between  $\tilde{Z}$  and  $\tilde{Y}$  which is captured by the statistical model of the variant:

$$\mathbb{P}_1(\tilde{Z} = \tilde{z}, \tilde{Y} = \tilde{y}) := W_\theta(\tilde{z}, \tilde{y})\mathbb{P}(\tilde{Y} = \tilde{y}).$$

- $\mathcal{H}_0$ : The black-box is not a variant of model  $m$ . There is no statistical dependence and

$$\mathbb{P}_0(\tilde{Z} = \tilde{z}, \tilde{Y} = \tilde{y}) := \mathbb{P}(\tilde{Z} = \tilde{z})\mathbb{P}(\tilde{Y} = \tilde{y}).$$

The well-celebrated Neyman-Pearson test is the optimal score for deciding which hypothesis holds. For  $L$  independent observations, it writes as

$$s = \sum_{j=1}^L \log \frac{\mathbb{P}_1(\tilde{Z} = \tilde{z}_j, \tilde{Y} = \tilde{y}_j)}{\mathbb{P}_0(\tilde{Z} = \tilde{z}_j, \tilde{Y} = \tilde{y}_j)} = \sum_{j=1}^L \log \frac{W_\theta(\tilde{z}_j, \tilde{y}_j)}{\mathbb{P}(\tilde{Z} = \tilde{z}_j)}. \quad (7.15)$$

We introduce the empirical joint probability distribution defined by

$$\hat{P}_{\tilde{Z}, \tilde{Y}}(\tilde{z}, \tilde{y}) := L^{-1} |\{j \in \llbracket L \rrbracket : \tilde{z}_j = \tilde{z} \text{ and } \tilde{y}_j = \tilde{y}\}| \quad (7.16)$$

in order to rewrite (7.15) as

$$s = L \sum_{(\tilde{z}, \tilde{y}) \in \mathcal{S}_k^2} \hat{P}_{\tilde{Z}, \tilde{Y}}(\tilde{z}, \tilde{y}) \log \frac{W_\theta(\tilde{z}, \tilde{y})}{\mathbb{P}(\tilde{Z} = \tilde{z})}. \quad (7.17)$$

This formalization is not tractable because  $W_\theta$  is not known: Alice does not know which variant  $\theta$  is in the black-box, and indeed it might be an unknown variant. Yet, (7.17) guides us to a more practical score function, the empirical mutual information:

$$\hat{I}(\tilde{Z}, \tilde{Y}) := \sum_{(\tilde{z}, \tilde{y}) \in \mathcal{S}_k^2} \hat{P}_{\tilde{Z}, \tilde{Y}}(\tilde{z}, \tilde{y}) \log \frac{\hat{P}_{\tilde{Z}, \tilde{Y}}(\tilde{z}, \tilde{y})}{\hat{P}_{\tilde{Z}}(\tilde{z}) \hat{P}_{\tilde{Y}}(\tilde{y})}, \quad (7.18)$$

with the empirical marginal probabilities:

$$\hat{P}_{\tilde{Z}}(\tilde{z}) := \sum_{\tilde{y} \in \mathcal{S}_k} \hat{P}_{\tilde{Z}, \tilde{Y}}(\tilde{z}, \tilde{y}), \quad \hat{P}_{\tilde{Y}}(\tilde{y}) := \sum_{\tilde{z} \in \mathcal{S}_k} \hat{P}_{\tilde{Z}, \tilde{Y}}(\tilde{z}, \tilde{y}). \quad (7.19)$$

In words, the model of the distributions  $(\mathbb{P}_0, \mathbb{P}_1)$  is replaced with empirical frequencies learned on the fly. Resorting to the empirical mutual information to decode transmitted messages in digital communication is known as Maximum Mutual Information (MMI), recently proven universally optimal [223].

The empirical mutual information is a kind of similarity (the bigger, the more  $\tilde{Z}$  looks like  $\tilde{Y}$ ). Its value lies in the interval  $[0, \min(\hat{H}(\tilde{Z}), \hat{H}(\tilde{Y}))]$  with the empirical entropy given by:

$$\hat{H}(\tilde{Z}) := - \sum_{\tilde{z}} P_{\tilde{Z}}(\tilde{z}) \log P_{\tilde{Z}}(\tilde{z}). \quad (7.20)$$

We prefer dealing with a normalized distance and we introduce:

$$\text{dist}(\mathbf{b}, \mathbf{m}) := 1 - \frac{\hat{I}(\tilde{Z}; \tilde{Y})}{\hat{H}(\tilde{Y}, \tilde{Z})} \in [0, 1]. \quad (7.21)$$

This defines the Rajsiki distance [188] between the models  $\mathbf{b}$  and  $\mathbf{m}$  respectively producing  $\tilde{Z}$  and  $\tilde{Y}$ .

The distances between all the pairs of models is shown in Section B.2 in Figure B.1. The

block diagonal shows that the distances between variants of the same vanilla models are small. We clearly see the cluster of variants centered on each vanilla model. Indeed, Figure 7.1 in the introduction is a t-SNE representation extracted from such pairwise distances between models in  $\mathcal{B}$ .

For a given model  $m$ , let us consider two extreme scenarios:

- The model  $m$  is in the black-box so that  $\tilde{z}_j = \tilde{y}_j, \forall j \in \llbracket L \rrbracket$ . Then  $P_{\tilde{Z}, \tilde{Y}}(\tilde{z}, \tilde{y}) = 1$  if  $\tilde{z} = \tilde{y}$ , and 0 otherwise, producing  $\text{dist}(b, m) = 0$ .
- The black-box and model  $m$  yield independent outputs so that  $P_{\tilde{Z}, \tilde{Y}}(\tilde{z}, \tilde{y}) = P_{\tilde{Z}}(\tilde{z})P_{\tilde{Y}}(\tilde{y})$ , then  $\text{dist}(b, m) = 1$ .

In the end, Alice deemed the hypothesis  $\mathcal{H}_1$  as being true when the distance is small enough:  $\text{dist}(b, m) < \tau \rightarrow \mathcal{H}_1$  is true. Alice makes two kinds of errors:

- False positive:  $\text{dist}(b, m) < \tau$  whereas  $\mathcal{H}_1$  is false.
- False negative:  $\text{dist}(b, m) \geq \tau$  whereas  $\mathcal{H}_1$  is true.

Alice sets the threshold  $\tau$  such that the probability of false positive is lower than a required level  $\alpha$ . The converse, *i.e.* controlling the probability of false negative, is an illusion. Appendix D shows for instance that there is no way to theoretically upper bound the distance between a variant and its original model, even if both of them share good accuracy. Our working assumption is that this mutual information is indeed large enough for a reliable hypothesis test and the experimental work confirms this in Section 7.4.4.

### **Selection of Inputs**

The empirical mutual information is a consistent estimator of the mutual information which depends on the channel transition matrix  $W_\theta$  and the input probability distribution  $P_{\tilde{Y}}$ . A result of the theory of communication is that for a given transmission channel, there is an input probability which maximises the mutual information. This is of utmost importance to design a communication system achieving the channel capacity as defined by C.E. Shannon. In our framework, this would make the distance between a model and its variant closer to 0 likely avoiding a false negative.

However, this idea is not applicable to our scheme because Alice may suspect a plurality of variants, each of them leading to a different optimal input distribution. The black-box may also contain an unknown variant excluding any optimization.

Yet, when Alice chooses random inputs, she has the feeling that these inputs must not be too easy to be classified otherwise any model outputs the same prediction. This is not discriminative of a given model in the black-box and it may lead to a false positive. On the other hand, these

inputs must not be too hard to be classified neither otherwise the prediction tends to be random, destroying the correlation between a model and its variant. This may lead to a false negative.

Our experimental work investigates several selection mechanisms of the inputs. All of them amount to randomly pick inputs from a subset  $\mathcal{X}'$  of  $\mathcal{X}$ .

- All. There is indeed no selection and  $\mathcal{X}' = \mathcal{X}$ .
- 50/50. Alice’s hypothesis concerns a family of variants derived from a vanilla model  $m$ .  $\mathcal{X}'$  is composed of 50% of inputs well classified by  $m$  (*i.e.*  $m(x) = c(x)$ ), 50% inputs for which  $m(x) \neq c(x)$ .
- 30/70. The same definition but with 30% well classified and 70% wrongly classified by  $m$ .
- Entropy.  $\mathcal{X}'$  is composed of the inputs whose top-1 predictions are highly random. For a given input, Alice computes the predictions from all the models in  $\mathcal{A}$  and measures the empirical entropy of these predicted labels. She then sorts the inputs of  $\mathcal{X}$  by their entropy, and  $\mathcal{X}'$  contains the head of this ranking.

The second and third options are dedicated to the detection task since they only need the vanilla model  $m$  at the root of Alice’s hypothesis. The last selection mechanism demands a long pre-processing step depending on how big the set of models  $\mathcal{A}$  is. It is dedicated to the identification task.

In the worst-case, a model consistently predicts the ground truth at the same top- $k$  position for all submitted images. This situation results in  $P_{\tilde{Y}}(\tilde{y} = k) = 1$  leading to a null entropy and an undefined distance  $\text{dist}$ . To mitigate this problem, Alice adopts a strategy where she ensures that at least one correct and one incorrect classification are selected for all models, following the introduced selection process. It highlights the limitation of our method for fingerprinting models achieving perfect accuracy.

### 7.4.3 Identification ( $l, \mathcal{F}, \mathcal{A} \subsetneq \mathcal{B}, k$ )

The identification task is nothing more than an extension of the detection. Instead of a binary hypothesis, Alice is now facing a multiple hypotheses test with  $M + 1$  choices:

- $\mathcal{H}_i$ : The black-box is a variant of vanilla model  $m_i$ , with  $1 \leq i \leq M$ ,
- $\mathcal{H}_0$ : The black-box is a variant of an unknown model.

The usual way is to compute distance  $\text{dist}(b, m_i)$  per vanilla model  $m_i \in \mathcal{A}$ , and to decide for model  $i^* = \arg \min_{1 \leq i \leq M} \text{dist}(b, m_i)$ , if  $\text{dist}(b, m_{i^*})$  is lower than a threshold, otherwise Alice chooses hypothesis  $\mathcal{H}_0$ . If a known model is in the black-box, only three events may occur:

- Alice makes a correct identification,

- Alice can not make any decision. She deems  $\mathcal{H}_0$  as true.
- Alice makes a wrong identification.

Again, by fine-tuning the threshold, Alice controls the probability of the last event. Note that the probability of success is expected to be smaller than for the previous task. Identification is more difficult since several hypotheses are competing.

### Compound Model

Information theory helps Alice again thanks to an analogy with the communication over a compound channel. In this communication problem, a message  $m_i$  has been emitted and transmitted through a channel  $W_\theta$ . The receiver knows a compound channel, *i.e.* a set of channels  $\{\theta_j\}_{j=1}^V \subset \Theta$ . It knows that the received signal has gone through one of them, but it does not know which one. There exists an optimal decoder for each channel in the set. The receiver just does not know which one to use. A theoretically grounded decoder is to decode the signal with each decoder and to aggregate this decoding with a min operator [1].

The analogy is the following: the inputs go through all the models  $\{m_i\}$  known by Alice, and the outputs are like messages. Bob has chosen one model, *i.e.* one of these messages. Yet, Bob uses a variant which emits noisy outputs observable to Alice. Now, suppose that Alice knows a set of variants in a given family:  $\{V(m_i, \theta_j)\}_j \subset \mathcal{F}$ . She uses these variants for computing distances  $\text{dist}(b, V(m_i, \theta_j))$  that she aggregates into one distance w.r.t. the family:

$$\text{dist}(b, \mathcal{F}) := \min_j \text{dist}(b, V(m_i, \theta_j)). \quad (7.22)$$

Intuitively, the black-box might be a very degraded version of a model which is indeed ‘closer’ to a milder variant than to the original model  $m_i$ .

#### 7.4.4 Experimental Work

The previous experimental work in Section 7.3.3 considers three kinds of family concerning the black-box as defined in Section 7.2.2. When the family is a singleton, because  $\mathcal{F} = \mathcal{F}(m, \{\theta\})$  or  $\mathcal{F} = \mathcal{F}(m, \Psi)$  and  $|\Psi| = 1$ , then the distance between the black-box and this unique model is exactly zero. This easy case is now excluded to focus on cases where Alice does not know the variant in the black-box.

Contrary to Section 7.3.3, Alice now resorts to statistical tests. Any distance between models is a random value since the queries are randomly selected. Our experimental protocol makes 20 measurements of any considered distance thanks to 20 independent inputs samples.

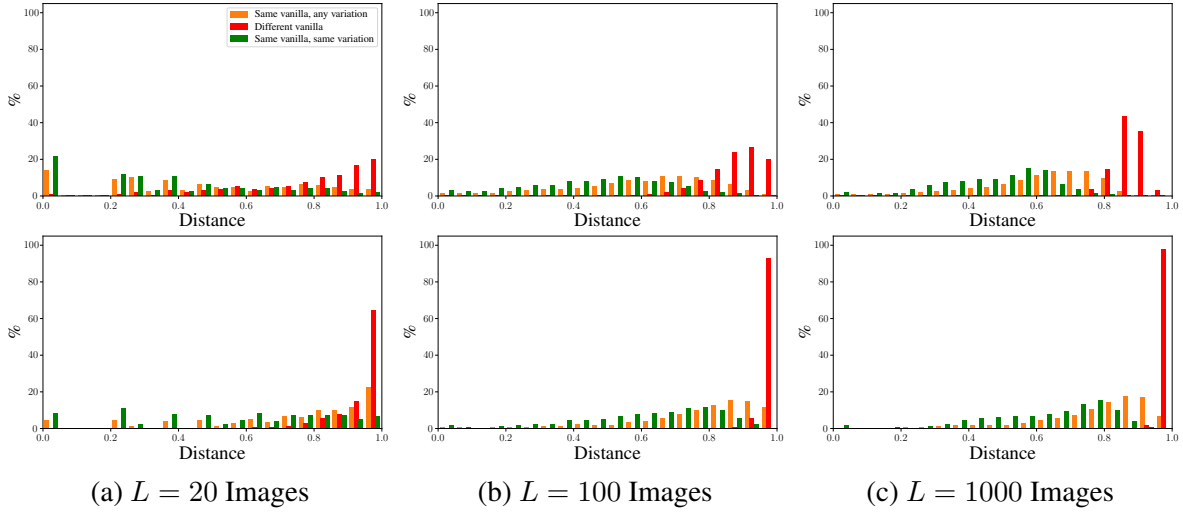


Figure 7.5 – Histogram of the distance  $\text{dist}(m_1, m_2)$  when  $(m_1, m_2) \in \mathcal{F}^2(m)$  (orange),  $(m_1, m_2) \in \mathcal{F}^2(m, \Psi)$  (green), or  $m_1$  and  $m_2$  are variants of different vanilla models (red). Inputs randomly sampled in  $\mathcal{X}$  (*top*) or in  $\mathcal{X}'$  with Entropy (See Section 7.4.2) (*bottom*).

### Assumptions about the Statistical Model

Section 7.4.1 makes two assumptions about the statistical dependence between the predictions of models in the same family  $\mathcal{F}$  and independence when coming from different families. Figure 7.5 experimentally verifies these working assumptions.

The distances between two models  $V(m, \theta)$  and  $V(m', \theta')$  for two vanilla models  $m$  and  $m'$  and any variants  $(\theta, \theta') \in \Theta \times \Theta$  are computed. This sums up to 583,740 combinations. Figure 7.5 shows the histogram of these distance values over 20 bins in red. A high number  $L$  of queries makes the measured distance more precise. The selection of the inputs has a major impact. When sampled on  $\mathcal{X}$  (first row), the distance rarely values the maximum showing imperfect independence. This phenomenon has been revealed in [149]. Yet, when sampled on  $\mathcal{X}'$  containing more inputs hardly correctly classified (second row), the distances are closer to one. The models tend to be independent when queried with a good selection of inputs.

Figure 7.5 also shows the histogram of distances between models belonging to the same family spanned by a vanilla model  $m$ , be it  $\mathcal{F}(m, \Psi)$  (same type of variation) or  $\mathcal{F}(m)$  (any kind of variation). It is not possible to get a non-trivial upper bound of the distance in this case (explained in Section B.4). We empirically observe that two models from the same type of variation are usually closer. It is therefore easier to detect or identify families  $\mathcal{F}(m, \Psi)$  than  $\mathcal{F}(m)$ .

**Detection** ( $\text{det}, \mathcal{F}, \mathcal{A} \subsetneq \mathcal{B}, k$ )

The experiment considers all combinations of hypothesis and model put in the black-box. There are 35 vanilla models and 1046 variants. This makes 35 families of type  $\mathcal{F}(m)$  with an average of 30 members per family. This represents 1081 positive cases and 36,754 negative cases. There are 377 families of type  $\mathcal{F}(m, \Psi)$  of which 203 with a size bigger than 1. This makes 907 positive cases and 218,536 negative cases. To assess the detection performance, Alice leverages the negative cases to determine the threshold  $\tau$  as the empirical  $\alpha$ -quantile of the False Positive Rate (FPR) (see Section 7.4.2).

**Selection of Inputs** Table 7.2 shows the TPR obtained when the black-box returns only *top-1* decisions. As expected, the performances for the families  $\mathcal{F}(m, \Psi)$  are higher. The selection Entropy is clearly the best option. Its drawback is that it needs statistics about the predictions of many vanilla models. As far as the detection task is concerned, the other selections are to be preferred. They only require the predictions of the suspected vanilla model. In the sequel, the selection 30/70 is used for further experiments on the detection task.

**The Delegate Model** Alice measures a single distance in between the black-box and a delegate model of the hypothesis' family  $\mathcal{F}$ . Which member of the family is the best delegate? Three choices are proposed based on the distance to the vanilla model spanning the family: *Close*, *Median*, and *Far*. For instance, the *Close* option means that the delegate is the closest member in the family to the vanilla model:

$$m_d = \arg \min_{m' \in \mathcal{F}} \text{dist}(m, m'). \quad (7.23)$$

In the case where  $\mathcal{F} = \mathcal{F}(m)$ , the closest member is  $m$ . It is not the case when  $\mathcal{F} = \mathcal{F}(m, \Psi)$ , because the vanilla model  $m$  is not in this family. Recall that the intersection between two families has to be the empty set, otherwise Alice could not distinguish them.

Table 7.3 evaluates the three options. Only the 180 families with more than 3 members are

Table 7.2 – True Positive Rate for  $(\text{det}, \mathcal{F}, \mathcal{A} \subsetneq \mathcal{B}, 1)$  with  $L = 100$  queries sampled in  $\mathcal{X}'$  (See Section 7.4.2). The delegate model is the closest to  $m$ . False Positive Rate is set to 5%.

|                        | All            | 50/50          | 30/70          | Entropy                          |
|------------------------|----------------|----------------|----------------|----------------------------------|
| $\mathcal{F}(m)$       | $83.4 \pm 1.4$ | $92.6 \pm 1.0$ | $92.8 \pm 0.8$ | <b><math>94.7 \pm 0.7</math></b> |
| $\mathcal{F}(m, \Psi)$ | $86.9 \pm 0.9$ | $95.3 \pm 0.9$ | $95.6 \pm 0.8$ | <b><math>97.2 \pm 0.5</math></b> |



considered here. For smaller families, the three options would give the same delegate.

The delegate greatly influences the results. The best choice is to select the delegate as lying at the ‘center’ of the family. It means the *Close* option for the family  $\mathcal{F}(m)$ , which is indeed the vanilla model  $m$ , or the *Median* option for family  $\mathcal{F}(m, \Psi)$ .

**Top- $k$  Observations** The detection is evaluated for top- $k$  outputs in Figure 7.6. The best results are surprisingly obtained for  $k = 1$  in Table 7.4 for a few queries. As the number of queries increases, the performance of  $k = 3$  surpasses it and all top- $k$  values converge to the same score after 500 queries. Our explanation is the following. The bigger  $k$  the richer the model. Yet, the empirical mutual information is calculated from  $(k + 1)^2$  estimated probabilities. For a given number of queries, the fewer estimations the more accurate they are. The top-1 is faster to estimate and reach good results quickly. Once the number of queries is enough, the top-3 takes the lead. They quickly get very good results close to 100%, simultaneously as top-5.

To summarize, the TPR reaches 95% for 150 queries under top-1, 110 under top-3, and 140 for top-5.

### Identification $(l, \mathcal{F}, \mathcal{A} \subsetneq \mathcal{B}, k)$

All conclusions obtained in the previous section are kept. Alice now has for delegate the vanilla model  $m$  for  $\mathcal{F}(m)$  and the *Median* model for  $\mathcal{F}(m, \Psi)$ . Images are sampled with Entropy as defined in Section 7.4.2.

**Experimental Protocol** We divide the identification task into three steps, each of them being prone to errors.

In the first step, Alice decides whether to abstain or proceed with identification. In the negative case where  $b \in \mathcal{F}(m')$  but  $m' \notin \mathcal{A}$ , the correct answer is to abstain and to consider the

Table 7.3 – True Positive Rate for  $(\text{det}, \mathcal{F}, \mathcal{A} \subsetneq \mathcal{B}, k)$  and different delegates with  $L = 100$  queries in 30/70, FPR = 5%.

| Delegate               |       | Close             | Median             | Far        |
|------------------------|-------|-------------------|--------------------|------------|
| $\mathcal{F}(m)$       | top-1 | <b>92.8 ± 0.8</b> | 91.3 ± 1.1         | 31.6 ± 2.8 |
|                        | top-3 | <b>94.2 ± 0.9</b> | 93.5 ± 0.2         | 36.8 ± 4.9 |
|                        | top-5 | <b>93.2 ± 0.7</b> | 92.9 ± 0.8         | 35.4 ± 3.5 |
| $\mathcal{F}(m, \Psi)$ | top-1 | 95.6 ± 0.8        | <b>96.3 ± 0.7</b>  | 82.2 ± 1.2 |
|                        | top-3 | 96.3 ± 0.6        | <b>97.7 ± 0.37</b> | 86.1 ± 1.3 |
|                        | top-5 | 96.0 ± 0.6        | <b>97.6 ± 0.4</b>  | 85.7 ± 0.9 |

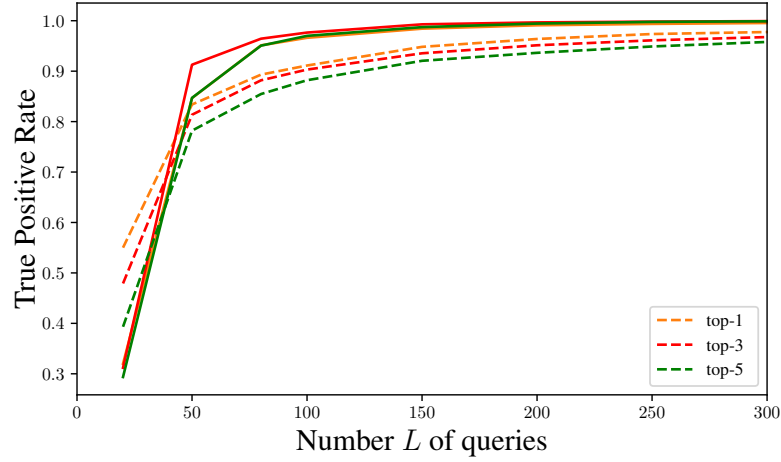


Figure 7.6 – True Positive Rate for  $(\text{det}, \mathcal{F}, \mathcal{A} \subsetneq \mathcal{B}, k)$  function of the number of queries randomly selected in 30/70, FPR = 5%, best delegate options for  $\mathcal{F}(m)$  (dash) and  $\mathcal{F}(m, \Psi)$  (plain).

null hypothesis  $\mathcal{H}_0$ . If  $b$  belongs to  $\mathcal{F}(m)$  and  $m \in \mathcal{A}$ , the correct answer is to move to the next step of identification. We set the probability of error in the negative cases to 5% by controlling the threshold  $\tau$ . Alice abstains if all distances are above the threshold. For this purpose,  $\mathcal{A}$  consists of 30 models, while the remaining 5 models are used to generate the negative cases. Alice computes the distances between  $b$  and the 30 vanilla models in  $\mathcal{A}$ . This process is repeated 20 times, with a random selection of 5 excluded models from  $\mathcal{P}$ .

