Università degli Studi di Padova

DEPARTMENT OF GENERAL PSYCHOLOGY
Ph.D. School in Brain, Mind and Computer Science
Curriculum Computer Science for Societal Challenges and Innovation

Trustworthy Machine Learning for Anomaly Detection and Computer Vision Applications

Supervisor
Prof. Gian Antonio Susto

Co-Supervisor & Coordinator
Prof.ssa Anna Spagnolli

Ph.D. candidate
Mattia Carletti

XXXV cycle

Università degli Studi di Padova

DEPARTMENT OF GENERAL PSYCHOLOGY
Ph.D. School in Brain, Mind and Computer Science
Curriculum Computer Science for Societal Challenges and Innovation

Trustworthy Machine Learning for Anomaly Detection and Computer Vision Applications

Supervisor
Prof. Gian Antonio Susto

Co-Supervisor & Coordinator
Prof.ssa Anna Spagnolli

Ph.D. candidate
Mattia Carletti

XXXV cycle

Università degli Studi di Padova:
www.unipd.it

Ph.D. School in Brain, Mind and Computer Science:
hit.psy.unipd.it/BMCS

Acknowledgements

First of all, I would like to express my gratitude to Prof. Gian Antonio Susto for his enthusiasm in guiding me through this exciting experience.

I would also like to mention Prof. Giorgio Quer, who gave me the opportunity to live a wonderful experience in San Diego.

I am also grateful to my family and my friends, my pillar of strength.

I would like to express a special thanks to Matteo, with whom I spent a great deal of time talking about research and life.

A special mention goes to my girlfriend Ortensia for her endless patience and support. She is now probably more expert than me in Machine Learning.

Padova, 9 January 2023 M. C.

Abstract

Technologies based on Artificial Intelligence (AI) have gained tremendous popularity in the past few years. This is made possible by the fact that Machine Learning models can now achieve amazing performance, even outperforming humans in several application domains. Unfortunately, there is still one fundamental aspect in which humans succeed and machines fail miserably: trustworthiness. Countless cases of unintended harm caused by AI-enabled technologies have been reported and have attracted wide media coverage. While unintentional, the consequences of these events could be devastating and affect our quality of life. Even more so if we consider that AI is being increasingly adopted in sensitive domains to support high-stakes decisions. In light of these observations, the need for Trustworthy AI is particularly pressing.

In this thesis, we profoundly investigate the properties of popular models that have been used for years in academia and industry and provide tools to improve their trustworthiness. The focus is on two specific dimensions in the space of Trustworthy AI, i.e., interpretability and robustness, and on two application domains of great practical interest, i.e., Anomaly Detection and Computer Vision. In the context of Anomaly Detection, we introduce novel model-specific methods to interpret the Isolation Forest, a popular model in this field, at both the global and local scales. In Computer Vision, we address the problem of robust image classification with Convolutional Neural Networks. We first unveil unknown properties of adversarially-trained models, elucidating inner mechanisms through which robustness against adversarial examples may be enforced by Adversarial Training. We also showcase failure modes related to the simplicity biases induced by Adversarial Training that may be harmful when robust models are deployed in the wild. Finally, we design a novel filtering procedure aimed at removing textures while preserving the image's semantic content. Such filtering procedure is then exploited to design a defense against adversarial attacks.

Keywords: Anomaly Detection, Computer Vision, Deep Learning, Interpretability, Robustness, Trustworthy Machine Learning

Sommario

Le tecnologie basate sull'Intelligenza Artificiale (IA) hanno guadagnato un'enorme popolarità negli ultimi anni. Ciò è reso possibile dal fatto che i modelli di Machine Learning possono ora raggiungere prestazioni sorprendenti, superando persino gli umani in diverse applicazioni. Sfortunatamente, c'è ancora un aspetto fondamentale in cui gli umani hanno successo e le macchine falliscono miseramente: l'affidabilità. Sono stati segnalati innumerevoli casi di danni non intenzionali causati da tecnologie abilitate dall'IA che hanno attirato un'ampia copertura mediatica. Sebbene non intenzionali, le conseguenze di questi eventi potrebbero essere devastanti e influire sulla qualità della nostra vita. A maggior ragione se si considera che l'IA viene sempre più usata in domini delicati per supportare decisioni ad alto rischio. Alla luce di queste osservazioni, la necessità di un'IA affidabile è particolarmente urgente.