Once Alice decides that the black box is identifiable, the second step is to disclose the family  $\mathcal{F}(m_i)$ . She decides for the hypothesis  $\mathcal{H}_i$  minimizing the distance. When multiple models achieve this minimum distance, Alice is unable to make a decision and chooses to abstain. This conservative choice is more likely to occur when few images are submitted.

Finally, Alice identifies the variation, knowing she has made a correct identification of the

Table 7.4 – True Positive Rate for  $(\text{det}, \mathcal{F}, \mathcal{A} \subsetneq \mathcal{B}, k)$  with random queries selected with 30/70, FPR = 5%.

| Number of queries      |       | $L = 20$    | $L = 50$    | $L = 100$   | $L = 500$   |
|------------------------|-------|-------------|-------------|-------------|-------------|
| $\mathcal{F}(m)$       | top-1 | <b>79.7</b> | 86.9        | 92.8        | <b>99.4</b> |
|                        | top-3 | 77.3        | <b>88.0</b> | <b>94.2</b> | 99.3        |
|                        | top-5 | 76.8        | 87.3        | 93.2        | 99.3        |
| $\mathcal{F}(m, \Psi)$ | top-1 | 83.1        | 91.5        | 96.3        | <b>99.7</b> |
|                        | top-3 | <b>84.3</b> | <b>94.1</b> | <b>97.7</b> | <b>99.7</b> |
|                        | top-5 | 83.6        | 94.0        | 97.6        | 99.6        |

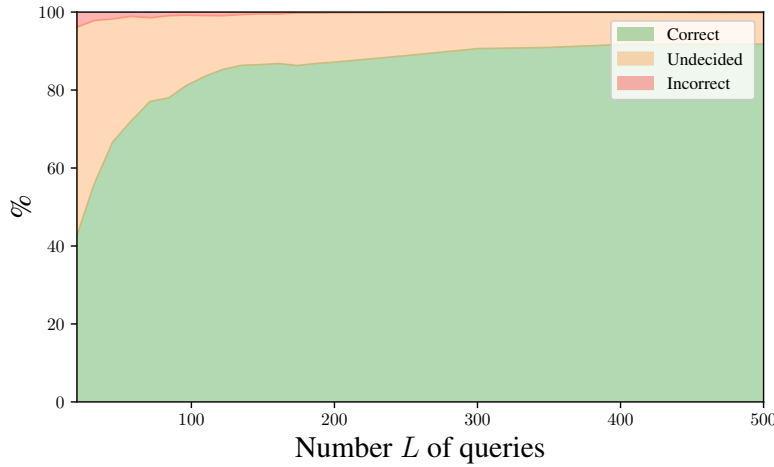


Figure 7.7 – Probability distribution for  $(l, \mathcal{F}(m), \mathcal{A} \subsetneq \mathcal{B}, 1)$  vs. number  $L$  of queries. Threshold set to have a maximum 5% errors in negative cases.

global family  $\mathcal{F}(m_i)$ . In this case, Alice has to identify the correct variation among 6 families  $\{\mathcal{F}(m, \Psi_j)\}_{j=1:6}$ : randomized smoothing, pruning (filter, all, last), JPEG, posterize (See App. A). Alice thus computes 6 distances based on their delegates and identifies the family  $i^* = \arg \min_j \text{dist}(b, \mathcal{F}(m, \Psi_j))$ . No thresholding is needed here. For each family, 20 variants with random parameters and complying with (7.1) are created. This leads to 700 new models tested in the black-box, different from the 1,081 models considered so far.

**Identifying  $\mathcal{F}(m)$**  Alice almost surely identifies the family  $\mathcal{F}(m)$  of the black-box as shown in Figure 7.7 and Table 7.5. She reaches her maximum success rate at around 300 queries. After 200 queries, no incorrect identification is made but 10% of abstention remains. This is due to the thresholding which prevents Alice from misclassification in the negative case. If no thresholding is done, the success rate reaches 94.7% within 100 queries and 99.1% at 500.

The number of queries is higher than for detection. For equivalent performance, 4 times more queries are necessary for identification than for detection. Nevertheless, identification proceeded by sequential detection would take on average 3.000 queries (24 times more w.r.t. detection) as foreseen by (7.9).

**Identifying  $\mathcal{F}(m, \Psi)$**  With a single delegate, Table 7.5 and Figure 7.8 show a rather difficult identification. Variants far from the vanilla model are correctly identified. The main difficulty comes from the variations that slightly modify the model. These variants are close to  $m$ , which is the center of the cluster  $\mathcal{F}(m)$  (see Figure 7.1), therefore it is hard to distinguish them. The

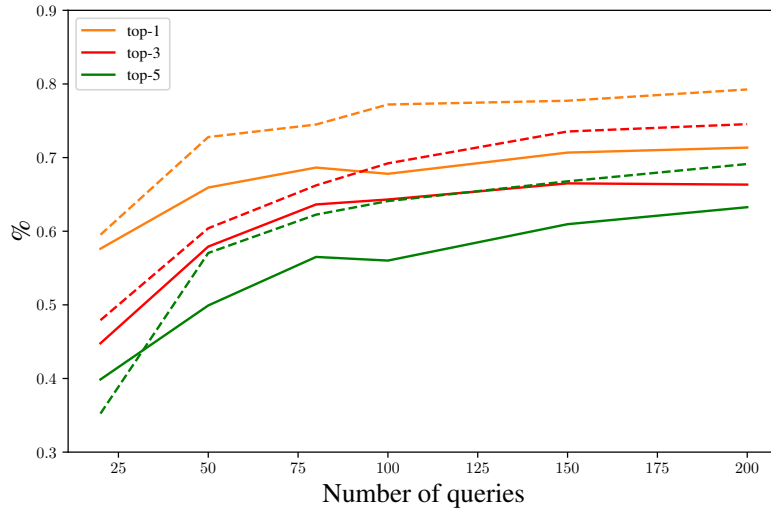


Figure 7.8 – Correct Identification Rate for  $\mathcal{F}(m, \Psi)$  as a function of the number of queries. One (plain) or two (dashed) delegates per family.

compound (7.22) with the median and the close delegates yields a boost if  $L$  is large enough.

**Top- $k$  Observations** The best results are obtained for  $k = 1$  in Table 7.5 on every task, like for detection. For the family  $\mathcal{F}(m)$ , the information gained by top- $k$  needs too many queries to catch up with the top-1. For family  $\mathcal{F}(m, \Psi)$ , the difference is smaller. Indeed, top- $k$  with  $k \leq 3$  gives slightly better results from  $\approx 1,000$  queries and above.

## 7.5 State-of-the-Art Benchmark

### 7.5.1 Previous Works

Since the work of IP-Guard [19], all the fingerprinting papers leverage adversarial examples. They start with a small collection of benign inputs (except [247] starting from random noise images) and apply a white-box attack like CW [22]. It forges adversarial examples that lie close to the decision boundaries, which are the signatures of a model.

Two trends are connected to two applications. The first one deals with the integrity of the model. In this scenario, Alice makes sure that Bob placed her model in the black-box without any alteration. The goal is to sense a *fragile fingerprint* such that any modification of the vanilla model is detectable because it changes the fingerprint. In that light, methods in [85, 119] create sensitive examples which are triggered only by modifications of the vanilla model.

The second application is *robust fingerprint* as considered so far in this chapter. The followers of IP-Guard [19] forge adversarial examples which are more robust in the sense that they remain adversarial for any variation of the model while being more specific to the vanilla model. Paper [178] proposes to use the universal adversarial perturbations of the vanilla model. Paper [140] introduces the concept of conferrable examples, *i.e.* adversarial examples which only transfer to the variations of the targeted model. AFA [284] activates dropout as a cheap surrogate of variants when forging adversarial examples. TAFA [172] extends this idea to other machine learning primitives.

Our take in this article is that using benign images is sufficient, and we addressed the fingerprinting problem without the need to rely on adversarial examples or any other technique to alter images to get them nearby the boundaries. Indeed, crafting adversarial examples is rather simple but forging them with extra specificities (fragile or robust to variation) is complex. It happens that all above-mentioned papers consider small input dimensions like MNIST or CIFAR ( $32 \times 32$  pixel images); none of them use ImageNet ( $224 \times 224$ ) except IP-Guard [19]. Also, no paper considers that the inputs can be reformed by a defense (in order to remove an adversarial perturbation before being classified) or detected as adversarial [111].

## 7.5.2 Fragile Fingerprinting

The application considered in [85] imagines that Alice wants to detect whether the black-box is exactly  $m$  and not a variant. This corresponds to our scenario  $(\text{det}, \mathcal{F}(m, \{\theta\}), \mathcal{A} = \mathcal{B}, 1)$  where  $\theta$  is the identity variation, and  $\mathcal{A} = \mathcal{F}(m)$ .

We create  $L = 20$  sensitive examples per model with 200 iterations and two distortion budgets ( $\epsilon = 8/255$  and  $16/255$ ) using the code<sup>1</sup> released by [85]. It happens that its performance on ImageNet (reported in Table 7.6) is lower than the one reported in [85] on small input size datasets (like CIFAR). Especially, this scheme can not distinguish the vanilla model and its variants ‘JPEG’ or ‘Half precision’ even with a number of queries ( $L = 20$ ) bigger than the one recommended ( $L = 8$ ) in [85]. Our scheme needs no more than *two* queries and perfectly accurate, except when pruning the last layer for five out of thirty-five models.

---

1. Sensitive Examples’ GitHub: [https://github.com/zechenghe/Sensitive\\_Sample\\_Fingerprinting](https://github.com/zechenghe/Sensitive_Sample_Fingerprinting)

### 7.5.3 Robust Fingerprinting

This application is related to our scenario  $(\text{det}, \mathcal{F}(m), \mathcal{A} \subsetneq \mathcal{B}, k)$ . `IP-Guard` [19] is the only work showing to be tractable and effective on large input size like in ImageNet. It leverages several white-box attacks to create adversarial examples. The best results demonstrated in the paper are with the attack `CW` [22]. We instead use `BP` [280, 15] because it exhibits similar performances while being much faster (only 50 iterations). The `BP` implementation is from GitHub<sup>2</sup>.

Table 7.7 compares the performances under 100 and 200 queries and top-1 observations. Any selection of the inputs beats `IP-Guard` [19]. Detailed results are reported in Table 7.8. Some variations are easier to detect (‘precision’, ‘pruning’) and the two methods are on par. On the contrary, randomized smoothing which is a popular variation yet never considered in the literature, is more difficult. `IP-Guard` [19] is based on crafting adversarial examples close to the decision boundaries which are greatly crumpled by randomized smoothing. Not relying on adversarial examples seems to be a clear advantage in this case. Our method offers more stability in the results: No variation pulls the TPR below 85%.

## 7.6 Related Work

Having reviewed the previous works dedicated to the fingerprinting task in the section above, we here review another closely related domain: the *watermarking* of models.

Watermarking is the active counterpart of fingerprinting: Instead of relying on specifics of a fixed model to devise its unique fingerprint, watermarking modifies the model for which ownership must be proven. While watermarking is a common practice for decades in the field of image processing [220], it has just recently been incepted into the machine learning domain. Uchida et al. [236] first proposed to watermark a deep neural network by embedding it into the weights and biases of the model. Quickly after this initial proposal, works instead focused on a black-box model, where the presence of a watermark can be assessed by Alice from remote interaction with the suspected deep neural network, just like for fingerprinting. In [120], authors insert information by altering the decision boundaries through finetuning. In [2], authors also retrain the model to obtain the wrong labels for a so-called *trigger set* of inputs, that constitutes the watermark. Please refer to [135] for a complete overview of the domain.

---

2. Boundary Projection’s GitHub: <https://github.com/hanwei0912/walking-on-the-edge-fast-low-distortion-adversarial-examples>

Some papers claim that robust fingerprinting could replace watermarking with the clear advantage that no modification of the model is needed [284]. We strongly disagree. No fingerprinting scheme, including ours, brings any formal guarantee on what is in the black-box. It is indeed the primary goal of watermarking to guarantee a certain level of trust.

## 7.7 Conclusion

The problem of accurate and efficient fingerprinting of valuable models is salient. This chapter demonstrates that such a demand can be fulfilled by solely using benign inputs, in not only the classic detection task, but also in the novel identification task we have introduced. This has the important implication that we no longer need models in white-box access to compute their fingerprints.

We provide the following takeaways.

*i)* In the walled garden setup of Section 7.3, less than ten inputs are needed but these are sequentially and carefully selected among a large collection depending on the previous outputs of the black-box. In other words, the key is the interaction between the greedy algorithm and the black-box. Observing top-1, top-3 or top-5 makes a difference. It is easier to spot inputs that single out a model with richer outputs.

*ii)* In the open-world setup of Section 7.4, hundreds of inputs are necessary but the scheme is not iterative and selection is less crucial. Surprisingly, observing richer outputs does not yield any gain in this setup.

*iii)* The identification task is merely more complex than detection. Our identification schemes are much more efficient than the naive sequential search.

*iv)* Bob's best defenses in our experimental protocol are randomized smoothing for robust fingerprinting and pruning the last layer for fragile fingerprinting. It means that the former reduces the statistical dependence of the outputs, while the latter hardly perturbs the outputs given by the vanilla model.

One limitation of our work is that it cannot handle classifiers whose accuracy is almost perfect. This would happen for too easy classification setups where the value of models is lower, and fingerprinting is less critical. We nevertheless expect future models and applications to continue to be complex tasks, where reaching high accuracy levels will remain a struggle for both the academia and industry.

Table 7.5 – Correct Identification Rate for  $(l, \mathcal{F}, \mathcal{A} \subsetneq \mathcal{B}, k)$  with random queries selected with Entropy.

| Number of queries                                    |       | $L = 50$    | $L = 100$   | $L = 500$   |
|--|-------|-------------|-------------|-------------|
| $\mathcal{F}(m)$<br>delegate = {close}               | top-1 | <b>67.1</b> | <b>80.0</b> | <b>98.6</b> |
|  | top-3 | 49.1        | 57.8        | 85.3        |
|  | top-5 | 48.4        | 55.7        | 80.4        |
| $\mathcal{F}(m, \Psi)$<br>delegate = {median}        | top-1 | 65.8        | 68.3        | 74.1        |
|  | top-3 | 58.2        | 64.5        | 71.4        |
|  | top-5 | 52.7        | 57.2        | 69.2        |
| $\mathcal{F}(m, \Psi)$<br>delegate = {close, median} | top-1 | <b>73.1</b> | <b>77.2</b> | <b>83.6</b> |
|  | top-3 | 61.8        | 70.0        | 80.2        |
|  | top-5 | 60.4        | 66.3        | 78.5        |

Table 7.6 – False Positive Rate for the  $(\text{det}, \mathcal{F} = \{m\}, \mathcal{A} = \mathcal{B}, 1)$  task.

| Method               |        | Sensitive Examples [85] |                     | FBI [147]   |
|----------------------|--------|-------------------------|---------------------|-------------|
| Parameter            |        | $\epsilon = 8/255$      | $\epsilon = 16/255$ | $L = 2$     |
| Finetuning           | All    | 18.1                    | 21.4                | <b>0</b>    |
|                      | Last   | 20.3                    | 1.8                 | <b>0</b>    |
| Prune                | All    | 0                       | 0                   | <b>0</b>    |
|                      | Filter | 35.7                    | 31.0                | <b>0</b>    |
|                      | Last   | 15.4                    | 10.0                | <b>0</b>    |
| Half Precision       |        | 31.5                    | 21.4                | <b>16.9</b> |
| Histogram            |        | 57.4                    | 48.5                | <b>0</b>    |
| JPEG                 |        | 77.0                    | 63.4                | <b>0</b>    |
| Posterize            |        | 100                     | 97.1                | <b>0</b>    |
| Randomized Smoothing |        | 100                     | 97.1                | <b>0</b>    |

Table 7.7 – True Positive Rate for  $(\text{det}, \mathcal{F}(m), \mathcal{A} \subsetneq \mathcal{B}, 1)$ , FPR set to 5%.

| Fingerprinting scheme | Parameter               | Number of queries |             |
|-----------------------|-------------------------|-------------------|-------------|
|                       |                         | $L = 100$         | $L = 200$   |
| IP-Guard [19]         | BP [280, 15] & 50 iter. | 66.9              | 72.7        |
| FBI [147]             | Random                  | 83.4              | 95.0        |
|                       | 30/70                   | 92.8              | 97.8        |
|                       | Entropy                 | <b>94.7</b>       | <b>98.0</b> |



Table 7.8 – True Positive Rate per variation under  $(\text{det}, \mathcal{F}(m), \mathcal{A} \subsetneq \mathcal{B}, 1)$ . False Positive Rate set to 5% and  $L = 100$  queries.

| Method               |        | IP-Guard [19]                 | FBI [147]  |             |             |
|----------------------|--------|-------------------------------|------------|-------------|-------------|
| Parameter            |        | BP [280, 15]<br>50 iterations | Random     | 30/70       | Entropy     |
| Finetuning           | All    | 0.5                           | 85.6       | <b>94.5</b> | <b>94.5</b> |
|                      | Last   | 92.3                          | 91.5       | <b>97.3</b> | <b>97.3</b> |
| Prune                | All    | 72.7                          | 65.0       | 87.4        | <b>91.9</b> |
|                      | Filter | 89.2                          | 87.8       | 97.3        | <b>99.5</b> |
|                      | Last   | <b>100</b>                    | 87.3       | 97.9        | <b>98.9</b> |
| Half Precision       |        | <b>100</b>                    | <b>100</b> | <b>100</b>  | <b>100</b>  |
| Histogram            |        | 27.3                          | 64.2       | 89.4        | <b>92.0</b> |
| JPEG                 |        | <b>100</b>                    | <b>100</b> | <b>100</b>  | <b>100</b>  |
| Posterize            |        | 9.2                           | 88.3       | 95.9        | <b>98.5</b> |
| Randomized Smoothing |        | 26.1                          | 60.0       | 78.1        | <b>85.5</b> |

# CONCLUSION

---

The primary goal of this thesis is to investigate vulnerabilities in AI within a realistic setup. In this particular setup, the attacker’s capabilities are severely restricted with regard to the usual assumptions made in a white-box setup, both in terms of their knowledge and resources.

In the first chapter, this scenario and its associated definitions were introduced. Remarkably, even within this constrained context, a significant number of vulnerabilities that were originally identified in the white-box literature were also found to be feasible in the black-box setup. As a result, the specific scenario we considered gave rise to three main focal points: model confidentiality, manipulation of inference inputs, and the confidentiality of training data. Each of these areas was briefly introduced, with subsequent emphasis on two significant aspects, namely adversarial examples and model confidentiality, explored in Chapter 2 and Chapter 3, respectively. These are domains in which substantial contributions have been made throughout this thesis.

Chapter 1 also delves into the intricacies of the decision boundary and its characteristics, serving as a gateway for potential attacks. Given the limited information accessible in this scenario, numerous attack strategies leverage the properties of this decision boundary. It essentially mirrors the learning process of the model and its interactions with the data. A profound understanding of the decision boundary equates to a more complete comprehension of the model’s vulnerabilities. For instance, adversarial examples hold a central position in this thesis, occupying a substantial portion, both in the state-of-the-art overview in Chapter 3 and through the numerous contributions presented in part II of this thesis. The fundamental premise of their existence and the plethora of methods introduced underline the critical importance of comprehending the relation between the decision boundaries and the model vulnerabilities.

The focal point of this thesis has been the security of neural networks, specifically concentrating on the task of image classification. This choice is driven by the extensive coverage of these tasks in existing literature. Nevertheless, the threats elucidated throughout this thesis are universally applicable to diverse tasks and datasets. For instance, a recent study [291] demonstrates that the concept of adversarial examples extends to language models. In such contexts, inputs can also be manipulated to produce undesired or forbidden outputs. Moreover, within the scope of the black-box setup, the contents of the black box can vary widely, encompassing a

---

neural network, a Support Vector Machine model, or even a human. As long as the assumptions laid out in this thesis hold, the methodologies discussed can be effectively employed.

## Limitations

This thesis considered a realistic scenario. However, there are reasons to question whether the scenario presented in Section 1.4.2 is genuinely "realistic". Several factors may cast doubt on this scenario and our associated definition. Firstly, the use of ImageNet in the majority of our experiments may be questioned. This is because the images, due to their size and especially their labels, do not necessarily represent true real-world images. For example, within ImageNet, there are 118 different labels for dogs among a total of 1,000 labels. Some classes exhibit high similarity, such as "projectile, missile" and "missile", resulting in frequent errors for both the human labeling of the dataset and the trained model [163, 14]. This raises questions about the applicability of the conclusions of the experiments in our contributions and most works in a real-world context.

Furthermore, the future deployment of models in production is likely to be dynamic rather than static. MLOps, designed to facilitate the integration of machine learning into daily-life applications and adapt to the continuous influx of new data, is gaining prominence. MLOps capabilities enable the automatic and continuous retraining of machine learning models according to a schedule. Consequently, models are subject to frequent changes, making attacks potentially more challenging. For instance, in a black-box setup, attacks based on historical query data may become obsolete. Moreover, the results of a membership inference attack could vary depending on when the attack was conducted.

Another limitation arises from evasion attacks. When crafting adversarial examples, most decision-based black-box attacks involve a line search to locate the decision boundary, as demonstrated in Section 3.4.3. During this phase, the attacker oscillates between the class they wish to evade and the target class. In the world we have defined, there is no distinction between query outcomes; all responses are equal. However, in a more security-sensitive context, this can pose significant risks. In such scenarios, the attacker may generate numerous "undesirable" queries in the process of crafting adversarial examples. For example, in a case outlined in [45], an attacker aiming to upload an image that raises concerns of inappropriate content might employ an evasion attack against a model designed to detect such content. Nevertheless, the current decision-based black-box attack methods are prone to generating these "undesirable" queries, mainly during the line search stage when seeking the decision boundary. A single such query

---

could result in the attacker being banned or, worse, facing legal action. Future developments in decision-based black-box attacks may focus on mitigating these risks by limiting the number of queries associated with a restricted class.