In questa tesi, vengono analizzate in profondità le proprietà di modelli popolari che sono stati utilizzati per anni nel mondo accademico e industriale e vengono forniti strumenti per migliorarne l'affidabilità. L'attenzione è rivolta a due dimensioni specifiche nell'ambito del Trustworthy AI, ovvero interpretabilità e robustezza, e a due domini applicativi di grande interesse pratico, ovvero Anomaly Detection e Computer Vision. Nel contesto dell'Anomaly Detection, vengono introdotti nuovi metodi 'model-specific' per interpretare la Isolation Forest, un modello popolare in questo ambito, sia su scala globale che locale. In Computer Vision, viene affrontato il problema della classificazione robusta delle immagini con reti neurali convoluzionali. Per prima cosa vengono svelate proprietà sconosciute dei modelli allenati con il paradigma Adversarial Training, chiarendo i meccanismi interni attraverso i quali la robustezza contro gli adversarial examples può essere indotta da Adversarial Training. Vengono messe in luce anche modalità di errore relative ai simplicity bias indotti da Adversarial Training, che possono rivelarsi dannose quando i modelli robusti vengono utilizzati nel mondo reale. Infine, viene proposta una nuova procedura di filtraggio volta a rimuovere le texture e al tempo stesso preservare il contenuto semantico dell'immagine. Tale procedura di filtraggio viene quindi sfruttata per progettare una difesa contro gli adversarial examples.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Trustworthy AI

In the last decade, amazing progress has been made in challenging tasks in Machine Learning (ML), Deep Learning (DL), and Artificial Intelligence (AI), ranging from Speech Recognition [Han et al., 2020, Baevski et al., 2020, Shi et al., 2021], and Natural Language Processing [Brown et al., 2020, Devlin et al., 2018] to Computer Vision [Vaswani et al., 2017, Yu et al., 2022], and Time Series Forecasting [Oreshkin et al., 2019, Lim et al., 2021]. Researchers and practitioners have been pushing the boundaries of predictive performance to the point that many ML models completed the shift from research laboratories to the real world and started being deployed in the wild, spanning basically every aspect of our lives. For example, virtual assistants like Apple's Siri, Microsoft's Cortana, and Amazon's Alexa are now embedded in smartphones and other devices that we use daily. Waymo's self-driving taxis are being tested across the roads of California. Netflix's recommender engine drives the choice of the movie we will watch after dinner. Fitbit devices can help us spot signs of Atrial Fibrillation so that we can monitor the health status of our heart. The list could continue endlessly and we would find out that our relationship with AI-enabled technologies is tighter than one would imagine. Not only ML affects our everyday routine, but it is also increasingly being used to tackle long-standing problems in science. AlphaFold [Jumper et al., 2021] is boosting research in biology and medicine, e.g., by supporting the design of vaccines to prevent malaria [Ko et al., 2022] or by accelerating the engineering of enzymes to fight plastic pollution [DeepMind,]. AlphaTensor [Fawzi et al., 2022] discovered novel and more efficient algorithms for matrix multiplication. GraphCast [Lam et al., 2022], an ML model based on Graph Neural Networks for weather forecasting, outperforms the most accurate deterministic medium-range weather forecasting system in the world. In light of this pervasive presence of AI algorithms, it is clear that the reliability of their predictions and the means by which interaction with human beings is implemented are aspects of the utmost importance to ensure safe and ethical