## Perspectives

The realm of adversarial examples stands as the most extensively explored subject within the domain of machine learning security. New methodologies for generating them, novel defense mechanisms, and theoretical insights into their existence emerge frequently. However, it is plausible that these challenges might not possess a universally applicable solution. The inherent characteristics of the high-dimensional input data we deal with, the overparametrization prevalent in most models, and the quality and quantity of available training data collectively contribute to a complex landscape that might not readily yield a definitive resolution. But perhaps the critical question is whether it is inherently problematic to lack a unique solution. Adversarial examples, much like their real-world human counterparts (as exemplified by optical illusions in Chapter 3), might be an inherent aspect of the landscape. The notion might not be to eliminate them outright but rather to coexist with them, unravel their intricacies, and manage them through context-specific strategies.

The previous paragraph inquires about the relevance of adversarial examples and the aspiration to eliminate them. However, this does not question their value. The war between the attacker and the defender will never end but adversarial examples also serve as a valuable tool for gaining insights into the inner workings of neural networks [170]. For instance, let's consider counterfactual examples [103]. These examples reveal what changes are required in an input to achieve a predetermined output. While they are adversarial, their purpose is not to deceive a model but rather to interpret it. Furthermore, it's a win-win relationship; adversarial examples aid in model interpretation, and this interpretation, in turn, enhances our understanding and ability to fight adversarial examples [17, 62]. Thus, research into adversarial examples should persist but from a different perspective, perhaps placing less emphasis on security concerns and more on gaining a deeper understanding.

The aspiration for a comprehensive and equitable evaluation metric has been expressed by the contributions made in this work. The introduction of `ROBIC` [144] in Appendix A and `FIT` in Chapter 7 respectively introduces a fresh measure of model robustness and a framework for quantifying transferability between source and target models. In light of the constant flux of research papers, contemporary benchmarks and objective measures are often absent.

---

In the realm of neural networks, confidentiality assumes paramount importance, particularly concerning both models and data. Clearly, it holds significant importance due to the cost and the competitive advantages in terms of technology. Equally important is the fact that it aligns with customer demands and expectations. A statistic from INSEE reveals that approximately 82% of individuals protect their personal data on the internet<sup>3</sup>. Initiatives like the European Union AI Act and the General Data Protection Regulation (GDPR) underline the imperative of data security. Throughout this thesis, there was not enough time to thoroughly investigate data confidentiality. However, in Section 1.4.3, we introduced membership inference and model inversion attacks as methods for extracting information about the training data within a realistic scenario. To tackle these concerns, efforts have been made to advance solutions that focus on securing, managing, and supervising these invaluable assets.

## Last Words: New Challenges, Old Problems

This thesis is very much attuned to its time. AI is on everyone’s lips, and its security has consequently become a current and pressing topic. However, many of the issues addressed are not novel. Throughout this thesis, we have referenced works that precede the widespread use of neural networks. For instance, the concept of model watermarking draws inspiration from image watermarking techniques dating back to 1990 [226]. Attacks against watermarks can be viewed as adversarial examples, where the objective is to deceive a detector. Defenses against oracle attacks are still up-to-date and relevant against current adversarial attacks but are not considered (See Section 3.5). The notion of network fingerprints was applied earlier, in the realm of websites. There are plenty of examples. Many methods have been reintroduced and applied to neural networks, even though improvements were previously proposed.

Some prior works attempted to establish connections with prior research, serving as a kind reminder of how far we have come and that research on security did not start in 2012. An exemplary instance among the contributions is the `SurFree` [146] method. Its roots can be traced back to the work of John Earl in 2007 [55]. In an effort to bridge the gap between watermarking and machine learning domains, a study by Erwin Quiring *et al.* [183] presented a unified notation for black-box attacks on machine learning and watermarking. They demonstrated the effectiveness of each field’s strategies in their respective counterparts. For instance, they highlighted how countermeasures from watermarking could mitigate recent model-extraction attacks.

In the rush to publish new research, the examination of previous work may be sometimes

---

3. <https://www.insee.fr/fr/statistiques/6475020>

---

neglected. However, delving into the existing literature, even the oldest, is crucial for gaining insights into the origins of problems and proposed solutions, preventing the reinvention of existing solutions. The neglect of prior research can occur due to a lack of familiarity with the literature or even a desire to hide the origins of a particular method. In either case, such neglect is detrimental to the progress of research and the broader scientific community. Navigating this complex landscape can indeed be challenging. Nevertheless, the field of AI security is intricately connected to various other domains. As a result, it is of paramount importance not to disregard the contributions of earlier researchers but rather to leverage them in order to enhance and refine existing methodologies. This approach could yield substantial time savings and improvements.

# LIST OF FIGURES

---

|     |   |    |
|-----|---|----|
| 1.1 | Samples from different image datasets. From left to right: MNIST [229], CIFAR-10 [115], CIFAR-100 [115], and ImageNet [46]. . . . .   | 25 |
| 1.2 | From the idea of the neuron to the MultiLayer Perceptron. . . . .   | 26 |
| 1.3 | Top-5 error evolution for ImageNet Classification Competition. . . . .  | 27 |
| 1.4 | Visualizations of the decision boundary centered on the image. The first row shows the predicted label in the input space. The second row shows the probability of the top-1 label in the input space. . . . .                  | 31 |
| 1.5 | Binary classification using a shallow model with 20 hidden units (solid line) and a deep model with two layers of 10 units each (dashed line) obtained in [154].  | 32 |
| 1.6 | Different setup illustration from [23]. . . . .   | 36 |
| 1.7 | An image obtained with a model inversion attack (left) and a corresponding training set image of the target model (right) from [63]. The attacker has access only to the person’s name and a facial recognition system. . . . . | 40 |
| 1.8 | Number of scientific papers investigating adversarial examples per year . . . .   | 42 |
| 2.1 | Overview of a Model Extraction Attack. . . . .  | 48 |
| 2.2 | Overview of the Data-Free Model Extraction Approach [233]. . . . .  | 50 |
| 3.1 | Adversarial example for human perception. . . . .   | 57 |
| 3.2 | Adversarial example of a panda misclassified as a gibbon [221]. . . . .   | 58 |
| 3.3 | Diagram of the different properties of a single adversarial perturbation. . . . .   | 63 |
| 3.4 | Process of finding universal adversarial perturbations for three images from [155]. Each color represents the classification region of an image. . . . .  | 64 |
| 3.5 | Gradient Estimation for query-based attacks by HSJA [23]. . . . .   | 68 |
| 3.6 | Estimation of the required number of queries as a function of the number of attacked images using CIFAR-10 for transfer-based attacks, surrogate-based attacks, and query-based attacks. . . . .                                | 70 |

---

|     |  |     |
|-----|--|-----|
| 4.1 | The perturbation distortion ( $\ell_2$ norm) vs. the number of queries for image ‘lizard’.<br>The estimation of a gradient surrogate results in plateaus of distortion for GeODA [187],<br>QEBA [126], and HSJA [23]. SurFree [146] avoids this waste of queries with<br>a random walk and geometric approximation. . . . .  | 80  |
| 4.2 | The geometrical configuration of the problem in $\mathcal{P}$ . . . . .  | 83  |
| 4.3 | Interpolation mechanism to refine the boundary point. . . . .  | 88  |
| 4.4 | Ablation study on SurFree [146]. Mean distortion $d(K)$ (4.10) vs. number $K$<br>of queries on MNIST. . . . .  | 90  |
| 4.5 | Ablation study on SurFree [146]. The deviation of the distortion over 20 runs<br>of SurFree [146] on one MNIST image. . . . .  | 91  |
| 4.6 | Benchmark on ImageNet. The amount of queries $K$ ( $x$ -axis) w.r.t. mean distur-<br>tion $\text{dist}(K)$ ( $y$ -axis). . . . .   | 94  |
| 4.7 | Success rate $\text{ASR}(\text{dist}_t, K)$ (4.11) vs. target distortion $\text{dist}_t$ with $K = 500$<br>queries over ImageNet. . . . .  | 94  |
| 5.1 | Evaluation of transferability by comparing the Attack Success Rate vs. distur-<br>tion trade-off of a white box, transferable, and black box attacks against model<br><code>CoatLite<sub>small</sub></code> (See Section 5.3.1 for details). The blue area is the range of<br>trade-offs operated by a transferable attack with random source models. A trans-<br>ferable attack may be worse than a black box attack without a good source se-<br>lection (like <code>FiT</code> ). . . . . | 96  |
| 5.2 | Transferability score $\hat{T}_{s,t}$ matrix of 48 sources and 48 targets listed in App. C.1<br>with attack <code>DI</code> [265]. . . . .   | 100 |
| 5.3 | $\hat{T}_{s,t}$ function of the number of available sources and attack <code>DI</code> [265]. The best<br>or worst model selected per image (solid line) or on average (dashed line). . . .  | 101 |
| 5.4 | Distribution of best image-model selection function of the number of sources.<br>Adversarial examples obtained with <code>DI</code> [265]. . . . .   | 102 |
| 5.5 | 2D Histogram of the minimum distortion for transferable and white box attacks. . . .   | 103 |
| 5.6 | $\hat{T}_{s,t}$ as a function of the number of available sources for several selection meth-<br>ods. Dotted lines refer to the selection of a unique model for all images and<br>solid lines refer to a model selected per image. Attack is <code>DI</code> [265]. . . . .   | 107 |
| 6.1 | Probability of being adversarial by following the direction $\mathbf{x}_{\text{adv}} - \mathbf{x}_o$ , Image $\mathbf{x}_o$<br>attacked with <code>BP</code> [280, 15] to get the best adversarial $\mathbf{x}_{\text{adv}}$ on the boundary of<br><code>ResNet50</code> . . . . .   | 114 |



---

|     |   |     |
|-----|---|-----|
| 6.2 | 2D slice in the image space of ResNet50 with and without RS. Each point is an image, his color represents the elected label. Centered on a real image, 2 random directions are taken. . . . .   | 115 |
| 6.3 | Distribution of the output of a binary search with RS. . . . .  | 116 |
| 6.4 | Certification for ResNet50 and RandTrain. . . . .   | 117 |
| 6.5 | Average adversarial $\ell_2$ distortion vs. Accuracy . . . . .  | 118 |
| 7.1 | A t-SNE representation of the pairwise distances of 1081 different models: 10 types of variation applied on 35 off-the-shelves vanilla models for ImageNet with different parameters (listed in App. B.1). . . . .  | 124 |
| 7.2 | Probability distribution of the number of queries for $(\text{det}, \mathcal{F}, \mathcal{A} = \mathcal{B}, k)$ when the black box returns top- $k$ classes with $k = 1$ ( <i>blue</i> ), $k = 3$ ( <i>red</i> ) or $k = 5$ ( <i>green</i> ). Family considered from left to right: $\mathcal{F}(m)$ (7.3), $\mathcal{F}(m, \Psi)$ (7.4), and $\mathcal{F}(m, \{\theta\})$ (7.5). . . . .   | 133 |
| 7.3 | Probability distribution of the number of queries for $(l, \mathcal{F}, \mathcal{A} = \mathcal{B}, k)$ when the black box returns top- $k$ classes with $k = 1$ ( <i>blue</i> ), $k = 3$ ( <i>red</i> ) or $k = 5$ ( <i>green</i> ). Family considered from left to right: $\mathcal{F}(m)$ (7.3), $\mathcal{F}(m, \Psi)$ (7.4), and $\mathcal{F}(m, \theta)$ (7.5). . . . .  | 134 |
| 7.4 | Average number of queries as a function of the number of families $n_{\mathcal{F}}$ for $(l, \mathcal{F}, \mathcal{A} = \mathcal{B}, k)$ with the expectation score (7.7) and when the black-box returns top- $k$ classes with $k = 1$ ( <i>blue</i> ), $k = 3$ ( <i>red</i> ) or $k = 5$ ( <i>green</i> ). The dotted lines represent the linear regressions (7.13). Family considered from left to right: $\mathcal{F}(m)$ (7.3), $\mathcal{F}(m, \Psi)$ (7.4), and $\mathcal{F}(m, \{\theta\})$ (7.5). . . . . | 135 |
| 7.5 | Histogram of the distance $\text{dist}(m_1, m_2)$ when $(m_1, m_2) \in \mathcal{F}^2(m)$ ( <i>orange</i> ), $(m_1, m_2) \in \mathcal{F}^2(m, \Psi)$ ( <i>green</i> ), or $m_1$ and $m_2$ are variants of different vanilla models ( <i>red</i> ). Inputs randomly sampled in $\mathcal{X}$ ( <i>top</i> ) or in $\mathcal{X}'$ with Entropy(See Section 7.4.2) ( <i>bottom</i> ). . . . .   | 142 |
| 7.6 | True Positive Rate for $(\text{det}, \mathcal{F}, \mathcal{A} \subsetneq \mathcal{B}, k)$ function of the number of queries randomly selected in 30/70, FPR = 5%, best delegate options for $\mathcal{F}(m)$ ( <i>dash</i> ) and $\mathcal{F}(m, \Psi)$ ( <i>plain</i> ). . . . .   | 145 |
| 7.7 | Probability distribution for $(l, \mathcal{F}(m), \mathcal{A} \subsetneq \mathcal{B}, 1)$ vs. number $L$ of queries. Threshold set to have a maximum 5% errors in negative cases. . . . .   | 146 |
| 7.8 | Correct Identification Rate for $\mathcal{F}(m, \Psi)$ as a function of the number of queries. One ( <i>plain</i> ) or two ( <i>dashed</i> ) delegates per family. . . . .  | 147 |

---

|     |  |     |
|-----|--|-----|
| A.1 | Evolution of $D_{1/2}$ with the complexity budget for black box setup. Attacks on EfficientNet [224] . . . . .   | 199 |
| A.2 | Evolution of $D_{1/2}$ with the complexity budget for white box setup. Attacks on EfficientNet [224] . . . . .   | 200 |
| A.3 | Black-box $D_{1/2}$ as a function of white-box $D_{1/2}$ . . . . .   | 202 |
| B.1 | Distances between pairs of models. . . . .   | 205 |
| B.2 | Lower bound of the distance between two models of accuracy $A$ and $B$ under top-1. . . . .  | 207 |
| B.3 | Comparison of the theoretical lower bound and the measured distance between models $m_1$ and $m_2$ , when $A = \eta(m_1) = 0.45$ and $(m_1, m_2) \in \mathcal{F}^2(m)$ (orange), $(m_1, m_2) \in \mathcal{F}^2(m, \Psi)$ (green), or $m_1$ and $m_2$ are variants of different vanilla models (red). . . . . | 208 |
| C.1 | Attack Success Rate function of the perturbation norm for different values of $\epsilon$ . . . . .   | 212 |
| C.2 | Transferability score $\hat{T}_{s,t}$ matrix of 48 sources and 48 targets listed in C.1 for DI [265], TAIG [96] and DWP [244]. . . . .   | 213 |
| C.3 | $\hat{T}_{s,t}$ function of the number of images used for FBI [147] to estimate the transferability between 45 sources and one target. Adversarial obtained with DI [265] and $\epsilon = 8$ . . . . .   | 214 |
| C.4 | $\hat{T}_{s,t}$ function of the number of models used for ensemble-model to attack xCiT <sub>nano</sub> . The models are randomly selected and added one by one and compared with FiT selecting the three best models for ensemble-model among the 20 models available. . . . .                              | 215 |

# NOTATIONS

---

Table 7.9 – General notations.

| Model Relative Notations                                  |   |
|---|---|
| $D$   | Input dimension   |
| $N_c$   | Number of channels of the input image                                 |
| $H$   | Height of the input image   |
| $W$   | Width of the input image  |
| $f$   | Model   |
| $\partial f$  | Decision boundary of the base classifier $f$                          |
| $C$   | Number of classes   |
| cl  | Elected class function  |
| $\mathbf{x}$  | Input   |
| $\mathbf{y}$  | Input label   |
| Adversarial Examples Relative Notations                   |   |
| $\eta(m)$   | Accuracy of model $m$   |
| $n$   | Number of images considered in attacks                                |
| $\mathbf{x}_o$  | Original image targeted for adversarial, well-classified initially    |
| $\mathcal{X} = \{\mathbf{x}_o^1, \dots, \mathbf{x}_o^n\}$ | Set of images considered for the attack                               |
| $\mathbf{x}_{adv}$  | image adversarial   |
| $\mathbf{x}_{adv}^*$                                      | Optimal adversarial image   |
| $\delta$  | Perturbation  |
| $\delta_u$  | Universal perturbation  |
| $\delta_t$  | Transferable perturbation   |
| $\delta_s$  | Sensitive perturbation  |
| $K$   | Number of queries   |
| Measure Relative Notations                                |   |
| $\text{dist}(\mathbf{x}_{adv}, \mathbf{x}_o)$             | Distortion between $\mathbf{x}_{adv}$ and $\mathbf{x}_o$ in Eq. (3.2) |
| $\bar{\text{dist}}$                                       | Mean Distortion in Eq. (3.7)  |
| $\text{ASR}(\cdot)$                                       | Operating Characteristic in Eq. (3.9)                                 |
| $\text{ASR}$  | Attack Success Rate in Eq. (3.6)                                      |

Table 7.10 – Notations for the contribution on  $\text{R}_\text{OBIC}$  [144] in Appendix A.

|               |                              |
|---------------|------------------------------|
| $\mathcal{A}$ | Attack                       |
| $\Pi$         | Parameters of the attack     |
| $D_{1/2}$     | Half-distortion in Eq. (A.3) |

Table 7.11 – Notations for the contribution on SurFree [146] in Chapter 4.

|                       |   |
|-----------------------|---|
| $\mathcal{O}$         | Outside region  |
| $\partial\mathcal{O}$ | Boundary of the outside region  |
| $\mathbf{x}_b$        | Point on the boundary   |
| $\mathbf{n}$          | Normal at the boundary  |
| $\mathbf{u}$          | Direction given by the $\mathbf{x}_b$ and the original image $\mathbf{x}_o$ |
| $d$                   | Distance to the boundary  |
| $\tau$                | Angle Ratio for Sign Search   |
| $\mathcal{V}_k$       | List of former picked directions  |
| $L$                   | Size of the memory of directions  |
| $T$                   | Number of steps in sign search  |
| $\mathbf{z}$          | Point on the circle   |
| $\mathbf{t}$          | random direction orthogonal to $\mathbf{u}$                                 |
| $\mathcal{P}$         | Affine plan of dimension 2 in which we work                                 |

Table 7.12 – Notations for the contribution in Chapter 6.

|                     |  |
|---------------------|--|
| $\mathbf{N}$        | Noise from Normal Distribution                                   |
| $\sigma$            | Standard Deviation of Normal Distribution in Monte-Carlo Process |
| $n_{\text{mc}}$     | Samples Number in Monte-Carlo Process                            |
| $g_\sigma$          | deterministic classifier obtained by Randomized Smoothing        |
| $\partial g_\sigma$ | Decision Boundary of $g_\sigma$                                  |
| ResNet50            | ResNet50 Normally trained  |
| AdvTrain            | ResNet50 adversarially trained                                   |
| RandTrain           | ResNet50 randomly trained  |
| $R$                 | Certified Robustness   |

Table 7.13 – Notations for the contribution FiT in Chapter 5.

|   |   |
|---|---|
| $s$                                       | Source  |
| $t$                                       | Target  |
| $f_t$                                     | Target model  |
| $f_s$                                     | Source model  |
| $m$                                       | Number of source models owned by the attacker                         |
| $\mathcal{F}_s = \{f_s^1, \dots, f_s^m\}$ | Collection of source models   |
| $u_{x,s}$                                 | Adversarial direction obtained on the source for the image $x$        |
| $P_t^{\text{wb}}(D)$                      | Operating characteristic white box attack applied to the target model |
| $P_t^{\text{bb}}(D)$                      | Operating characteristic black box attack applied to the target model |
| $T_{s,t}$                                 | Transferability from the source to the target                         |
| ModSim                                    | Measure of the model similarity                                       |
| TransQ                                    | Measure of the quality of the transferable adversarial example        |

Table 7.14 – Notations for the contribution FBI [147] in Chapter 7 .

|  |  |
|--|--|
| $\text{det}$                                   | The task of detection  |
| $\text{I}$                                     | The task of identification   |
| $\text{m}$                                     | A vanilla model as listed in Table B.1                                   |
| $\text{V}(\text{m}, \theta)$                   | Variant obtained by applying procedure $\theta$ on $\text{m}$            |
| $\Theta$                                       | Set of all variation procedures and parameters                           |
| $\text{b}$                                     | The model in the black box   |
| $z$  | Top- $k$ output of the black box   |
| $\mathcal{C}$                                  | The set of classes labeled from 1 to $C$                                 |
| $\mathcal{Z}_k$                                | The set of top- $k$ outcomes   |
| $(C)_k$  | Falling factorial $(C)_k := C(C - 1) \dots (C - k + 1)$                  |
| $\mathcal{B}$                                  | Set of models Bob can create as defined in (7.2)                         |
| $\mathcal{A}$                                  | Set of models known by Alice   |
| $\mathcal{P}$                                  | Set of public vanilla models listed in Table B.1                         |
| $\mathcal{F}(\text{m})$                        | Family spanned by vanilla model $\text{m}$ (7.3)                         |
| $\mathcal{F}(\text{m}, \Psi)$                  | Family spanned by $\text{m}$ and variation $\Psi \subset \Theta$ (7.4)   |
| $N$  | Cardinality of $\mathcal{X}$   |
| $q_{1:\ell}$                                   | An ordered list of indices in $\llbracket N \rrbracket$                  |
| $\mathcal{D}(x)$                               | Set of outputs given by models in $\mathcal{D}$ for input $x$            |
| $\mathcal{M}(x, y, \mathcal{D})$               | Subset of models of $\mathcal{D}$ giving $y$ for input $x$               |
| $(\mathcal{A} \setminus \mathcal{F})^{(\ell)}$ | Subset of candidate models at step $\ell$ (7.6)                          |
| $s^{(\ell+1)}(x)$                              | Score of input $x$ at step $\ell + 1$ when $\mathcal{A} = \mathcal{B}$   |
| $L^{\text{pos}}, L^{\text{neg}}$               | Nb. of queries for a positive / negative detection                       |
| $\mathcal{S}_k$                                | Surjection from $\mathcal{Z}_k$ to $\mathcal{S}_k := \{0, 1, \dots, k\}$ |
| $z, y$   | Top- $k$ output of black box $\text{b}$ or of model $\text{m}$           |
| $Z, Y$   | Random top- $k$ output when the input is random                          |
| $\tilde{z}, \tilde{Z}, \tilde{y}, \tilde{Y}$   | Similar outputs after the surjection                                     |
| $W_\theta$                                     | $(k + 1) \times (k + 1)$ Transition matrix                               |
| $\hat{I}(\tilde{Z}, \tilde{Y})$                | Empirical mutual information in bits                                     |
| $\hat{H}_{\tilde{Z}}(\tilde{z})$               | Empirical entropy in bits  |
| $\text{dist}(\text{m}_1, \text{m}_2)$          | Distance between models with $L$ queries                                 |
| $\text{dist}(\text{b}, \mathcal{F})$           | Distance of the black box from family $\mathcal{F}$                      |

# BIBLIOGRAPHY

---

- [1] Emmanuel Abbe and Lihong Zheng, « Linear Universal Decoding for Compound Channels », *in: IEEE Transactions on Information Theory* (2010).
- [2] Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet, « Turning your weakness into a strength: Watermarking deep neural networks by backdoor-ing », *in: 27th USENIX Security Symposium*, 2018.
- [3] Ahmed Aldahdooh, Wassim Hamidouche, Sid Ahmed Fezza, and Olivier Déforges, « Adversarial example detection for DNN models: A review and experimental comparison », *in: Artificial Intelligence Review* (2022).
- [4] Motasem Alfarrar, Adel Bibi, Hasan Hammoud, Mohamed Gaafar, and Bernard Ghanem, « On the decision boundaries of neural networks: A tropical geometry perspective », *in: IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).
- [5] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, « Square attack: a query-efficient black-box adversarial attack via random search », *in: ECCV*, 2020.
- [6] Giovanni Apruzzese, Mauro Andreolini, Luca Ferretti, Mirco Marchetti, and Michele Colajanni, « Modeling Realistic Adversarial Attacks against Network Intrusion Detection Systems », *in: Digital Threats* (2022).
- [7] 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.
- [8] Randall Balestriero, Romain Cosentino, Behnaam Aazhang, and Richard Baraniuk, « The geometry of deep networks: Power diagram subdivision », *in: Advances in Neural Information Processing Systems* (2019).
- [9] Mauro Barni, Pedro Comesaña-Alfaro, Fernando Pérez-González, and Benedetta Tondi, « Are you threatening me?: Towards smart detectors in watermarking », *in: Media Watermarking, Security, and Forensics 2014*, SPIE, 2014.
- [10] Mauro Barni, Kassem Kallas, and Benedetta Tondi, « A new backdoor attack in cnns by training set corruption without label poisoning », *in: International Conference on Image Processing (ICIP)*, IEEE, 2019.



- 
- [11] Lejla Batina, Shivam Bhasin, Dirmanto Jap, and Stjepan Picek, « CSI neural network: Using side-channels to recover your artificial neural network information », *in: arXiv preprint arXiv:1810.09076* (2018).
- [12] Robbert van den Berg, Boris Skoric, and Vincent van der Leest, « Bias-Based Modeling and Entropy Analysis of PUFs », *in: Proc. of the 3rd Int. Workshop on Trustworthy Embedded Devices*, 2013.
- [13] B. Bonnet, T. Furon, and P. Bas, « What If Adversarial Samples Were Digital Images? », *in: IH&MMSec*, 2020.
- [14] Benoit Bonnet, « Understanding, Taming and Defending from Adversarial Examples », *in: PhD Thesis* (2023).
- [15] Benoit Bonnet, Teddy Furon, and Patrick Bas, « Generating Adversarial Images in Quantized Domains », *in: IEEE Transactions on Information Forensics and Security* (2022).
- [16] Benoit Bonnet, Teddy Furon, and Patrick Bas, « Generating adversarial images in quantized domains », *in: IEEE Transactions on Information Forensics and Security* (2021).
- [17] Akhilan Boopathy, Sijia Liu, Gaoyuan Zhang, Cynthia Liu, Pin-Yu Chen, Shiyu Chang, and Luca Daniel, « Proper network interpretability helps adversarial robustness in classification », *in: International Conference on Machine Learning*, PMLR, 2020.
- [18] Wieland Brendel, Jonas Rauber, and Matthias Bethge, « Decision-Based Adversarial Attacks: Reliable Attacks Against Black-Box Machine Learning Models », *in: International Conference on Learning Representations*, 2018.
- [19] Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong, « IPGuard: Protecting Intellectual Property of Deep Neural Networks via Fingerprinting the Classification Boundary », *in: Proc. of the 2021 ACM Asia Conference on Computer and Communications Security*, 2021.
- [20] Nicholas Carlini, Matthew Jagielski, and Ilya Mironov, « Cryptanalytic Extraction of Neural Network Models », *in: CRYPTO*, Springer-Verlag, 2020.
- [21] Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al., « Extracting training data from large language models », *in: 30th USENIX Security Symposium (USENIX Security 21)*, 2021.
- [22] Nicholas Carlini and David Wagner, « Towards evaluating the robustness of neural networks », *in: Symposium on Security and Privacy*, IEEE, 2017.

- 
- [23] Jianbo Chen, Michael I Jordan, and Martin J Wainwright, « HopSkipJumpAttack: A query-efficient decision-based attack », *in: Symposium on Security and Privacy*, IEEE, 2020.
- [24] Jinghui Chen and Quanquan Gu, « RayS: A Ray Searching Method for Hard-label Adversarial Attack », *in: ACM SIGKDD*, 2020.
- [25] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh, « ZOO: Zeroth Order Optimization Based Black-Box Attacks to Deep Neural Networks without Training Substitute Models », *in: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, Association for Computing Machinery, 2017.
- [26] Steven Chen, Nicholas Carlini, and David Wagner, « Stateful detection of black-box adversarial attacks », *in: Proceedings of the 1st ACM Workshop on Security and Privacy on Artificial Intelligence*, 2020.
- [27] Minhao Cheng, Thong Le, Pin-Yu Chen, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh, « Query-Efficient Hard-label Black-box Attack: An Optimization-based Approach », *in: ICLR*, 2019.
- [28] Minhao Cheng, Simranjit Singh, Patrick H. Chen, Pin-Yu Chen, Sijia Liu, and Cho-Jui Hsieh, « Sign-OPT: A Query-Efficient Hard-label Adversarial Attack », *in: International Conference on Learning Representations*, 2020.
- [29] Christopher A. Choquette-Choo, Florian Tramer, Nicholas Carlini, and Nicolas Papernot, « Label-Only Membership Inference Attacks », *in: Proceedings of the 38th International Conference on Machine Learning*, Proceedings of Machine Learning Research, PMLR, 2021.
- [30] Michael Chui, E Hazan, R Roberts, A Singla, K Smaje, A Sukharevsky, L Yee, and R Zemel, *The economic potential of generative AI: The next productivity frontier*, 2023.
- [31] Antonio Emanuele Cinà, Ambra Demontis, Battista Biggio, Fabio Roli, and Marcello Pelillo, *Energy-Latency Attacks via Sponge Poisoning*, 2023, arXiv: 2203.08147.
- [32] Antonio Emanuele Cinà, Alessandro Torcinovich, and Marcello Pelillo, « A black-box adversarial attack for poisoning clustering », *in: Pattern Recognition (2022)*.
- [33] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter, « Certified Adversarial Robustness via Randomized Smoothing », *in: ICML*, 2019.
- [34] Pedro Comesana, Luis Pérez-Freire, and Fernando Pérez-González, « Blind newton sensitivity attack », *in: IEE Proceedings-Information Security (2006)*.
- [35] Jacson Rodrigues Correia-Silva, Rodrigo F Berriel, Claudine Badue, Alberto F de Souza, and Thiago Oliveira-Santos, « Copycat cnn: Stealing knowledge by persuading confes-

- 
- sion with random non-labeled data », *in: 2018 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2018.
- [36] I. J. Cox and J. -. M. G. Linnartz, « Public watermarks and resistance to tampering », *in: Proceedings of International Conference on Image Processing*, 1997.
- [37] F Crecchi, D Bacciu, B Biggio, et al., « Detecting adversarial examples through nonlinear dimensionality reduction », *in: ESANN 2019-Proceedings, 27th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2019.
- [38] F. Croce, M. Andriushchenko, V. Sehwag, N. Flammarion, M. Chiang, P. Mittal, and M. Hein, « RobustBench: a standardized adversarial robustness benchmark », *in: arXiv preprint arXiv:2010.09670* (2020).
- [39] G. Cybenko, « Approximation by superpositions of a sigmoidal function », *in: Mathematics of Control, Signals and Systems* (1989).
- [40] D. Han, S. Yun, B. Heo, and Y. Yoo, « ReXNet: Diminishing Representational Bottleneck on CNN », *in: arXiv e-prints arXiv:2007.00992* (2020).
- [41] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Siwei Li, Li Chen, Michael E. Kounavis, and Duen Horng Chau, « SHIELD: Fast, Practical Defense and Vaccination for Deep Learning Using JPEG Compression », *in: Proc. of the 24th ACM SIGKDD International Conference on Knowledge Discovery; Data Mining, KDD '18*, 2018.
- [42] John Daugman, « Information Theory and the IrisCode », *in: IEEE Trans. on Information Forensics and Security* (2016).
- [43] Marco De Angelis and Ander Gray, *Why the 1-Wasserstein distance is the area between the two marginal CDFs*, 2021.
- [44] Jeff Dean, David Patterson, and Cliff Young, « A New Golden Age in Computer Architecture: Empowering the Machine-Learning Revolution », *in: IEEE Micro* (2018).
- [45] Edoardo Debenedetti, Nicholas Carlini, and Florian Tramèr, « Evading Black-box Classifiers Without Breaking Eggs », *in: arXiv preprint arXiv:2306.02895* (2023).
- [46] Jia Deng, R. Socher, Li Fei-Fei, Wei Dong, Kai Li, and Li-Jia Li, « ImageNet: A large-scale hierarchical image database », *in: IEEE CVPR*, 2009.
- [47] Guneet S. Dhillon, Kamyar Azizzadenesheli, Jeremy D. Bernstein, Jean Kossaiji, Aran Khanna, Zachary C. Lipton, and Animashree Anandkumar, « Stochastic activation pruning for robust adversarial defense », *in: International Conference on Learning Representations*, 2018.

- 
- [48] Antreas Dionysiou, Vassilis Vassiliades, and Elias Athanasopoulos, « Exploring Model Inversion Attacks in the Black-box Setting », in: *Proceedings on Privacy Enhancing Technologies* (2023).
- [49] Hadi M Dolatabadi, Sarah Erfani, and Christopher Leckie, «  $\ell_\infty$ -robustness and beyond: Unleashing efficient adversarial training », in: *European Conference on Computer Vision*, Springer, 2022.
- [50] Junhao Dong, Junxi Chen, Xiaohua Xie, Jianhuang Lai, and Hao Chen, *Adversarial Attack and Defense for Medical Image Analysis: Methods and Applications*, 2023.
- [51] Y. Dong, Q.-A. Fu, X. Yang, T. Pang, H. Su, Z. Xiao, and J. Zhu, « Benchmarking Adversarial Robustness on Image Classification », in: *CVPR*, 2020.
- [52] Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu, « Evading defenses to transferable adversarial examples by translation-invariant attacks », in: *CVPR*, 2019.
- [53] Vasisht Duddu, Debasis Samanta, D Vijay Rao, and Valentina E Balas, « Stealing neural networks via timing side channels », in: *arXiv preprint arXiv:1812.11720* (2018).
- [54] Gintare Karolina Dziugaite, Zoubin Ghahramani, and Daniel M. Roy, « A study of the effect of JPG compression on adversarial images », in: *ArXiv abs/1608.00853* (2016).
- [55] John W. Earl, « Tangential sensitivity analysis of watermarks using prior information », in: *Security, Steganography, and Watermarking of Multimedia Contents IX*, International Society for Optics and Photonics, SPIE.
- [56] Ivan Evtimov, Pascal Sturmfels, and Tadayoshi Kohno, « Foggysight: A scheme for facial lookup privacy », in: (2021).
- [57] Kevin Eykholt, Ivan Evtimov, Earlece Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song, « Robust physical-world attacks on deep learning visual classification », in: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018.
- [58] Alhussein Fawzi, Omar Fawzi, and Pascal Frossard, « Analysis of classifiers' robustness to adversarial perturbations », in: *Machine learning* (2018).
- [59] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard, « Robustness of classifiers: from adversarial to random noise », in: *Advances in Neural Information Processing Systems*, ed. by D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Curran Associates, Inc., 2016.
- [60] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, Pascal Frossard, and Stefano Soatto, « Empirical Study of the Topology and Geometry of Deep Networks », in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

- 
- [61] Ryan Feng, Ashish Hooda, Neal Mangaokar, Kassem Fawaz, Somesh Jha, and Atul Prakash, « Investigating Stateful Defenses Against Black-Box Adversarial Examples », *in: arXiv preprint arXiv:2303.06280* (2023).
- [62] Gil Fidel, Ron Bitton, and Asaf Shabtai, « When explainability meets adversarial learning: Detecting adversarial examples using shap signatures », *in: IJCNN, IEEE*, 2020.
- [63] 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.
- [64] Kunihiro Fukushima, « Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position », *in: Biological Cybernetics* (1980).
- [65] Teddy Furon and Luis Pérez-Freire, « EM decoding of Tardos traitor tracing codes », *in: Proceedings of the 11th ACM workshop on Multimedia and security*, 2009.
- [66] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky, « Domain-adversarial training of neural networks », *in: The journal of machine learning research* 17.1 (2016), pp. 2096–2030.
- [67] Yue Gao, Ilya Shumailov, Kassem Fawaz, and Nicolas Papernot, « On the limitations of stochastic pre-processing defenses », *in: Advances in Neural Information Processing Systems* (2022).
- [68] Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S. Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian J. Goodfellow, « The Relationship Between High-Dimensional Geometry and Adversarial Examples », *in: 2018*.
- [69] Xavier Glorot and Yoshua Bengio, « Understanding the difficulty of training deep feed-forward neural networks », *in: Proceedings of the thirteenth international conference on artificial intelligence and statistics, JMLR Workshop and Conference Proceedings*, 2010.
- [70] Micah Goldblum, Dimitris Tsipras, Chulin Xie, Xinyun Chen, Avi Schwarzschild, Dawn Song, Aleksander Mądry, Bo Li, and Tom Goldstein, « Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses », *in: Transactions on Pattern Analysis and Machine Intelligence* (2022).
- [71] Xueluan Gong, Qian Wang, Yanjiao Chen, Wang Yang, and Xinchang Jiang, « Model extraction attacks and defenses on cloud-based machine learning models », *in: Communications Magazine* (2020).

- 
- [72] I. J. Goodfellow, J. Shlens, and C. Szegedy, « Explaining and Harnessing Adversarial Examples », *in: ICLR*, 2015.
- [73] Shuyue Guan and Murray Loew, « Analysis of generalizability of deep neural networks based on the complexity of decision boundary », *in: 2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, IEEE, 2020.
- [74] Linda Guiga and AW Roscoe, « Neural Network Security: Hiding CNN Parameters with Guided Grad-CAM. », *in: ICISSP*, 2020.
- [75] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens van der Maaten, « Countering Adversarial Images using Input Transformations », *in: International Conference on Learning Representations*, 2018.
- [76] Kavya Gupta, Jean-Christophe Pesquet, Beatrice Pesquet-Popescu, Fateh Kaakai, and Fragkiskos D Malliaros, « An Adversarial Attacker for Neural Networks in Regression Problems. », *in: AISafety IJCAI*, 2021.
- [77] Sicong Han, Chenhao Lin, Chao Shen, Qian Wang, and Xiaohong Guan, « Interpreting Adversarial Examples in Deep Learning: A Review », *in: ACM Comput. Surv.* (2023).
- [78] Song Han, Huizi Mao, and William J Dally, « Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding », *in: arXiv preprint arXiv:1510.00149* (2015).
- [79] Jamie Hayes, « Extensions and Limitations of Randomized Smoothing for Robustness Guarantees », *in: CVPR*, 2020.
- [80] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, « Deep Residual Learning for Image Recognition », *in: CVPR*, 2016.
- [81] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, « Deep Residual Learning for Image Recognition », *in: CVPR*, 2016.
- [82] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, « Delving deep into rectifiers: Surpassing human-level performance on imagenet classification », *in: Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.
- [83] Warren He, Bo Li, and Dawn Song, « Decision boundary analysis of adversarial examples », *in: International Conference on Learning Representations*, 2018.
- [84] Yang He, Shadi Rahimian, Bernt Schiele, and Mario Fritz, « Segmentations-leak: Membership inference attacks and defenses in semantic image segmentation », *in: Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16*, Springer, 2020.

- 
- [85] Zecheng He, Tianwei Zhang, and Ruby Lee, « Sensitive-Sample Fingerprinting of Deep Neural Networks », in: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [86] Byeongho Heo, Minsik Lee, Sangdoon Yun, and Jin Young Choi, « Knowledge distillation with adversarial samples supporting decision boundary », in: *Proceedings of the AAAI conference on artificial intelligence*, 2019.
- [87] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, « Distilling the knowledge in a neural network », in: *arXiv preprint arXiv:1503.02531* (2015).
- [88] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh, « A fast learning algorithm for deep belief nets », in: *Neural computation* (2006).
- [89] Hokuto Hirano, Akinori Minagi, and Kazuhiro Takemoto, « Universal adversarial attacks on deep neural networks for medical image classification », in: *BMC Medical Imaging* (2021).
- [90] Sepp Hochreiter and Jürgen Schmidhuber, « Long Short-Term Memory », in: *Neural Computation* (1997).
- [91] Kurt Hornik, Maxwell Stinchcombe, and Halbert White, « Multilayer feedforward networks are universal approximators », in: *Neural Networks* (1989).
- [92] Jie Hu, Li Shen, and Gang Sun, « Squeeze-and-Excitation Networks », in: *CVPR*, 2018.
- [93] Qian Huang, Isay Katsman, Horace He, Zeqi Gu, Serge Belongie, and Ser-Nam Lim, « Enhancing adversarial example transferability with an intermediate level attack », in: *ICCV*, 2019.
- [94] Tianjin Huang, Vlado Menkovski, Yulong Pei, Yuhao Wang, and Mykola Pechenizkiy, « Direction-aggregated attack for transferable adversarial examples », in: *JETC* (2022).
- [95] Yen-Wei Huang and Pierre Moulin, « On the saddle-point solution and the large-coalition asymptotics of fingerprinting games », in: *IEEE Transactions on Information Forensics and Security* (2011).
- [96] Yi Huang and Adams Wai-Kin Kong, « Transferable Adversarial Attack based on Integrated Gradients », in: *ICLR*, 2022.
- [97] Ahmed Imtiaz Humayun, Randall Balestriero, and Richard Baraniuk, « Exact visualization of deep neural network geometry and decision boundary », in: *NeurIPS 2022 Workshop on Symmetry and Geometry in Neural Representations*, 2022.
- [98] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin, « Black-box Adversarial Attacks with Limited Queries and Information », in: *Proceedings of Machine Learning Research*, PMLR, 2018.

- 
- [99] Andrew Ilyas, Logan Engstrom, and Aleksander Madry, « Prior Convictions: Black-box Adversarial Attacks with Bandits and Priors », in: *International Conference on Learning Representations*, 2019.
- [100] A.G. Ivakhnenko, V.G. Lapa, V.G. Lapa, and R.N. McDonough, *Cybernetics and Forecasting Techniques*, Modern analytic and computational methods in science and mathematics, American Elsevier Publishing Company, 1967.
- [101] Andrii Ivaniuk and Galyna Kriukova, « On Geometric Properties of Adversarial Examples », in: *2021 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, IEEE, 2021.
- [102] Matthew Jagielski, Nicholas Carlini, David Berthelot, Alex Kurakin, and Nicolas Papernot, « High accuracy and high fidelity extraction of neural networks », in: *29th USENIX security symposium (USENIX Security 20)*, 2020.
- [103] Guillaume Jeanneret, Loic Simon, and Frederic Jurie, « Adversarial Counterfactual Visual Explanations », in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.
- [104] Susmit Jha, Uyeong Jang, Somesh Jha, and Brian Jalaian, « Detecting adversarial examples using data manifolds », in: *MILCOM 2018-2018 IEEE Military Communications Conference (MILCOM)*, IEEE, 2018.
- [105] Jinyuan Jia, Xiaoyu Cao, Binghui Wang, and Neil Zhenqiang Gong, « Certified Robustness for Top-k Predictions against Adversarial Perturbations via Randomized Smoothing », in: *ICLR (2020)*.
- [106] Mika Juuti, Sebastian Szyller, Samuel Marchal, and N Asokan, « PRADA: protecting against DNN model stealing attacks », in: *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, IEEE, 2019, pp. 512–527.
- [107] Mostafa Kahla, Si Chen, Hoang Anh Just, and Ruoxi Jia, « Label-only model inversion attacks via boundary repulsion », in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.
- [108] Hamid Karimi and Jiliang Tang, « Decision Boundary of Deep Neural Networks: Challenges and Opportunities », in: *Proceedings of the 13th International Conference on Web Search and Data Mining*, Association for Computing Machinery, 2020.
- [109] Sanjay Kariyappa, Atul Prakash, and Moinuddin K Qureshi, « Protecting dnns from theft using an ensemble of diverse models », in: *International Conference on Learning Representations*, 2020.



- 
- [110] Sanjay Kariyappa and Moinuddin K Qureshi, « Defending against model stealing attacks with adaptive misinformation », in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.
- [111] Anouar Kherchouche, Sid Ahmed Fezza, Wassim Hamidouche, and Olivier Déforges, « Detection of Adversarial Examples in Deep Neural Networks with Natural Scene Statistics », in: *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020.
- [112] Byungju Kim and Junmo Kim, « Adjusting Decision Boundary for Class Imbalanced Learning », in: *IEEE Access* (2020).
- [113] Joel Klinger, Juan Mateos-Garcia, and Konstantinos Stathoulopoulos, « A narrowing of AI research? », in: *arXiv preprint arXiv:2009.10385* (2020).
- [114] A. Krizhevsky, « One weird trick for parallelizing CNN », in: *CoRR* (2014), URL: <http://arxiv.org/abs/1404.5997>.
- [115] A. Krizhevsky and G. Hinton, « Learning multiple layers of features from tiny images », in: *Master's thesis, Department of Computer Science, University of Toronto* (2009).
- [116] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, « ImageNet Classification with Deep Convolutional Neural Networks », in: *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2012.
- [117] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio, « Adversarial examples in the physical world. », in: (2016).
- [118] Bethge Lab, « Robust Vision Benchmark », URL: <https://robust.vision/> (visited on ).
- [119] Erwan Le Merrer and Trédan Gilles, « Tampernn: efficient tampering detection of deployed neural nets », in: *2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE)*, IEEE, 2019.
- [120] Erwan Le Merrer, Patrick Perez, and Gilles Trédan, « Adversarial frontier stitching for remote neural network watermarking », in: *Neural Computing and Applications* (2020).
- [121] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner, « Gradient-based learning applied to document recognition », in: *Proceedings of the IEEE* (1998).
- [122] Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana, « Certified Robustness to Adversarial Examples with Differential Privacy », in: *IEEE S&P*, 2019.
- [123] Saehyung Lee, Hyungyu Lee, and Sungroh Yoon, « Adversarial vertex mixup: Toward better adversarially robust generalization », in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 272–281.