use of AI. Although it is obvious that these technologies should undergo socio-political deliberation and consensus before deployment [Floridi, 2019], along with extensive testing routines to prevent unexpected behaviors, this was not standard practice, and many AI-based products have been prematurely launched on the market. Some ignominious failures attracted great attention from the media and we learned the lesson the hard way, i.e., by directly witnessing technologies we were overexcited about failing miserably. On March 18th, 2018 a woman in Tempe (Arizona) died after being hit by an Uber self-driving car while jaywalking with her bicycle [Wakabayashi,]. IBM's Watson, a supercomputer designed to provide cancer treatment recommendations, suggested a fake patient with severe bleeding be given a drug that could cause the bleeding to worsen [Chen,]. OpenAI's large language model GPT-3 suggested a fake patient with suicidal thoughts to kill himself [Quach,]. Amazon's Alexa, after being asked to propose a challenge, told a child to touch an electrical plug with a penny [Shead,]. These are only a few examples of how harmful AI can turn in the wild - although not deliberately - if extensive analysis and adequate stress tests are not included in the design phase.

The (unintended) untrustworthy sides of AI that emerged in the last few years pose new challenges for all stakeholders, which lead to new research questions for AI scientists, the need for new regulations for lawmakers, and the redefinition of AI-related operations for industrial players. Most importantly, as Trustworthy AI [Kaur et al., 2022,Floridi, 2019,Liu et al., 2022] is a strongly multidisciplinary field, academia, industry, and governments are called to cooperate closely to carry out a joint research agenda aimed at building the new generation of Trustworthy AI technologies. The promise to create AI operating in the service of humanity with no harm and in a way that we can fully trust is exciting, but it does not come without obstacles, especially when bringing together people with extremely different backgrounds. Establishing a common ground for effective communication is not an easy task and requires ad-hoc training to make non-technical professionals aware of what can and cannot be achieved with currently available technologies. Domain experts, on the other hand, should encompass ethical principles in their workflow and figure out how to translate them into technical characteristics of the algorithms to be developed. Moreover, as often happens for new areas of research that have no clear and unique definitions of the core concepts, there are currently more than 70 frameworks and lists of principles about the ethics of AI [Floridi, 2019]. This generates confusion among stakeholders and offers the possibility of cherry-picking the kind of ethics that is best retrofitted to justify their behaviors, rather than revising their behaviors to make them consistent with a socially accepted ethical framework [Floridi, 2019].

The path toward Trustworthy AI will not be smooth and will require much more effort compared to the standard design paradigms that are only concerned with the improvement of predictive performance. But if we fail at it, we run the risk of deploying unreliable and flawed models that could be misused and harm users, breaking the promises of a safe and trusted AI designed for the benefit

of the whole mankind. In the long term, being people aware of the flaws and limitations of AI, these technologies might be abandoned, and soon all that will be left will be disquisitions about their unfulfilled potential.

1.2 The Dimensions of Trustworthy AI

In this Section, we follow [Liu et al., 2022] and briefly overview six main dimensions along which the principles of Trustworthy AI are usually implemented: robustness, interpretability, fairness, accountability, privacy, and environmental well-being. There exists some overlap between concepts defined under different dimensions and the landscape of Trustworthy AI is way richer than the picture given below. Nevertheless, here we focus on the most prominent and established perspectives on the field, leaving other more recently introduced dimensions out of the discussion. This allows us to substantiate the analysis with references to a solid body of research works that is missing for dimensions with limited literature (such as human agency and creditability). The main focus is on the robustness and interpretability dimensions, which will be central concepts in the methods and analyses described in Chapters 3, 5 and 6.