- 
- [124] Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin, « Certified Adversarial Robustness with Additive Noise », *in: NIPS*, 2019.
- [125] Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin, « Second-Order Adversarial Attack and Certifiable Robustness », *in: (2019)*.
- [126] H. Li, X. Xu, X. Zhang, S. Yang, and B. Li, « QEBA: Query-Efficient Boundary-Based Blackbox Attack », *in: CVPR*, Los Alamitos, CA, USA: IEEE, 2020.
- [127] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf, « Pruning Filters for Efficient ConvNets », *in: Int. Conf. on Learning Representations*, 2016.
- [128] Huiying Li, Shawn Shan, Emily Wenger, Jiayun Zhang, Haitao Zheng, and Ben Y Zhao, « Blacklight: Scalable defense for neural networks against {Query-Based}{Black-Box} attacks », *in: 31st USENIX Security Symposium (USENIX Security 22)*, 2022.
- [129] Jie Li, Rongrong Ji, Peixian Chen, Baochang Zhang, Xiaopeng Hong, Ruixin Zhang, Shaoxin Li, Jilin Li, Feiyue Huang, and Yongjian Wu, « Aha! adaptive history-driven attack for decision-based black-box models », *in: Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.
- [130] Jie Li, Rongrong Ji, Hong Liu, Xiaopeng Hong, Yue Gao, and Qi Tian, « Universal perturbation attack against image retrieval », *in: Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.
- [131] Shasha Li, Shitong Zhu, Sudipta Paul, Amit Roy-Chowdhury, Chengyu Song, Srikanth Krishnamurthy, Ananthram Swami, and Kevin S Chan, « Connecting the dots: Detecting adversarial perturbations using context inconsistency », *in: Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16*, Springer, 2020.
- [132] Y. Chen and J. Li, H. Xiao, X. Jin, S. Yan, and J. Feng, « Dual Path Networks », *in: NIPS*, 2017.
- [133] Yingwei Li, Song Bai, Yuyin Zhou, Cihang Xie, Zhishuai Zhang, and Alan Yuille, « Learning transferable adversarial examples via ghost networks », *in: Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.
- [134] Yuanchun Li, Ziqi Zhang, Bingyan Liu, Ziyue Yang, and Yunxin Liu, « ModelDiff: Testing-Based DNN Similarity Comparison for Model Reuse Detection », *in: Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis*, Association for Computing Machinery, 2021.
- [135] Yue Li, Hongxia Wang, and Mauro Barni, « A survey of Deep Neural Network watermarking techniques », *in: (2021)*.

- 
- [136] Jiadong Lin, Chuanbiao Song, Kun He, Liwei Wang, and John E Hopcroft, « Nesterov Accelerated Gradient and Scale Invariance for Adversarial Attacks », *in: ICLR*, 2020.
- [137] J. Linnartz and Marten van Dijk, « Analysis of the Sensitivity Attack against Electronic Watermarks in Images », *in: Information Hiding*, 1998.
- [138] Jean -Paul MG Linnartz and Marten Van Dijk, « Analysis of the sensitivity attack against electronic watermarks in images », *in: Information Hiding: Second International Workshop*, Springer, 1998.
- [139] S. Liu, P. -Y. Chen, B. Kailkhura, G. Zhang, A. O. Hero III, and P. K. Varshney, « A Primer on Zeroth-Order Optimization in Signal Processing and Machine Learning: Principals, Recent Advances, and Applications », *in: IEEE Signal Processing Magazine* ().
- [140] Nils Lukas, Yuxuan Zhang, and Florian Kerschbaum, « Deep Neural Network Fingerprinting by Conferrable Adversarial Examples », *in: International Conference on Learning Representations*, 2021.
- [141] Nils Lukas, Yuxuan Zhang, and Florian Kerschbaum, « Deep Neural Network Fingerprinting by Conferrable Adversarial Examples », *in: ICLR*, 2021.
- [142] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu, « Towards Deep Learning Models Resistant to Adversarial Attacks », *in: ICLR*, 2017.
- [143] 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*, 2018.
- [144] Thibault Maho, Benoît Bonnet, Teddy Furon, and Erwan Le Merrer, *RoBIC: A benchmark suite for assessing classifiers robustness*, 2021.
- [145] Thibault Maho, Teddy Furon, and Erwan Le Merrer, « Randomized Smoothing under Attack: How Good is it in Praticice? », *in: ICASSP*, 2022.
- [146] Thibault Maho, Teddy Furon, and Erwan Le Merrer, « Surferee: a fast surrogate-free black-box attack », *in: CVPR*, 2021.
- [147] Thibault Maho, Teddy Furon, and Erwan Le Merrer, « FBI: Fingerprinting models with Benign Inputs », *in: (2023)*.
- [148] Gallil Maimon and Lior Rokach, « A universal adversarial policy for text classifiers », *in: Neural Networks* (2022).

- 
- [149] Horia Mania, John Miller, Ludwig Schmidt, Moritz Hardt, and Benjamin Recht, « Model Similarity Mitigates Test Set Overuse », *in: Advances in Neural Information Processing Systems*, 2019.
- [150] Sébastien Marcel and Yann Rodriguez, « Torchvision the Machine-Vision Package of Torch », *in: Proceedings of the 18th ACM International Conference on Multimedia*, 2010.
- [151] Warren Mcculloch and Walter Pitts, « A Logical Calculus of Ideas Immanent in Nervous Activity », *in: Bulletin of Mathematical Biophysics* (1943).
- [152] Dongyu Meng and Hao Chen, « Magnet: a two-pronged defense against adversarial examples », *in: Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, 2017, pp. 135–147.
- [153] Smitha Milli, Ludwig Schmidt, Anca D. Dragan, and Moritz Hardt, « Model Reconstruction from Model Explanations », *in: Proceedings of the Conference on Fairness, Accountability, and Transparency*, Association for Computing Machinery, 2019.
- [154] Guido F Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio, « On the number of linear regions of deep neural networks », *in: Advances in neural information processing systems* (2014).
- [155] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard, « Universal Adversarial Perturbations », *in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [156] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard, « Deepfool: a simple and accurate method to fool deep neural networks », *in: CVPR*, 2016.
- [157] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard, « Robustness via curvature regularization, and vice versa », *in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.
- [158] Seyed-Mohsen Moosavi-Dezfooli, Ashish Shrivastava, and Oncel Tuzel, *Divide, Denoise, and Defend against Adversarial Attacks*, 2019, arXiv: 1802.06806 [cs.CV].
- [159] John Morris, Eli Liland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi, « TextAttack: A Framework for Adversarial Attacks, Data Augmentation, and Adversarial Training in NLP », *in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020, pp. 119–126.
- [160] Paarth Neekhara, Shehzeen Hussain, Prakhar Pandey, Shlomo Dubnov, Julian McAuley, and Farinaz Koushanfar, « Universal adversarial perturbations for speech recognition

- 
- systems », in: *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2019.
- [161] Yu. Nesterov, « Efficiency of Coordinate Descent Methods on Huge-Scale Optimization Problems », in: *SIAM Journal on Optimization* (2012), DOI: 10.1137/100802001.
- [162] Ngoc-Bao Nguyen, Keshigeyan Chandrasegaran, Milad Abdollahzadeh, and Ngai-Man Cheung, « Re-thinking Model Inversion Attacks Against Deep Neural Networks », in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.
- [163] Curtis G Northcutt, Anish Athalye, and Jonas Mueller, « Pervasive label errors in test sets destabilize machine learning benchmarks », in: *arXiv preprint arXiv:2103.14749* (2021).
- [164] Seong Joon Oh, Max Augustin, Bernt Schiele, and Mario Fritz, « Towards Reverse-Engineering Black-Box Neural Networks », in: *Sixth International Conference on Learning Representations*, 2018.
- [165] R. Olivier, B. Raj, and M. Shah, « High-Frequency Adversarial Defense for Speech and Audio », in: *ICASSP*, 2021.
- [166] Daryna Oliynyk, Rudolf Mayer, and Andreas Rauber, « I know what you trained last summer: A survey on stealing machine learning models and defences », in: *ACM Computing Surveys* (2023).
- [167] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz, « Knockoff nets: Stealing functionality of black-box models », in: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019.
- [168] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz, « Prediction Poisoning: Towards Defenses Against DNN Model Stealing Attacks », in: *International Conference on Learning Representations*, 2020.
- [169] Guillermo Ortiz-Jimenez, Apostolos Modas, Seyed-Mohsen Moosavi, and Pascal Frossard, « Hold me tight! influence of discriminative features on deep network boundaries », in: *Advances in Neural Information Processing Systems* (2020).
- [170] Guillermo Ortiz-Jiménez, Apostolos Modas, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard, « Optimism in the face of adversity: Understanding and improving deep learning through adversarial robustness », in: *Proceedings of the IEEE* (2021).
- [171] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al., « Training language

- 
- models to follow instructions with human feedback », *in: Advances in Neural Information Processing Systems* (2022).
- [172] Xudong Pan, Mi Zhang, Yifan Lu, and Min Yang, « TAFE: A Task-Agnostic Fingerprinting Algorithm for Neural Networks », *in: European Symposium on Research in Computer Security*, Springer, 2021.
- [173] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow, « Transferability in machine learning: from phenomena to black-box attacks using adversarial samples », *in: arXiv preprint arXiv:1605.07277* (2016).
- [174] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami, « Practical Black-Box Attacks against Machine Learning », *in: ASIA CCS*, 2017.
- [175] Razvan Pascanu, Guido Montufar, and Yoshua Bengio, « On the number of response regions of deep feed forward networks with piece-wise linear activations », *in: arXiv preprint arXiv:1312.6098* (2013).
- [176] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala, « PyTorch: An Imperative Style, High-Performance Deep Learning Library », *in: Advances in Neural Information Processing Systems 32*, Curran Associates, Inc., 2019.
- [177] Anibal Pedraza, Oscar Deniz, and Gloria Bueno, « Approaching Adversarial Example Classification with Chaos Theory », *in:* (2020).
- [178] Zirui Peng, Shaofeng Li, Guoxing Chen, Cheng Zhang, Haojin Zhu, and Minhui Xue, « Fingerprinting Deep Neural Networks Globally via Universal Adversarial Perturbations », *in: arXiv preprint arXiv:2202.08602* (2022).
- [179] Luis Perez-Freire, Pedro Comesana, Juan Ramon Troncoso-Pastoriza, and Fernando Perez-Gonzalez, « Watermarking security: a survey », *in: Transactions on data hiding and multimedia security I*, Springer, 2006.
- [180] Deyan Petrov and Timothy M Hospedales, « Measuring the transferability of adversarial examples », *in: arXiv preprint arXiv:1907.06291* (2019).
- [181] Maura Pintor, Fabio Roli, Wieland Brendel, and Battista Biggio, « Fast Minimum-norm Adversarial Attacks through Adaptive Norm Constraints », *in: NeurIPS*, 2021.

- 
- [182] Rabiah Al-qudah, Moayad Aloqaily, Bassem Ouni, Mohsen Guizani, and Thierry Lestable, *An Incremental Gray-box Physical Adversarial Attack on Neural Network Training*, 2023.
- [183] Erwin Quiring, Daniel Arp, and Konrad Rieck, « Forgotten Siblings: Unifying Attacks on Machine Learning and Digital Watermarking », *in: IEEE European Symposium on Security and Privacy*, 2018.
- [184] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár, « Designing Network Design Spaces », *in: CVPR*, 2020.
- [185] Edward Raff, Jared Sylvester, Steven Forsyth, and Mark McLean, « Barrage of Random Transforms for Adversarially Robust Defense », *in: Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [186] Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John Duchi, and Percy Liang, « Adversarial Training Can Hurt Generalization », *in: ICML 2019 Workshop on Identifying and Understanding Deep Learning Phenomena*, 2019.
- [187] Ali Rahmati, Seyed-Mohsen Moosavi-Dezfooli, Pascal Frossard, and Huaiyu Dai, « GeoDA: a geometric framework for black-box adversarial attacks », *in: CVPR*, 2020.
- [188] C. Rajsiki, « A metric space of discrete probability distributions », *in: Information and Control* (1961).
- [189] Karthikeyan Natesan Ramamurthy, Kush Varshney, and Krishnan Mody, « Topological data analysis of decision boundaries with application to model selection », *in: International Conference on Machine Learning*, PMLR, 2019.
- [190] Jonas Rauber, Wieland Brendel, and Matthias Bethge, « Foolbox: A Python toolbox to benchmark the robustness of machine learning models », *in: Reliable Machine Learning in the Wild Workshop, 34th International Conference on Machine Learning*, 2017.
- [191] Jonas Rauber, Roland Zimmermann, Matthias Bethge, and Wieland Brendel, « Foolbox Native: Fast adversarial attacks to benchmark the robustness of machine learning models in PyTorch, TensorFlow, and JAX », *in: Journal of Open Source Software* (2020).
- [192] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer, « High-resolution image synthesis with latent diffusion models. 2022 IEEE », *in: CVF CVPR*, 2021.
- [193] Jerome Rony, Luiz G. Hafemann, Luiz S. Oliveira, Ismail Ben Ayed, Robert Sabourin, and Eric Granger, « Decoupling Direction and Norm for Efficient Gradient-Based L2 Adversarial Attacks and Defenses », *in: The IEEE CVPR*.

- 
- [194] Frank Rosenblatt, « The perceptron: a probabilistic model for information storage and organization in the brain. », *in: Psychological review* (1958).
- [195] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams, « Learning representations by back-propagating errors », *in: nature* (1986).
- [196] Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, and Hervé Jégou, « Radioactive data: tracing through training », *in: International Conference on Machine Learning*, PMLR, 2020.
- [197] Hadi Salman, Mingjie Sun, Greg Yang, Ashish Kapoor, and J. Zico Kolter, « Denoised Smoothing: A Provable Defense for Pretrained Classifiers », *in: NIPS*, 2020.
- [198] Pouya Samangouei, Maya Kabkab, and Rama Chellappa, « Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models », *in: International Conference on Learning Representations*, 2018, URL: <https://openreview.net/forum?id=BkJ3ibb0->.
- [199] Pouya Samangouei, Maya Kabkab, and Rama Chellappa, « Defense-gan: Protecting classifiers against adversarial attacks using generative models », *in: arXiv preprint arXiv:1805.06605* (2018).
- [200] S. Samonas and D. Coss, « The CIA strikes back: Redefining confidentiality, integrity and availability in security », *in: Journal of Information System Security* (2014).
- [201] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen, « MobileNetV2: Inverted Residuals and Linear Bottlenecks », *in: CVPR*, 2018.
- [202] Sunandini Sanyal, Sravanti Addepalli, and R Venkatesh Babu, « Towards data-free model stealing in a hard label setting », *in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.
- [203] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry, « Adversarially robust generalization requires more data », *in: Advances in neural information processing systems* (2018).
- [204] Ali Shafahi, W. Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein, « Are adversarial examples inevitable? », *in: International Conference on Learning Representations*, 2019.
- [205] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein, « Adversarial training for free! », *in: Advances in Neural Information Processing Systems* (2019).



- 
- [206] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrappalli, « The pitfalls of simplicity bias in neural networks », *in: Advances in Neural Information Processing Systems* (2020).
- [207] Uri Shaham, James Garritano, Yutaro Yamada, Ethan Weinberger, Alex Cloninger, Xiuyuan Cheng, Kelly P. Stanton, and Yuval Kluger, « Defending against Adversarial Images using Basis Functions Transformations », *in: (2018)*, arXiv: 1803.10840.
- [208] Yash Sharma, Gavin Weiguang Ding, and Marcus Brubaker, « On the effectiveness of low frequency perturbations », *in: arXiv preprint arXiv:1903.00073* (2019).
- [209] Lujia Shen, Xuhong Zhang, Shouling Ji, Yuwen Pu, Chunpeng Ge, Xing Yang, and Yanghe Feng, « TextDefense: Adversarial Text Detection based on Word Importance Entropy », *in: (2023)*.
- [210] Meng Shen, Changyue Li, Hao Yu, Qi Li, Liehuang Zhu, and Ke Xu, « Decision-based Query Efficient Adversarial Attack via Adaptive Boundary Learning », *in: Transactions on Dependable and Secure Computing* (2023).
- [211] Yucheng Shi, Yahong Han, and Qi Tian, « Polishing decision-based adversarial noise with a customized sampling », *in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.
- [212] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov, « Membership inference attacks against machine learning models », *in: Symposium on security and privacy (SP)*, IEEE, 2017.
- [213] Ilia Shumailov, Yiren Zhao, Daniel Bates, Nicolas Papernot, Robert Mullins, and Ross Anderson, « Sponge Examples: Energy-Latency Attacks on Neural Networks », *in: European Symposium on Security and Privacy*, IEEE, 2021.
- [214] Karen Simonyan and Andrew Zisserman, « Very Deep Convolutional Networks for Large-Scale Image Recognition », *in: CoRR* (2014).
- [215] Gowthami Somepalli, Liam Fowl, Arpit Bansal, Ping Yeh-Chiang, Yehuda Dar, Richard Baraniuk, Micah Goldblum, and Tom Goldstein, « Can neural nets learn the same model twice? investigating reproducibility and double descent from the decision boundary perspective », *in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.
- [216] Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman, « Pixeldefend: Leveraging generative models to understand and defend against adversarial examples », *in: arXiv preprint arXiv:1710.10766* (2017).

- 
- [217] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, « Dropout: A Simple Way to Prevent Neural Networks from Overfitting », in: *Journal of Machine Learning Research* (2014).
- [218] Octavian Suci, Radu Marginean, Yigitcan Kaya, Hal Daume III, and Tudor Dumitras, « When does machine learning {FAIL}? generalized transferability for evasion and poisoning attacks », in: *27th USENIX Security Symposium (USENIX Security 18)*, 2018, pp. 1299–1316.
- [219] Xuxiang Sun, Gong Cheng, Hongda Li, Lei Pei, and Junwei Han, « Exploring effective data for surrogate training towards black-box attack », in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 15355–15364.
- [220] M.D. Swanson, Bin Zhu, and A.H. Tewfik, « Transparent robust image watermarking », in: *Proceedings of 3rd IEEE International Conference on Image Processing*, 1996.
- [221] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus, « Intriguing properties of neural networks », in: *ICLR*, ed. by Yoshua Bengio and Yann LeCun, 2014.
- [222] Naoya Takahashi, Shota Inoue, and Yuki Mitsufuji, « Adversarial Attacks on Audio Source Separation », in: *ICASSP*, 2021.
- [223] Ran Tamir and Neri Merhav, *The MMI Decoder is Asymptotically Optimal for the Typical Random Code and for the Expurgated Code*, 2020.
- [224] M. Tan and Q. Le, « EfficientNet: Rethinking Model Scaling for CNN », in: *ICML*, 2019.
- [225] Mingxing Tan and Quoc V. Le, « MixConv: Mixed Depthwise Convolutional Kernels », in: *arXiv e-prints arxiv:1907.09595* (), eprint: 1907.09595 (cs.CV).
- [226] K. Tanaka, Y. Nakamura, and K. Matsui, « Embedding secret information into a dithered multi-level image », in: *IEEE Conference on Military Communications*, 1990.
- [227] Thomas Tanay and Lewis Griffin, « A boundary tilting perspective on the phenomenon of adversarial examples », in: *arXiv preprint arXiv:1608.07690* (2016).
- [228] Rhian Taylor, Varun Ojha, Ivan Martino, and Giuseppe Nicosia, « Sensitivity analysis for deep learning: ranking hyper-parameter influence », in: *33rd International Conference on Tools with Artificial Intelligence (ICTAI)*, IEEE, 2021.
- [229] « The MNIST database of handwritten digits », in: <http://yann.lecun.com/exdb/mnist/> ().

- 
- [230] Benedetta Tondi, Pedro Comesaña-Alfaro, Fernando Pérez-González, and Mauro Barni, « Smart Detection of Line-Search Oracle Attacks », in: *IEEE Transactions on Information Forensics and Security* (2017), DOI: 10.1109/TIFS.2016.2624280.
- [231] Florian Tramèr, Fan Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart, « Stealing Machine Learning Models via Prediction APIs », in: *USENIX Security Symposium*, 2016.
- [232] Brandon Tran, Jerry Li, and Aleksander Madry, « Spectral Signatures in Backdoor Attacks », in: *Advances in Neural Information Processing Systems*, ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, 2018.
- [233] Jean-Baptiste Truong, Pratyush Maini, Robert J Walls, and Nicolas Papernot, « Data-free model extraction », in: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021.
- [234] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry, « There Is No Free Lunch In Adversarial Robustness (But There Are Unexpected Benefits) », in: *ArXiv* (2018).
- [235] Chun-Chen Tu, Paishun Ting, Pin-Yu Chen, Sijia Liu, Huan Zhang, Jinfeng Yi, Cho-Jui Hsieh, and Shin-Ming Cheng, « AutoZOOM: Autoencoder-Based Zeroth Order Optimization Method for Attacking Black-Box Neural Networks », in: *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019.
- [236] Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh, « Embedding watermarks into deep neural networks », in: *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, 2017.
- [237] Vladimir Vapnik, Esther Levin, and Yann Le Cun, « Measuring the VC-dimension of a learning machine », in: *Neural computation* (1994).
- [238] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, « Attention is all you need », in: *Advances in neural information processing systems* (2017).
- [239] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio, « Manifold mixup: Better representations by interpolating hidden states », in: *International conference on machine learning*, PMLR, 2019.
- [240] Slava Voloshynovskiy, Taras Holotyak, and Fokko Beekhof, « Soft Content Fingerprinting With Bit Polarization Based on Sign-Magnitude Decomposition », in: *IEEE Transactions on Information Forensics and Security* (2015).

- 
- [241] Binghui Wang and Neil Zhenqiang Gong, « Stealing hyperparameters in machine learning », *in: Symposium on Security and Privacy (SP)*, IEEE, 2018.
- [242] C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I-H. Yeh, « CSPNet: A new backbone that can enhance learning capability of CNN », *in: CVPR Workshops*, 2020.
- [243] Haohan Wang, Xindi Wu, Zeyi Huang, and Eric P Xing, « High-frequency component helps explain the generalization of convolutional neural networks », *in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020.
- [244] Hung-Jui Wang, Yu-Yu Wu, and Shang-Tse Chen, « Enhancing Targeted Attack Transferability via Diversified Weight Pruning », *in: arXiv preprint arXiv:2208.08677* (2022).
- [245] Si Wang and Chip-Hong Chang, « Fingerprinting Deep Neural Networks - a DeepFool Approach », *in: 2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2021.
- [246] Siyue Wang, Xiao Wang, Pin-Yu Chen, Pu Zhao, and Xue Lin, « Characteristic Examples: High-Robustness, Low-Transferability Fingerprinting of Neural Networks », *in: Proc. of the Thirtieth Int. Joint Conference on Artificial Intelligence, IJCAI-21*, 2021.
- [247] Siyue Wang, Pu Zhao, Xiao Wang, Sang Chin, Thomas Wahl, Yunsi Fei, Qi A Chen, and Xue Lin, « Intrinsic Examples: Robust Fingerprinting of Deep Neural Networks », *in: British Machine Vision Conference (BMVC)*, 2021.
- [248] Wenxuan Wang, Bangjie Yin, Taiping Yao, Li Zhang, Yanwei Fu, Shouhong Ding, Jilin Li, Feiyue Huang, and Xiangyang Xue, « Delving into data: Effectively substitute training for black-box attack », *in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.
- [249] Xiaosen Wang, Xuanran He, Jingdong Wang, and Kun He, « Admix: Enhancing the transferability of adversarial attacks », *in: ICCV*, 2021.
- [250] Xiaosen Wang, Jiadong Lin, Han Hu, Jingdong Wang, and Kun He, « Boosting adversarial transferability through enhanced momentum », *in: BMVC*, 2021.
- [251] Xiaosen Wang, Zeliang Zhang, Kangheng Tong, Dihong Gong, Kun He, Zhifeng Li, and Wei Liu, « Triangle attack: A query-efficient decision-based adversarial attack », *in: European Conference on Computer Vision*, Springer, 2022.
- [252] Xinran Wang, Yu Xiang, Jun Gao, and Jie Ding, « Information Laundering for Model Privacy », *in: International Conference on Learning Representations*, 2021.

- 
- [253] Yixu Wang, Jie Li, Hong Liu, Yan Wang, Yongjian Wu, Feiyue Huang, and Rongrong Ji, « Black-Box Dissector: Towards Erasing-Based Hard-Label Model Stealing Attack », *in: ECCV*, 2022.
- [254] Zhibo Wang, Hengchang Guo, Zhifei Zhang, Wenxin Liu, Zhan Qin, and Kui Ren, « Feature importance-aware transferable adversarial attacks », *in: ICCV*, 2021.
- [255] Zi Wang, « Zero-shot knowledge distillation from a decision-based black-box model », *in: International Conference on Machine Learning*, PMLR, 2021.
- [256] « Watermarking security: theory and practice », *in: IEEE Transactions on signal processing* (2005).
- [257] Xingxing Wei, Ying Guo, and Jie Yu, « Adversarial sticker: A stealthy attack method in the physical world », *in: IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).
- [258] Ross Wightman, *PyTorch Image Models*, <https://github.com/rwightman/pytorch-image-models>, 2019, DOI: 10.5281/zenodo.4414861.
- [259] Eric Wong, Leslie Rice, and J. Zico Kolter, « Fast is better than free: Revisiting adversarial training », *in: International Conference on Learning Representations*, 2020, URL: <https://openreview.net/forum?id=BJx040EFvH>.
- [260] Han Wu, Syed Yunas, Sareh Rowlands, Wenjie Ruan, and Johan Wahlstrom, *Adversarial Driving: Attacking End-to-End Autonomous Driving*, 2023.
- [261] Lei Wu and Zhanxing Zhu, « Towards understanding and improving the transferability of adversarial examples in deep neural networks », *in: ACML*, PMLR, 2020, pp. 837–850.
- [262] Weibin Wu, Yuxin Su, Michael R Lyu, and Irwin King, « Improving the transferability of adversarial samples with adversarial transformations », *in: CVPR*, 2021, pp. 9024–9033.
- [263] Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L. Yuille, and Quoc V. Le, « Adversarial Examples Improve Image Recognition », *in: CVPR*, 2020.
- [264] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille, « Mitigating Adversarial Effects Through Randomization », *in: International Conference on Learning Representations*, 2018.
- [265] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille, « Improving transferability of adversarial examples with input diversity », *in: CVPR*, 2019.

- 
- [266] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le, « Self-Training With Noisy Student Improves ImageNet Classification », *in: CVPR*, 2020.
- [267] Yifeng Xiong, Jiadong Lin, Min Zhang, John E Hopcroft, and Kun He, « Stochastic Variance Reduced Ensemble Adversarial Attack for Boosting the Adversarial Transferability », *in: CVPR*, 2022.
- [268] Weilin Xu, David Evans, and Yanjun Qi, « Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks », *in: Proceedings 2018 Network and Distributed System Security Symposium* (2018).
- [269] Ying Xu, Xu Zhong, Antonio Jimeno Yepes, and Jey Han Lau, « Grey-box Adversarial Attack And Defence For Sentiment Classification », *in: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, 2021.
- [270] Metodi P. Yankov, Martin A. Olsen, Mikkel B. Stegmann, Søren Sk. Christensen, and Søren Forchhammer, « Fingerprint Entropy and Identification Capacity Estimation Based on Pixel-Level Generative Modelling », *in: IEEE Transactions on Information Forensics and Security* (2020).
- [271] Hengshuai Yao, « The Vanishing Decision Boundary Complexity and the Strong First Component », *in: arXiv preprint arXiv:2211.16209* (2022).
- [272] Honggang Yu, Kaichen Yang, Teng Zhang, Yun-Yun Tsai, Tsung-Yi Ho, and Yier Jin, « CloudLeak: Large-Scale Deep Learning Models Stealing Through Adversarial Examples », *in: NDSS Symposium*, 2020.
- [273] Honghai Yu and Pierre Moulin, « Regularized Adaboost Learning for Identification of Time-Varying Content », *in: IEEE Transactions on Information Forensics and Security* (2014).
- [274] Liping Yuan, Xiaoqing Zheng, Yi Zhou, Cho-Jui Hsieh, and Kai-Wei Chang, « On the Transferability of Adversarial Attacks against Neural Text Classifier », *in: arXiv preprint arXiv:2011.08558* (2020).
- [275] Hadi Zanddizari, Behnam Zeinali, and J Morris Chang, « Generating black-box adversarial examples in sparse domain », *in: Transactions on Emerging Topics in Computational Intelligence* (2021).
- [276] Santiago Zanella-Beguelin, Shruti Tople, Andrew Paverd, and Boris Köpf, « Grey-box Extraction of Natural Language Models », *in: Proceedings of the 38th International Conference on Machine Learning*, Proceedings of Machine Learning Research, PMLR, 2021.

- 
- [277] Chaoning Zhang, Philipp Benz, Adil Karjauv, and In So Kweon, « Data-free universal adversarial perturbation and black-box attack », in: *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 7868–7877.
- [278] Cheng Zhang and Pan Gao, « Countering adversarial examples: combining input transformation and noisy training », in: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.
- [279] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals, « Understanding deep learning (still) requires rethinking generalization », in: *Communications of the ACM* (2021).
- [280] Hanwei Zhang, Yannis Avrithis, Teddy Furon, and Laurent Amsaleg, « Walking on the edge: Fast, low-distortion adversarial examples », in: *IEEE Trans. on IFS* (2020).
- [281] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz, « mixup: Beyond Empirical Risk Minimization », in: *International Conference on Learning Representations*, 2018.
- [282] Jianping Zhang, Weibin Wu, Jen-tse Huang, Yizhan Huang, Wenxuan Wang, Yuxin Su, and Michael R Lyu, « Improving Adversarial Transferability via Neuron Attribution-Based Attacks », in: *CVPR*, 2022.
- [283] Jie Zhang, Bo Li, Jianghe Xu, Shuang Wu, Shouhong Ding, Lei Zhang, and Chao Wu, « Towards Efficient Data Free Black-Box Adversarial Attack », in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [284] Jingjing Zhao, Qingyue Hu, Gaoyang Liu, Xiaoqiang Ma, Fei Chen, and Mohammad Mehedi Hassan, « AFA: Adversarial fingerprinting authentication for deep neural networks », in: *Computer Communications* (2020).
- [285] Pu Zhao, Pin-Yu Chen, Siyue Wang, and Xue Lin, « Towards Query-Efficient Black-Box Adversary with Zeroth-Order Natural Gradient Descent », in: (2020).
- [286] Zhengli Zhao, Dheeru Dua, and Sameer Singh, « Generating Natural Adversarial Examples », in: *International Conference on Learning Representations*, 2018.
- [287] Zhengyu Zhao, Hanwei Zhang, Renjue Li, Ronan Sicre, Laurent Amsaleg, and Michael Backes, « Towards Good Practices in Evaluating Transfer Adversarial Attacks », in: *arXiv preprint arXiv:2211.09565* (2022).
- [288] Haizhong Zheng, Ziqi Zhang, Juncheng Gu, Honglak Lee, and Atul Prakash, « Efficient adversarial training with transferable adversarial examples », in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

- 
- [289] Mingyi Zhou, Jing Wu, Yipeng Liu, Shuaicheng Liu, and Ce Zhu, « Dast: Data-free substitute training for adversarial attacks », *in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.
- [290] Wen Zhou, Xin Hou, Yongjun Chen, Mengyun Tang, Xiangqi Huang, Xiang Gan, and Yong Yang, « Transferable adversarial perturbations », *in: ECCV*, 2018.
- [291] Andy Zou, Zifan Wang, J Zico Kolter, and Matt Fredrikson, « Universal and transferable adversarial attacks on aligned language models », *in: arXiv preprint arXiv:2307.15043* (2023).
- [292] Junhua Zou, Zhisong Pan, Junyang Qiu, Xin Liu, Ting Rui, and Wei Li, « Improving the transferability of adversarial examples with resized-diverse-inputs, diversity-ensemble and region fitting », *in: ECCV*, 2020.





# ROBIC [144]: A BENCHMARK SUITE FOR ASSESSING CLASSIFIERS ROBUSTNESS

---

## A.1 Introduction

The domain of evasion attacks is a very active research area. A deluge of research papers now proposes defenses to block such an attacker and adaptive attacks against these defenses. This is an endless arms race, and systematic benchmarks to evaluate the state of the threat are greatly required.

It is currently extremely difficult to have a clear view of what is truly working in this domain. The cliché is that no two papers report the same statistics for the same attack against the same model over the same image set. This is mostly due to that an attack is an algorithm with many parameters. Its power is indeed highly dependent on these parameters. These values are rarely specified in research papers. There exist benchmarks in the literature, such as ARES [51], RobustBench [38], RobustVision [118], ADBD [24]. They aim to provide a better understanding of the robustness of image classifiers. Yet, they fall short because their slowness prevents them from tackling large image datasets like ImageNet. They only operate on CIFAR-10 or MNIST. Also, they resort to attacks that are not all state-of-the-art.

This chapter develops ROBIC [144], a parameter-free benchmark to consider these concerns. It fairly evaluates the robustness of image classifiers using a new *half-distortion* measure. It gauges the robustness of the network against white and black box attacks, independently of its accuracy.

*All the notations unique to this chapter are listed in Table 7.10 at the conclusion of this manuscript. Additionally, for general notation, please refer to Table 7.9.*

---

## A.2 Difficulties

This section explains the difficulties of setting up a benchmark measuring the robustness of image classifiers.

### A.2.1 Notation

An attack is a process forging an image  $\mathbf{x}_{\text{adv}} = \mathcal{A}(\mathbf{x}_o, f, \Pi)$ , where  $\mathbf{x}_o$  is the original image,  $f$  is the target model, and  $\Pi$  is a set of attack parameters. The ground truth label of  $\mathbf{x}_o$  is denoted by  $y_o$ . The boolean function  $\mathcal{K}(\mathbf{x}_{\text{adv}}, y_o) = [\text{cl}(\mathbf{x}_{\text{adv}}) \neq y_o]$  tells whether the attack deludes classifier  $f$  in the untargeted attack scenario: the prediction  $f(\mathbf{x}_{\text{adv}})$  is not the ground truth. The distortion between  $\mathbf{x}_o$  and  $\mathbf{x}_{\text{adv}}$  is denoted by  $\text{dist}(\mathbf{x}_{\text{adv}}, \mathbf{x}_o)$ .

Some statistics like the probability of success and the average distortion are extracted from the adversarial images forged from the test set. They depend on the attack  $\mathcal{A}$  and its set of parameters  $\Pi$ . Therefore, it can not play the role of a measure of the robustness of a given model. The first difficulty is to get rid of the impact of parameters  $\Pi$ .

### A.2.2 The best-effort mode

The parameters  $\Pi$  have a huge impact on the power of an attack. For instance, some attacks like FGSM, I-FGSM [117], PGD [142] are distortion constrained in the sense that  $\Pi$  is strongly connected to a distortion budget. If this budget is small, the probability of the success of the attack is small. If it is large, this probability is close to 1 but the distortion is too big. Hence, it is hard to find the best setting to make these attacks competitive. Our strategy, so-called ‘best-effort mode’, reveals the intrinsic power of an attack by finding the best setting for any image:  $\mathbf{x}_{\text{adv}} = \mathcal{A}(\mathbf{x}_o, f, \Pi^*)$  with

$$\Pi^* = \arg \min_{\Pi: \mathcal{K}(\mathcal{A}(\mathbf{x}_o, f, \Pi), y_o)=1} \text{dist}(\mathcal{A}(\mathbf{x}_o, f, \Pi), \mathbf{x}_o). \quad (\text{A.1})$$

The best-effort mode makes the measurement of the robustness independent from an arbitrary global setting  $\Pi$ . Yet, it is costly in terms of computations. Attacks with few parameters are preferred since the search space is smaller.

---

### A.2.3 worst-case attacks

A second difficulty is to make the robustness score independent of the attack. Ideally, we would like to know the worst-case attack to certify the robustness of a model. An option proposed by benchmarks `RobustVision` [118] and `ARES` [51] is to consider a set of  $J = 11$  attacks as outlined in table A.1.

This is again costly as each image of the test set has to be attacked  $J$  times. Yet, a benchmark happens to be useful if it is fast enough to assess the robustness of many models. The best-effort mode over an ensemble of attacks is out of reach. This is the reason why we need to focus on fast worst-case attacks in the sense that they achieve their best-effort mode within limited complexity. Section A.4 focuses on these attacks.

### A.2.4 The choice of the metric

The game between attack  $\mathcal{A}$  and model  $f$  over the test set is summarized by the operating characteristic  $P$  defined in Equation 3.9 in Section 3.1.3. As reminder,  $P(D)$  is the fraction of images that the attack succeeded in hacking within a distortion budget  $D$ . Many benchmarks gauge the robustness by  $P(D_b)$  at an arbitrary distortion  $D_b$ : e.g. `RobustBench` [38] score is  $P(D = 0.5)$ . This measure is pointwise and dependent on  $\eta(0)$ .

## A.3 The benchmark

This section justifies the recommendations made in our benchmark and defines the measure of robustness.

**Pixel domain.** Our benchmark is dedicated to image classification. As a consequence, the distortion is defined on the pixel domain: An image  $\mathbf{x}$  is defined in the space  $\llbracket 0, 255 \rrbracket^D$  with  $D = N_c HW$  pixels for  $N_c$  color channels,  $H$  rows and  $W$  columns. Most papers in the field measure distortion after the transformation of the image in a tensor  $\mathbf{x} \in \mathcal{X}^D$ . This is a mistake preventing a fair comparison: for most models  $\mathcal{X} = [0, 1]$  but for some others  $\mathcal{X} = [-1, 1]$  or  $\mathcal{X} = [-3, 3]$ .

We outline that an adversarial image is above all an image, *i.e.* a discrete object  $\mathbf{x}_{\text{adv}} \in \llbracket 0, 255 \rrbracket^n$ . Again, most attacks output a continuous tensor  $\mathbf{x}_{\text{adv}} \in \mathcal{X}^D$ , neglecting the quantization. This is a mistake: in real life, the attacker has no access to  $x_a$ , which is auxiliary data internal to the model.

**Distortion.** The distortion measure is defined in Section 3.1.2

---

**Test set.** The input of the model is a natural and large image. Assessing the robustness of models on specific datasets like MNIST (almost black and white), or on tiny images like CIFAR does not reflect the complexity of the problem. Our benchmark considers natural images of at least  $224 \times 224$  pixels as provided in ImageNet.

**Measure of robustness.** Let us define the accuracy function  $\eta(D) := 1 - P(D)$ . The value  $\eta(0)$  is the classical accuracy of the model over the original images. Function  $\eta(D)$  is by construction non-increasing and should converge to 0 as the distortion  $D$  increases. After observing many accuracy functions  $\eta$  for different models and attacks, we notice that they share the same prototype:

$$\eta(D) = \eta(0) e^{-\lambda D} \quad \text{with } \lambda \in \mathbb{R}^+. \quad (\text{A.2})$$

Like in nuclear physics, we define the half-distortion  $D_{1/2}$  as the distortion needed to reduce to half the initial accuracy:

$$\eta(D_{1/2}) = \eta(0)/2, \quad D_{1/2} = \lambda^{-1} \log(2). \quad (\text{A.3})$$

This approximation is verified experimentally with an average coefficient of determination  $R^2$  of 99%. The half-distortion  $D_{1/2}$  will be the keystone of the proposed metric of robustness. A model is then characterized by three separate concepts: its generalization ability  $\eta(0)$  and its robustness  $D_{1/2}$  against black-box and white-box attacks.

## A.4 Fast Attacks

The recent trend in adversarial examples is to design fast attacks with state-of-the-art performances.

### A.4.1 Fast black-box attacks

In the black-box decision-based setup, the attacker can query a model and observe the predicted class. The complexity of the attack is gauged by the number of queries  $K$  needed to find an adversarial image of low distortion.

There has been a huge improvement in the number of queries recently. Brendel *et al.* report in the order of one million queries for one image in one of the first decision-based black-box BA [18, Fig. 6]. Then, the order of magnitude went down to tens of thousands [23, Fig. 4] [126, Fig. 5] and even some thousands in [187, Fig. 2]. Current benchmarks use other black-box

---

attacks, which are either decision-based (Square Attack [5] in RobustBench [38] is score-based), or not state-of-the-art (like Gaussian noise in RobustVision [118], or BA [18] in ARES [51]).

SurFree [146] and RayS [24] are the only *decision-based* papers with less than one thousand calls on ImageNet. Yet, RayS [24] is designed to minimize the  $\ell_\infty$  distortion, whereas SurFree [146] targets  $\ell_2$ . Section A.5 investigates which attack is the best candidate for a fast benchmark.

## A.4.2 Fast white-box attacks

In the white-box setup, the attacker can compute a loss function and its gradient thanks to auto-differentiation and back-propagation. The complexity is usually gauged by the number of gradient computations. Current benchmarks use different white-box attacks: RobustBench [38] relies on PGD [142] (with 2 parameters  $\Pi$ ), RobustVision [118] use DeepFool [156], and ARES [51] CW [22].

Again, the need for powerful but fast attacks is of utmost importance for a practical benchmark. A promising attack is BP [280, 15] designed for a low complexity budget. Its first stage finds an adversarial example as quickly as possible. It is nothing more than a gradient descent of the loss  $L$  with acceleration. At iteration  $t + 1$ :

$$\mathbf{x}_{\text{adv}}^{(t+1)} = \mathbf{x}_{\text{adv}}^{(t)} - \alpha \gamma(t+1) \eta \left( \nabla L(\mathbf{x}_{\text{adv}}^{(t)}) \right), \quad (\text{A.4})$$

where  $\mathbf{x}_{\text{adv}}^{(0)} = \mathbf{x}_o$ ,  $\eta(\mathbf{x}) = \mathbf{x} / \|\mathbf{x}\|_2$ , and  $\gamma(t)$  is a series of increasing values, hence the acceleration. Stage 1 finishes when  $\mathbf{x}_{\text{adv}}^{(t+1)}$  becomes adversarial. Stage 2 aims at lowering the distortion while maintaining the image adversarial (see [280]).

We develop a variant to aggressively downsize the number of gradient computations. Parameter  $\alpha$  is heuristically set up to 0.03 in [280]. This value is certainly too big for images close to the class boundary and too small for those further away. One costly option is the best effort mode which finds the best  $\alpha$  thanks to a line search (see Section A.2). We propose the following simple method inspired by DeepFool [156]. When applying (A.4) to the first-order approximation of the loss:

$$L(\mathbf{x}_o + p) \approx L(\mathbf{x}_o) + p^\top \nabla L(\mathbf{x}_o), \quad (\text{A.5})$$

---

then  $\eta(\nabla L(I_a^{(t)})) = \eta(\nabla L(\mathbf{x}_o))$  and BP cancels the loss for

$$\alpha = \frac{L(\mathbf{x}_o)}{\|\nabla L(\mathbf{x}_o)\|_2 \sum_{j=1}^{\kappa} \gamma(k)} \quad (\text{A.6})$$

within  $\kappa$  iterations. We fix  $\kappa = \lfloor K/3 \rfloor$  where  $K$  is the total iteration budget encompassing stages 1 and 2.

Section A.5 compares these attacks to identify the worst case.

### A.4.3 Quantization

The adversarial samples are quantified in the pixel domain to create images. The first option considers the quantization as post-processing not interfering with the attack. These options are tested on several black-and-white box attacks. The quantization will be post-processing for white-box attacks as recommended in [13], whereas the second option gives better results on black-box attacks.

## A.5 Experiments

All the attacks are run on  $n = 1000$  ImageNet images from the ILSVRC2012’s validation set with size  $D = 3 \times 224 \times 224$ .

### A.5.1 Selecting the worst case attacks

**Black box attacks:** Figure A.1 compares the evolution of the half-distortion (A.3) in function of the query amount for four decision-based black-box attacks: `SurFree` [146], `RayS` [24], `GeoDA` [187], and `QEBA` [126]. `SurFree` [146] and `RayS` [24] reach their best effort within  $K = 3000$  queries, while `QEBA` [126] and `GeoDA` [187] do not since their  $D_{1/2}$  still decrease after 5000 queries. Yet, `SurFree` [146] obtains quantified adversarial with much lower distortion. Therefore, our benchmark only needs this attack. The number of queries is kept at  $K = 5000$  to be sure to reach the optimal value of  $D_{1/2}$ .

**White box attacks:** Figure A.2 compares three white-box-attacks in the best effort mode: `PGD` [142], `CW` [22], and `BP` [280, 15] with our trick (A.6). They all reach the same  $D_{1/2}$  when given a large complexity budget. Yet, `BP` [280, 15] converges faster than the others. Our benchmark uses this version of `BP` [280, 15] to evaluate the white-box robustness.

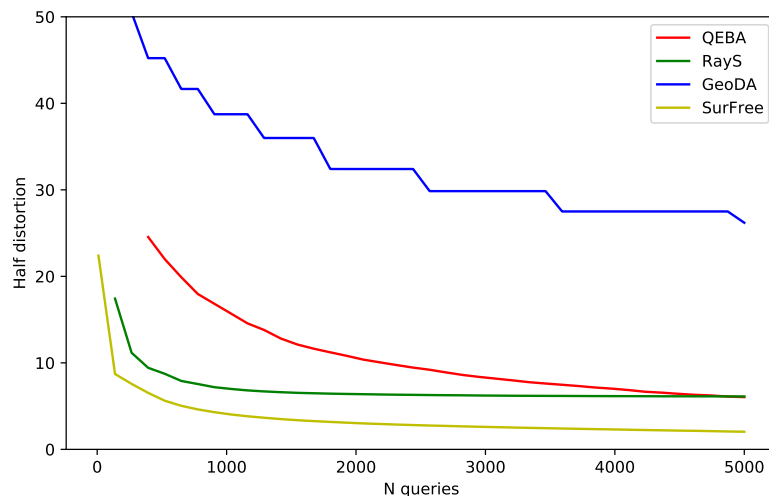


Figure A.1 – Evolution of  $D_{1/2}$  with the complexity budget for black box setup. Attacks on EfficientNet [224]

## A.5.2 Comparison with other benchmarks

Table A.1 lists several benchmarks. Most of them evaluate the robustness as the success rate under a prescribed  $l_2$  or  $l_\infty$  distortion budget. But, these budgets are set arbitrarily or even not constant within the same benchmark for RobustML. Our half-distortion (A.3) is parameter-free. It returns an accurate, reliable and fair measurement of robustness.

Some benchmarks need many attacks to get a full vision of the robustness: ARES [51] and RobustVision [118] use 11 attacks. This is too time-consuming. On the contrary, ADBD [24] focuses on a single black-box attack, which is indeed outdated. RobustBench [38] condenses four attacks in one measure elegantly: for a given image, if the first simple attack does not

| Benchmark          | Domain         | Nb. attacks  | Measures  | Runtime  |
|--------------------|----------------|--------------|---|----------|
| RoBIC [144]        | $[[0, 255]]^D$ | 1 WB + 1 BB  | Half-distortion $l_2$                                 | 43s      |
| RobustBench [38]   | $[0, 1]^D$     | 3 WB + 1 BB  | Success-Rate for fixed budget ( $l_2$ or $l_\infty$ ) | 48s      |
| ADBD [24]          | $[0, 1]^D$     | 1 BB         | Distance $l_\infty$                                   | 360s     |
| RobustVision [118] | $[0, 1]^D$     | 6 WB + 5 BB  | Median Distance $l_2$                                 | 200s     |
| ARES [51]          | $[0, 1]^D$     | 5 WB + 10 BB | Success-Rate vs Budget ( $l_2, l_\infty$ or queries)  | Too long |

Table A.1 – Benchmarks Comparison. Average Runtimes per ImageNet Image with ResNet50 [142].



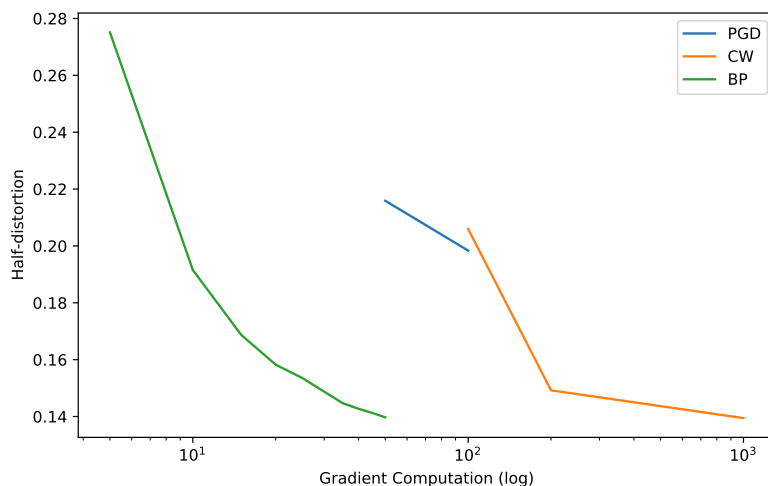


Figure A.2 – Evolution of  $D_{1/2}$  with the complexity budget for white box setup. Attacks on EfficientNet [224]

succeed within the distortion budget, then the second more complex one is launched *etc.* The total runtime heavily depends on the distortion budget. Yet, black-box and white-box attacks use different mechanisms. Our benchmark reports a measurement for each separately.

### A.5.3 Benchmarking models

Table A.2 compares standard models from *timm* [258] and *torchvision* [150] libraries. Here are some intriguing results.

**Robustness in white box vs. black box.** One does not imply the other. Figure A.3 even shows a negative correlation. However, some models escape this rule. For instance, VGG16 is neither robust in black-box nor in white-box. EfficientNet AdvProp [263] follows the opposite trend. We believe that black-box robustness reveals the complexity of the borders between classes, and white-box robustness indicates how close natural images are to the borders. This highlights the importance of having two different measurements.

**The importance of the training procedure.** There is on average a factor of 20 between the half-distortions in white and black box. This factor drops to 4 and 10 for the models adversarially trained: ResNet50 [142], EfficientNet AdvProp [263].

Table A.2 lists four EfficientNet models sharing the same architecture but different training procedures. Their accuracies are similar but there is up to a factor of 2 between the robustnesses. The same holds for the three variants of Resnet50. The gaps in accuracy and robustness are noticeable with standard models from *timm* [258] and *torchvision* [150]. It is even more visible

| Model                            | Parameters<br>(millions) | Accuracy<br>$\eta(0)$ | $D_{1/2}$   |             |
|----------------------------------|--------------------------|-----------------------|-------------|-------------|
|                                  |                          |                       | white box   | black box   |
| AlexNet [114]                    | 62.38                    | 56.8                  | 0.19        | 2.17        |
| CSPResNeXt50 [242]               | 20.57                    | 84.6                  | 0.13        | 4.48        |
| DualPathNetworks 68b [132]       | 12.61                    | 83.8                  | 0.08        | 3.82        |
| MixNet Large [225]               | 7.33                     | 84.2                  | 0.12        | 2.96        |
| MobileNetV2 [201]                | 5.83                     | 80.1                  | 0.09        | 2.90        |
| ReXNet 200 [40]                  | 16.37                    | 85.4                  | 0.14        | 3.89        |
| RegNetY 032 [184]                | 19.44                    | 85.8                  | 0.11        | 4.94        |
| SEResNeXt50 32x4d [92]           | 27.56                    | <b>85.9</b>           | 0.12        | <b>5.01</b> |
| VGG16 [214]                      | 138.00                   | 74.9                  | 0.09        | 2.44        |
| EfficientNet AdvProp [263]       | 5.29                     | 84.3                  | <b>0.31</b> | 4.35        |
| EfficientNet EdgeTPU Small [224] | 5.44                     | 82.8                  | 0.15        | 3.16        |
| EfficientNet NoisyStudent [266]  | 5.29                     | 82.7                  | 0.19        | 2.37        |
| EfficientNet [224]               | 5.29                     | 82.8                  | 0.17        | 3.56        |
| ResNet50 (torchvision) [81]      | 25.56                    | 77.9                  | 0.10        | 2.77        |
| ResNet50 (timm) [81]             | 25.56                    | 80.5                  | 0.15        | 4.35        |
| ResNet50 AdvTrain [142]          | 25.56                    | 60.8                  | <b>2.56</b> | <b>9.88</b> |

Table A.2 – Benchmark of models with 1,000 ImageNet Images

with adversarial training from [142]: the gain in robustness is impressive but at the cost of a big drop in accuracy.

## A.6 Conclusion

The chapter introduces a rigorous benchmark based on a new and independent measurement of robustness: the half distortion.  $\text{ROBIC}$  [144] is faster than the other benchmarks. This allows for tackling larger images which is more realistic.

In addition to the accuracy,  $\text{ROBIC}$  [144] gives the black box robustness and white box robustness. We believe that the first indicates how far away the class boundaries lie from the images whereas the last reflects how curved are the boundaries. As with the other benchmarks, two limitations hold: The network must be differentiable to run a white box attack, and deterministic to run a black box attack.

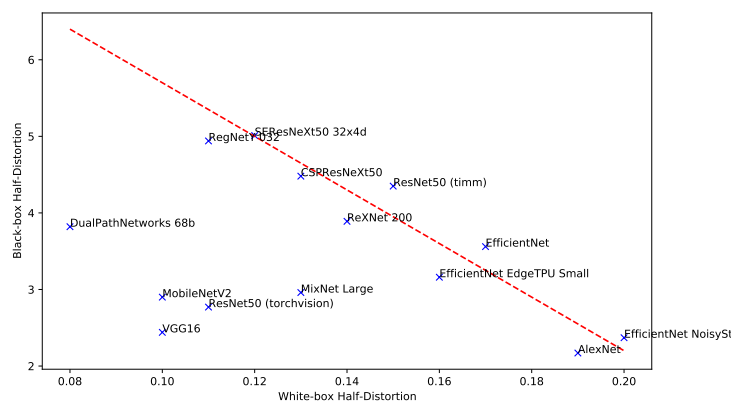


Figure A.3 – Black-box  $D_{1/2}$  as a function of white-box  $D_{1/2}$ .

# APPENDIX: FINGERPRINTING CLASSIFIERS WITH BENIGN INPUTS

---

## B.1 Description of the Set of Models

The core of the set  $\mathcal{A}$  contains 35 ‘off-the-shelves’ vanilla models which were trained with supervision for the task ImageNet over one million annotated  $224 \times 224$  pixel images. These models come from the Timm [258] and Torchvision [150] libraries. Among them, some are very close. For instance, there are 6 versions of EfficientNet-b0. These models share the same architecture but they result from different training session. Yet, they are considered as different models and this proves the efficiency of our method.

To forge a variant, we have selected 8 transformations. These are simple procedures easily applicable by Bob. They move the decision boundary of the model with a limited drop in accuracy. As such, all of them have already proven themselves as a defense against adversarial examples [185].

- **Identity:** The variant is an exact copy of the model.
- **Model Precision:** Deep neural networks usually encode weights and biases on 32 bits floating point precision. The Torch class attribute `half` is used to reduce the precision to 16 bits.
- **JPEG Compression:** Before being classified, the input image goes through a JPEG compression. This has been proposed as a defense: JPEG coarsely quantizes the high frequencies while adversarial perturbations are essentially composed of high frequencies. JPEG compression and decompression act as a reformer [207, 75, 41]. No training is needed because models are robust to JPEG compression. The quality factor ranges from 30 to 90 in step of 10.
- **Histogram Equalizer:** This increases the contrast in an image and it has been proposed as a defense against adversarial perturbation in [185, 177]. We use the function `transforms.functional.equalize` from Torchvision.

- 
- **Color Depth Reduction:** Another defense is to reduce the depth of color channels to less than 8 bits [268], here from 3 and 7 bits. The image is posterized with the function `transforms.functional.posterize` from Torchvision.
  - **Randomized Smoothing:** Randomized smoothing provides robustness guarantees [33, 122, 125] and is also efficient against black-box attacks [145]. We take the Github implementation<sup>1</sup> of [33]. The number of samples is set to 100 as recommended in the previous works. The standard deviations  $\sigma$  selected are: 0.01, 0.02, 0.04, 0.06, 0.08, 0.1.
  - **Finetuning:** Finetuning updates the weights of the model during a new training. We consider finetuning all the layers or only the last one. Finetuning runs over 50 epochs with SGD optimizer.
  - **Pruning:** Pruning compresses the model by removing the less important weights. It is applied with the function `nn.utils.prune.l1_unstructured` from Torch package. It removes the weights with the lowest  $\ell_1$  norm. Pruning can be applied on all the layers or just some particular ones. Filter pruning [127] cuts the less important output channels of the convolutional layers. We consider three options with the following fraction of weights removed:
    - Pruning All layers: 1%, 2%, 3%, 4%
    - Filter Pruning: 10%, 20%, 30%,
    - Pruning the Last layer: 70%, 80%, 90%, 95%

This makes a total of 33 procedures, most of them being easily and quickly applicable to any vanilla model. Few of them imply a light retraining which is done with a subset of 50,000 images of the ImageNet validation set. We end up with 1189 variants. The accuracy of each of them is measured on the remaining part of the ImageNet validation set. If the drop of accuracy is bigger than 15% compared to the original model, then the variant is discarded. Our final collection contains 1081 models and variants.

Alice has a collection of 20.000 images randomly taken from the ImageNet test set. These images are not annotated with a ground truth. These were not used for training the models or retraining the variants.

## B.2 Distance between Models

The distances between all the pairs of models is shown in Figure B.1. The block diagonal shows that the distances between variants of the same vanilla models are small. This distance

---

1. Randomized Smoothing GitHub: <https://github.com/locuslab/smoothing>

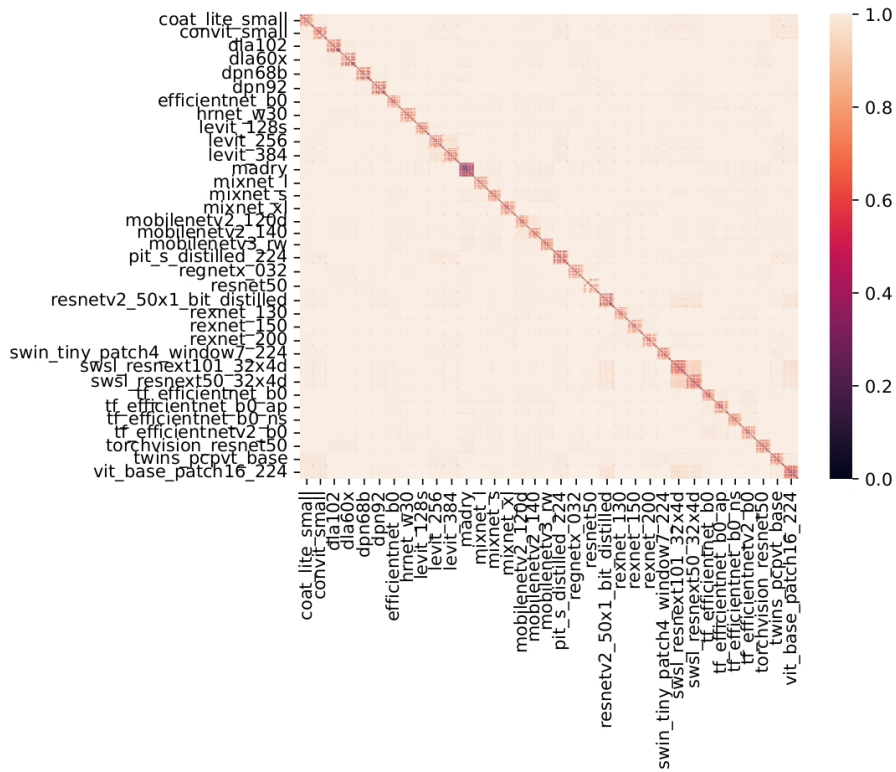


Figure B.1 – Distances between pairs of models.

matrix is the input of the t-SNE algorithm which creates the 2D representation of Figure 7.1. We clearly see the cluster of variants centered on each vanilla model.

### B.3 Proof of Equation (10)

Suppose that Bob selects  $b \in \mathcal{F}_j$ . Alice makes a random permutation  $\sigma$  of the  $n_{\mathcal{F}}$  families and sequentially tests them. Then, she spends  $L_i^{\text{neg}}$  queries to discover that the black-box does not belong to  $\mathcal{F}_i$ , for any  $i$  s.t.  $\sigma(i) < \sigma(j)$  (i.e. the families ranked by the permutation before  $\mathcal{F}_j$ ) and  $L_j^{\text{pos}}$  queries to discover that the black-box is a member of  $\mathcal{F}_j$ . Over all the random permutations, there is statistically one chance out of two that  $\mathcal{F}_i$  is ranked before  $\mathcal{F}_j$ . On

expectation, this makes the following number of queries:

$$\mathbb{E}(L_j^{\text{pos}}) + \frac{1}{2} \sum_{i \neq j} \mathbb{E}(L_i^{\text{neg}}) = \mathbb{E}(L_j^{\text{pos}}) + \frac{1}{2} \left( -\mathbb{E}(L_j^{\text{neg}}) + \sum_{i=1}^{n_{\mathcal{F}}} \mathbb{E}(L_i^{\text{neg}}) \right). \quad (\text{B.1})$$

We now suppose that Bob select a family uniformly at random, to obtain the following average:

$$\mathbb{E}(L) = \frac{1}{n_{\mathcal{F}}} \sum_{j=1}^{n_{\mathcal{F}}} \mathbb{E}(L_j^{\text{pos}}) + \frac{n_{\mathcal{F}} - 1}{2n_{\mathcal{F}}} \sum_{j=1}^{n_{\mathcal{F}}} \mathbb{E}(L_j^{\text{neg}}). \quad (\text{B.2})$$

## B.4 Lower bound of the distance dist

This section provides an example of the computation of the distance between the outputs of model  $m$  and the black-box  $b$ . It considers that these classifiers yield only top-1 outputs. We assume that the surjection  $S_1$  is defined in (7.14) w.r.t. the ground truth:  $\tilde{z} = 1$  when the black-box predict the ground truth. The joint probability distribution is denoted by:

|                                    |                 |                 |
|------------------------------------|-----------------|-----------------|
| $\mathbb{P}(\tilde{Z}, \tilde{Y})$ | $\tilde{Y} = 0$ | $\tilde{Y} = 1$ |
| $\tilde{Z} = 0$                    | $a$             | $b$             |
| $\tilde{Z} = 1$                    | $c$             | $1 - a - b - c$ |

We suppose that the accuracies of both models are known,  $\eta(m) = A$  and  $\eta(b) = B$ , so that:

$$\eta(m) = \mathbb{P}(\tilde{Y} = 1) = 1 - a - c = A, \quad (\text{B.3})$$

$$\eta(b) = \mathbb{P}(\tilde{Z} = 1) = 1 - a - b = B. \quad (\text{B.4})$$

These equations are constraints reducing the problem with three unknown parameters  $(a, b, c)$  to a single one: From  $a$ , we can easily deduce  $(b, c)$  from the above equations. Since these joint probabilities are between 0 and 1, this implies that

$$\max(0, 1 - (A + B)) \leq a \leq \min(1 - A, 1 - B). \quad (\text{B.5})$$

The mutual information between  $\tilde{Z}$  and  $\tilde{Y}$  is given by  $I(\tilde{Z}; \tilde{Y}) = H(\tilde{Z}) + H(\tilde{Y}) - H(\tilde{Z}, \tilde{Y})$ ,

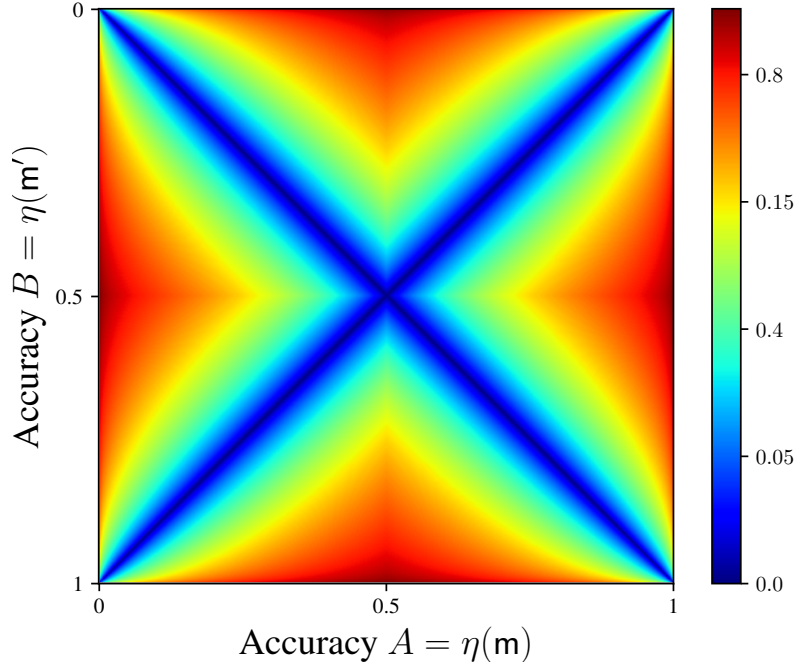


Figure B.2 – Lower bound of the distance between two models of accuracy  $A$  and  $B$  under top-1.

so that the distance can be written as

$$\text{dist}(b, m) = 2 - \frac{h(A) + h(B)}{H(\tilde{Z}, \tilde{Y})} \quad (\text{B.6})$$

$$= 2 - \frac{h(A) + h(B)}{\sum_{(z, \tilde{y}) \in \{0,1\}^2} f(\mathbb{P}(\tilde{Z} = z, \tilde{Y} = \tilde{y}))} \quad (\text{B.7})$$

where  $f(x) := -x \log_2 x$  and  $h(p) := f(x) + f(1-x)$  is the binary entropy in bits for  $p \in [0, 1]$ . Thanks to the constraints (B.3) and (B.4), the distance is a function of  $a$  whose derivative cancels at only one value giving  $\text{dist}(b, m) = 1$  achieved when  $\tilde{Z}$  and  $\tilde{Y}$  are independent, *i.e.*

$$a = (1 - A)(1 - B), \quad b = A(1 - B), \quad c = (1 - A)B, \quad d = AB.$$

This is thus a global maximum and the minimum of  $\text{dist}(b, m)$  lies on the boundary of the range of  $a$ . Suppose that  $A \geq B$  and  $A + B > 1$ , then

$$\begin{aligned} H(\tilde{Z}; \tilde{Y}) &\geq f(1 - A) \\ &+ \min(f(B) + f(A - B), f(1 - B) + f(A + B - 1)) \quad (\text{B.8}) \end{aligned}$$



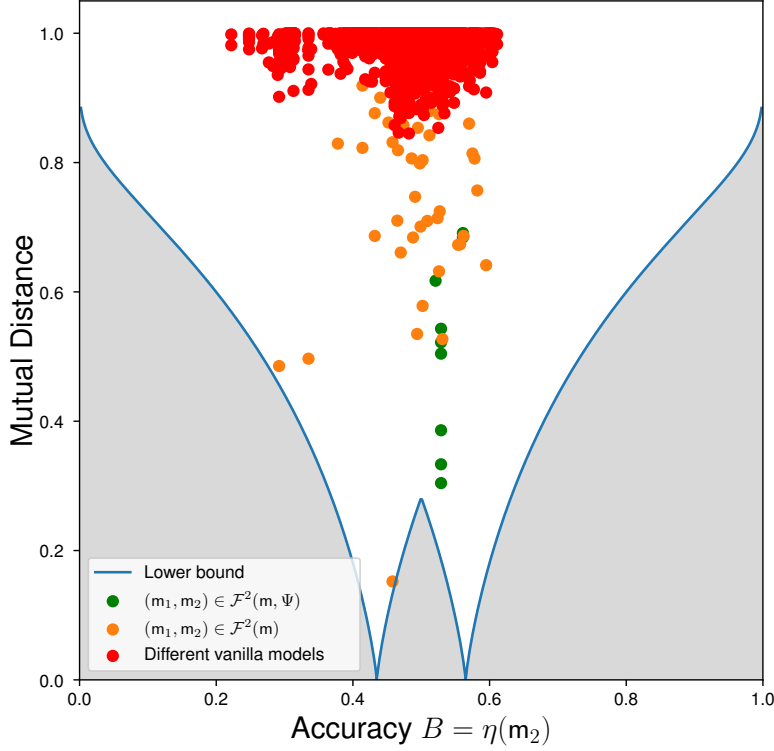


Figure B.3 – Comparison of the theoretical lower bound and the measured distance between models  $m_1$  and  $m_2$ , when  $A = \eta(m_1) = 0.45$  and  $(m_1, m_2) \in \mathcal{F}^2(m)$  (orange),  $(m_1, m_2) \in \mathcal{F}^2(m, \Psi)$  (green), or  $m_1$  and  $m_2$  are variants of different vanilla models (red).

which is converted into a lower bound of the distance  $\text{dist}(b, m)$

$$2 - \frac{h(A) + h(B)}{f(1 - A) + \min(f(B) + f(A - B), f(1 - B) + f(A + B - 1))}.$$

This function is illustrated in Figure B.3 where we see that  $\text{dist}(b, m)$  may cancel only when  $A = B$  or  $A = 1 - B$ . Figure B.3 compares this lower bound with the actual measurements for  $A = 0.45$ . It shows that most attacks considered are far from being the worst case. We also plot the performance of the following attack: Bob modifies the decision of model  $m$  with a probability  $\epsilon \in [0, 1/2]$ , which makes the following distribution:

| $\mathbb{P}(\tilde{Z}, \tilde{Y})$ | $\tilde{Y} = 0$         | $\tilde{Y} = 1$   |
|------------------------------------|-------------------------|-------------------|
| $\tilde{Z} = 0$                    | $(1 - A)(1 - \epsilon)$ | $A\epsilon$       |
| $\tilde{Z} = 1$                    | $(1 - A)\epsilon$       | $1(1 - \epsilon)$ |

Table B.1 – List of the 35 vanilla models and their variants. *all* means that all values listed in Section B.1 comply with (7.1).

|  | Nb of variants | Finetuning |      | Half Precision | Histogram | JPEG | Posterize  | Prune            |          | Randomized Smoothing |                              |
|--|----------------|------------|------|----------------|-----------|------|------------|------------------|----------|----------------------|------------------------------|
|  |                | All        | Last |                |           |      |            | All              | Filter   |                      | Last                         |
| <i>CoatLiteSmall</i>                       | 29             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01             | all      | 0.7, 0.8, 0.9        | all                          |
| <i>ConViT<sub>small</sub></i>              | 28             | ✓          | ✓    |                | ✓         | all  | all        | 0.01             | all      | 0.7, 0.8, 0.9        | all                          |
| <i>DLA<sub>102</sub></i>                   | 30             | ✓          | ✓    | ✓              |           | all  | 4, 5, 6, 7 | all              | all      | all                  | 0.01, 0.02, 0.04, 0.06, 0.08 |
| <i>DLA<sub>60</sub></i>                    | 32             | ✓          | ✓    | ✓              |           | all  | all        | all              | all      | all                  | all                          |
| <i>DLA<sub>68b</sub></i>                   | 32             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01, 0.02, 0.03 | all      | all                  | all                          |
| <i>DPN<sub>92</sub></i>                    | 33             | ✓          | ✓    | ✓              | ✓         | all  | all        | all              | all      | all                  | all                          |
| <i>(Torch) EfficientNet<sub>b0</sub></i>   | 28             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01, 0.02       | 0.1, 0.2 | 0.7, 0.8, 0.9        | 0.01, 0.02, 0.04, 0.06, 0.08 |
| <i>(TF) EfficientNet<sub>b0</sub></i>      | 25             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01             | 0.1, 0.2 | 0.7, 0.8             | 0.01, 0.02, 0.04             |
| <i>(TF) EfficientNet<sub>b0,AP</sub></i>   | 28             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01, 0.02       | 0.1, 0.2 | 0.7, 0.8             | all                          |
| <i>(TF) EfficientNet<sub>b0,NS</sub></i>   | 28             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01, 0.02       | 0.1, 0.2 | 0.7, 0.8, 0.9        | 0.01, 0.02, 0.04, 0.06, 0.08 |
| <i>(TF) EfficientNetV<sub>2b0</sub></i>    | 30             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01, 0.02, 0.03 | all      | 0.7, 0.8, 0.9        | 0.01, 0.02, 0.04, 0.06, 0.08 |
| <i>HRNet<sub>v30</sub></i>                 | 31             | ✓          | ✓    | ✓              |           | all  | all        | all              | all      | all                  | 0.01, 0.02, 0.04, 0.06, 0.08 |
| <i>LeViT<sub>128s</sub></i>                | 28             |            | ✓    | ✓              | ✓         | all  | all        | 0.01             | 0.1, 0.2 | all                  | all                          |
| <i>LeViT<sub>256</sub></i>                 | 30             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01, 0.02       | all      | all                  | all                          |
| <i>LeViT<sub>384</sub></i>                 | 32             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01, 0.02, 0.03 | all      | all                  | all                          |
| <i>(Torch) ResNet50</i>                    | 28             | ✓          | ✓    | ✓              |           | all  | 4, 5, 6, 7 | all              | all      | 0.7, 0.8, 0.9        | 0.01, 0.02, 0.04, 0.06       |
| <i>ResNet50<sub>madry</sub></i>            | 30             | ✓          | ✓    | ✓              |           | all  | all        | all              | all      | 0.7, 0.8, 0.9        | 0.01, 0.02, 0.04, 0.06, 0.08 |
| <i>MixNet<sub>1</sub></i>                  | 29             | ✓          | ✓    | ✓              | ✓         | all  | 4, 5, 6, 7 | 0.01, 0.02, 0.03 | all      | 0.7, 0.8, 0.9        | 0.01, 0.02, 0.04, 0.06, 0.08 |
| <i>MixNet<sub>s</sub></i>                  | 25             | ✓          | ✓    | ✓              |           | all  | 4, 5, 6, 7 | 0.01, 0.02       | 0.1, 0.2 | 0.7, 0.8, 0.9        | 0.01, 0.02, 0.04, 0.06, 0.08 |
| <i>MixNet<sub>xl</sub></i>                 | 32             | ✓          | ✓    | ✓              | ✓         | all  | all        | all              | all      | 0.7, 0.8, 0.9        | all                          |
| <i>MobileNetV<sub>2</sub>120d</i>          | 27             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01, 0.02, 0.03 | 0.1, 0.2 | 0.7, 0.8             | 0.01, 0.02, 0.04, 0.06       |
| <i>MobileNetV<sub>2</sub>140</i>           | 25             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01, 0.02       | 0.1      | 0.7, 0.8             | 0.01, 0.02, 0.04, 0.06       |
| <i>MobileNetV<sub>3</sub><sub>rw</sub></i> | 25             | ✓          | ✓    | ✓              |           | all  | 4, 5, 6, 7 | 0.01, 0.02, 0.03 | 0.1, 0.2 | 0.7, 0.8             | 0.01, 0.02, 0.04, 0.06       |
| <i>PiT<sub>s, distilled</sub></i>          | 33             | ✓          | ✓    | ✓              | ✓         | all  | all        | all              | all      | all                  | all                          |
| <i>RegNetX<sub>32</sub></i>                | 31             | ✓          | ✓    | ✓              |           | all  | all        | all              | all      | 0.7, 0.8, 0.9        | all                          |
| <i>ResNet50</i>                            | 32             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01, 0.02, 0.03 | all      | all                  | all                          |
| <i>BiT<sub>R50x1</sub></i>                 | 32             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01, 0.02, 0.03 | all      | all                  | all                          |
| <i>ReXNet<sub>130</sub></i>                | 30             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01, 0.02, 0.03 | 0.1, 0.2 | 0.7, 0.8, 0.9        | all                          |
| <i>ReXNet<sub>150</sub></i>                | 31             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01, 0.02, 0.03 | 0.1, 0.2 | all                  | all                          |
| <i>ReXNet<sub>200</sub></i>                | 31             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01, 0.02, 0.03 | 0.1, 0.2 | all                  | all                          |
| <i>SwinT</i>                               | 28             | ✓          | ✓    | ✓              | ✓         | all  | all        |                  | all      | 0.7, 0.8, 0.9        | all                          |
| <i>ResNeXt<sub>101,32x4d</sub></i>         | 33             | ✓          | ✓    | ✓              | ✓         | all  | all        | all              | all      | all                  | all                          |
| <i>ResNeXt<sub>50,32x4d</sub></i>          | 33             | ✓          | ✓    | ✓              | ✓         | all  | all        | all              | all      | all                  | all                          |
| <i>Twins – PCPVT<sub>base</sub></i>        | 28             | ✓          | ✓    | ✓              | ✓         | all  | all        | all              | all      | 0.7, 0.8, 0.9        | all                          |
| <i>ViT<sub>base, patch=16</sub></i>        | 29             | ✓          | ✓    | ✓              | ✓         | all  | all        | 0.01, 0.02       | all      | 0.7, 0.8             | all                          |



# APPENDIX: HOW TO CHOOSE YOUR BEST ALLIES FOR A TRANSFERABLE ATTACK?

---

## C.1 Experimental Setup

In this study, we evaluated the transferability of adversarial examples on a diverse set of 48 models trained for image classification on the ImageNet dataset with over one million annotated  $224 \times 224$  images. The models were obtained from the Timm library [258], with the exception of `ResNet50AdvTrain`, which was obtained from the GitHub repository of the original paper<sup>1</sup>. To ensure adequate representation, we randomly selected models from each architecture, with a minimum of three models per architecture. The only exception was the ReXNet architecture, which had two distinct models. The 48 selected models are:

- ConViT architecture: `ConViTbase`, `ConViTsmall`, `ConViTtiny`
- LeViT architecture: `LeViT192`, `LeViT256`, `LeViT128`
- DenseNet architecture: `DenseNet169`, `DenseNet121`, `DenseNet161`
- PiT architecture: `PiTsmall`, `PiTtight`, `PiTtight-dist`, `PiTsmall-dist`
- MobileNet architecture (V2): `MobileNetV2110d`, `MobileNetV2100`, `MobileNetV2120d`
- CoaT architecture: `CoatLitetiny`, `CoatLitemini`, `CoatLitesmall`
- xCiT architecture: `xCiTmedium`, `xCiTnano`, `xCiTsmall`
- Twins architecture: `Twinssmall`, `Twinslarge`, `Twinsbase`,
- MixNet architecture: `MixNetlarge`, `MixNetsmall`, `MixNetmedium`, `MixNetsmall-TensorFlow`,  
`MixNetlarge-TensorFlow`, `MixNetmedium-TensorFlow`
- EfficientNet architecture: `EfficientNetB0`, `EfficientNetB0AdvProp`, `EfficientNe`
- ResNet architecture: `ResNet50`, `ResNet50d`, `ResNet50AdvTrain`
- ResNetV2 architecture: `ResNetV250x1-dist`, `ResNetV2101`, `ResNetV250`
- ReXNet architecture: `RexNet150`, `RexNet130`

---

1. <https://github.com/MadryLab/robustness>

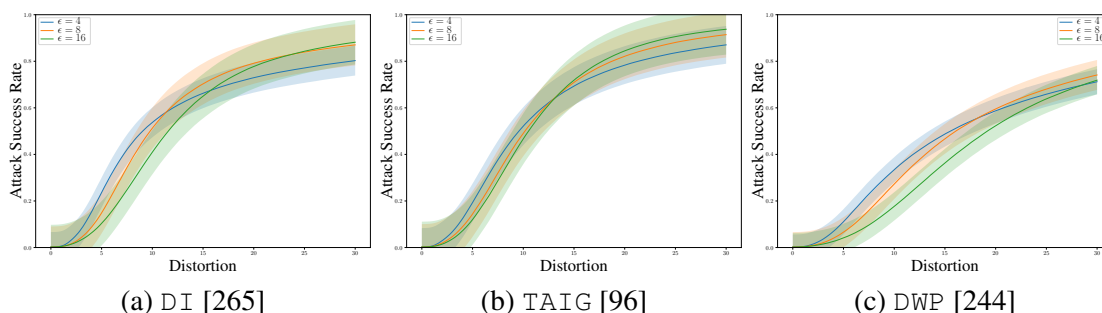


Figure C.1 – Attack Success Rate function of the perturbation norm for different values of  $\epsilon$ .

- DPN architecture: DPN<sub>92</sub>, DPN<sub>107</sub>, DPN<sub>68b</sub>
- DLA architecture: DLA<sub>60</sub>, DLA<sub>102</sub>, DLA<sub>169</sub>

## C.2 Preliminaries

### C.2.1 Epsilon Parameter

All transferable attacks share a common parameter  $\epsilon$ . It controls the maximum perturbation norm added on a single pixel for the adversarial example built. Figure C.1 demonstrates the ASR obtained for various values of  $\epsilon$  as a function of the perturbation norm. It shows that even if more freedom is given to the perturbation, in the sense that a larger maximum perturbation norm is allowed, the transferable directions remain consistent. Irrespective of the value of  $\epsilon$  for a given attack, all scores for a given norm of the perturbation are similar.

## C.3 Transferability Dependences

The 48 models considered in C.1 are evaluated as both sources and targets in this study. For each possible pair of models, each source model is evaluated for its ability to transfer to each target model. This results in a total of  $48^2 = 2304$  evaluations. The transferability is evaluated using the score defined in 5.2.2 and their matrices for the attacks DI [265], TAIG [96], and DWP [244] are presented in C.2. Each matrix exhibits a similar structure, with models that have high transferability values appearing in each matrix. However, the values achieved are different for each attack. The DI [265] and TAIG [96] attacks achieve higher values than DWP [244], indicating that these attacks create better quality transferable examples.

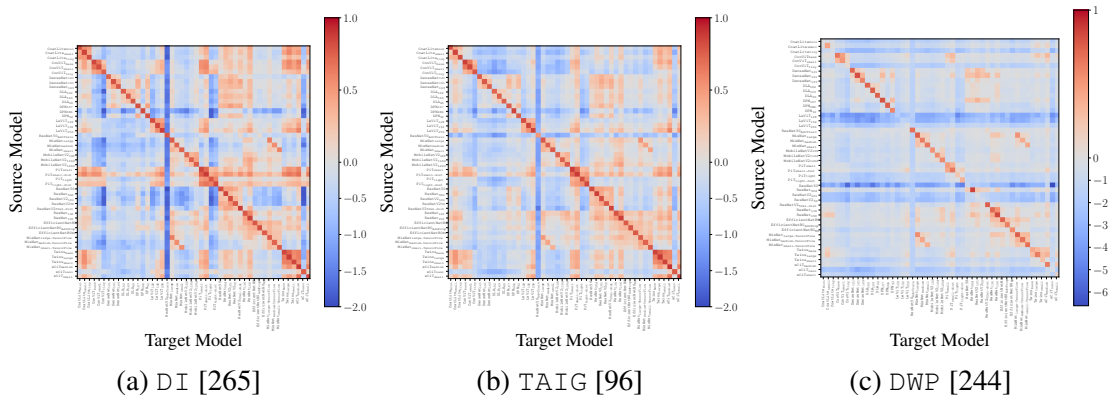


Figure C.2 – Transferability score  $\hat{T}_{s,t}$  matrix of 48 sources and 48 targets listed in C.1 for DI [265], TAIG [96] and DWP [244].

## C.4 Results

### C.4.1 Fingerprinting

Transferability can be divided into three components: the attack, the model, and the attacked image. To estimate transferability, the `Fit` measure defined in 5.2.2 first estimate the similarity between the source and the target models. In a defensive scenario, fingerprinting methods have been proposed to estimate model similarity without accessing one of the models. These methods do not modify the model during training but instead take an already trained model and find images that are its signatures. They usually generate adversarial examples specially designed for this model [19, 141, 178]. `FBI` [147] is the only method using benign images to assess the similarity of two models by measuring the independence between the two models using mutual information. All fingerprinting methods are sensitive to the number of images used for fingerprinting. More images lead to more accurate similarity scores, but they also have a cost. In the scenario considered here, the number of images submitted must be minimized. Figure C.3 shows the  $\hat{T}_{s,t}$  function of the number of images used for `FBI` [147]. Increasing the number of images submitted provides a better estimation of the transferability. The score reaches a plateau at 200 images submitted.

### C.4.2 Ensemble model attack

When attackers have access to multiple models, they can perform an ensemble-model attack to generate transferable adversarial examples. This approach has been shown to offer better

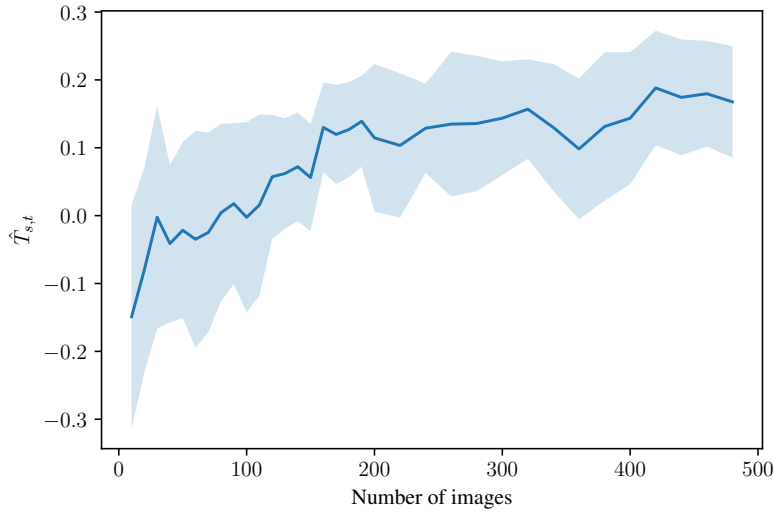


Figure C.3 –  $\hat{T}_{s,t}$  function of the number of images used for FBI [147] to estimate the transferability between 45 sources and one target. Adversarial obtained with DI [265] and  $\epsilon = 8$ .

transferability than the best single-model attack. However, existing methods for performing ensemble-model attacks have only been evaluated with a limited number of source models, typically with a maximum of three models. In this chapter, a high number of models is used to build large ensemble-model attacks in the scenario described in the experimental setup in 5.3.1. At each step of the attack, a model is randomly selected from the available sources and added to the ensemble-model. To build transferable adversarial examples, the logits of the models are averaged together, as proposed in [250]. Transferability is computed for ensemble-model attacks of up to 20 models. Figure C.4 shows the FiT score as a function of the ensemble-model size and compares the results with FiT scores obtained by selecting the three best models for the ensemble-model among the 20 models available. Ensemble-model attacks demonstrate significant improvements when only a few models are considered, but beyond 5 models, the improvements become negligible. Additionally, the FiT score for the ensemble-model attack with 20 models was lower than that of the ensemble-model attack with only three models, which were carefully selected using FiT. These findings suggest that the quality of the selected models is more crucial than the quantity of models for effective ensemble-model attacks.

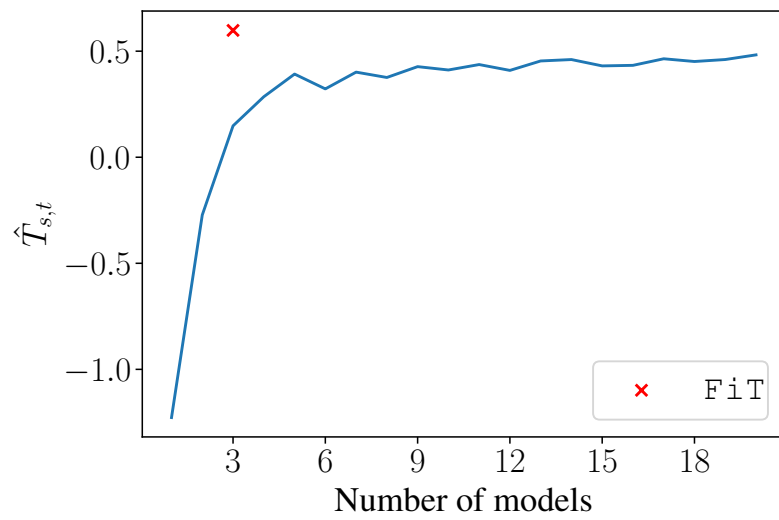


Figure C.4 –  $\hat{T}_{s,t}$  function of the number of models used for ensemble-model to attack  $\text{xCiT}_{\text{nano}}$ . The models are randomly selected and added one by one and compared with `FiT` selecting the three best models for ensemble-model among the 20 models available.







---

**Titre :** La Sécurité des Réseaux de Neurones dans un Contexte Réaliste

**Mot clés :** Réseaux de Neurones, Sécurité, Contexte Réaliste

**Résumé :** L'Intelligence Artificielle est aujourd'hui sur toutes les lèvres, porté par la révolution des réseaux de neurones qui ont fait leurs preuves dans diverses tâches. Ils ont notamment surpassé les capacités humaines en vision par ordinateur. Cette thèse se concentre ainsi sur les réseaux de neurones, en mettant l'accent sur les tâches de classification d'images.

Cependant, ce succès remarquable s'accompagne de certaines failles. Les réseaux de neurones présentent des faiblesses en termes de confidentialité, d'intégrité et de disponibilité de leurs composants. Les données d'entraînement, le modèle et les données d'inférence sont exposés à des attaques. Même dans le scénario réaliste considéré dans cette thèse,

où le modèle fonctionne dans un environnement boîte noire avec des limitations sur le nombre de requêtes, il est possible de voler et de reconstruire le modèle et les données d'entraînement, ainsi que de manipuler les données d'inférence.

Cette thèse met l'accent sur la protection de la confidentialité du modèle compromise l'extraction de modèle et l'extraction de paramètres. Elle explore également le domaine des exemples adverses. Ces perturbations soigneusement conçues entraînent des erreurs de classification, menaçant l'intégrité du modèle à l'inférence. Par conséquent, une partie de cette thèse est dédiée à l'exploration de leurs origines, de leur création et des stratégies de défense contre eux.

---

**Title:** Neural Networks Security under Realistic Scenario

**Keywords:** Deep Neural Networks, Security, Realistic Scenario

**Abstract:** Artificial Intelligence is a hot topic today, driven by the revolution of neural networks that have shown impressive performances across various tasks. Notably, in Computer Vision, they have even outperformed humans. This thesis centers on neural networks applied to image classification tasks.

Yet, this remarkable success is not without its vulnerabilities. Neural networks exhibit weaknesses in terms of confidentiality, integrity, and availability of their components. The training data, the model, and the inference data, are susceptible to potential attacks. Even in the realistic scenario considered in this thesis where the model operates in a black-box setup with limitations on the number

of queries, it remains possible for an attacker to steal and reconstruct the model and training data, as well as manipulate inference data.

This thesis places a particular emphasis on safeguarding the confidentiality of the model, which can be compromised through techniques such as model extraction and parameter extraction. Additionally, it delves into the realm of adversarial examples, which pose threats to the integrity of model inference. The deliberate introduction of small, well-crafted perturbations can result in misclassifications. Consequently, a significant portion of this thesis is dedicated to exploring the origins of adversarial examples, their creation, and strategies for defending against them.