1.2.1 Robustness

The robustness of ML models can be broadly defined as the ability to provide high accuracy under perturbations of the data that do not compromise their semantics. In other words, a trustworthy ML model should not exhibit unstable behaviors when irrelevant characteristics of the input data are slightly altered. At the present moment, this ability represents a substantial difference between AI and human beings: while the former struggles to maintain satisfactory performance in noisy settings, the latters are extremely effective and efficient at this task. In the robustness literature, it is usually assumed the existence of a malign attacker that is willing to undermine the performance of an ML model. Depending on whether the attack takes place during training or at test time, we distinguish between

- **Poisoning attacks:** the attacker has access to the training set and designs fake training samples to impair the training process globally [Biggio et al., 2012] or to get bad predictions for specific subgroups of data [Zügner et al., 2018]. Notice that, in practice, poisoning attacks can occur when training samples are generated by the users themselves, e.g., for training recommender systems. In this case, poisoning the training set simply means altering the online behavior with malicious intent. *Backdoor attacks* [Gao et al., 2020, Liu et al., 2020] are a special case of poisoning attacks, where the attacker adds a trigger to specific training samples, associated with a target wrong label. This attack biases the training process and

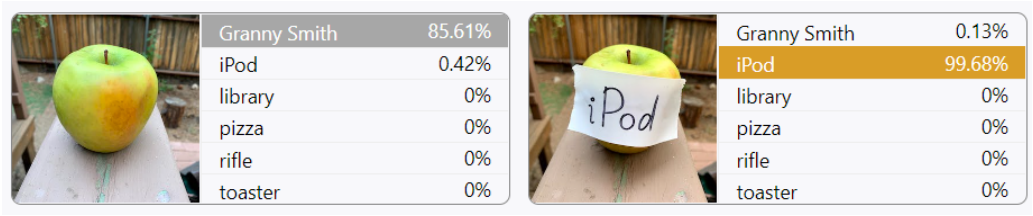

Figure 1.1: Typographic attack against CLIP with text patch (image from [Goh et al., 2021]).

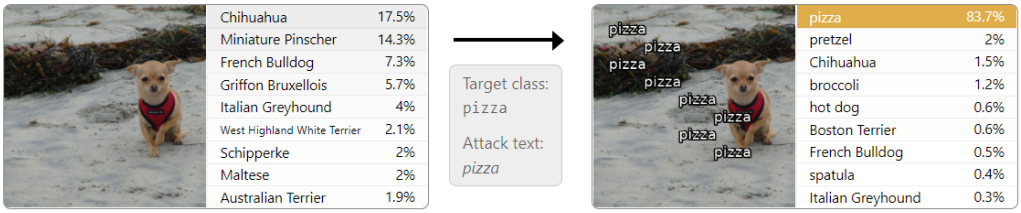

Figure 1.2: Typographic attack against CLIP with plain text (image from [Goh et al., 2021]).

induces the model to leverage the trigger shortcut. At test time, all the samples where the trigger is present will be assigned the target label.

- **Evasion attacks:** the attacker does not have access to the training set, and the model has already been trained. The objective of evasion attacks is to craft fake test samples that fool the trained model. If the attacker has also access to the structure of the trained model (e.g., to the architecture, weights, and gradients of a DL model) and leverages that information, the attack is called *white-box*. *Black-box* attacks, instead, no information about the model is available.

Another interesting type of attack against OpenAI's multi-modal vision and language model Contrastive Language-Image Pre-training (CLIP) has been recently discovered [Goh et al., 2021]. The attack, dubbed *typographic attack*, leverages the multi-modal nature of representations learned by CLIP. Specifically, the authors discovered neurons in CLIP that respond to the same concept, independently of the modality in which it is represented (e.g., image or text). The attack consists of superimposing text patches or plain text over the original image, where the content of the text is deliberately unrelated to the object represented in the image. Some examples are given in Figures 1.1 and 1.2.

It is worth mentioning that the robustness of ML models can be questioned also in cases where there is no explicit adversary. Indeed, distribution shifts arising from the natural diversity of real data can completely impair the performance of ML models, including those trained with ad-hoc countermeasures to prevent harm from other types of attacks [Taori et al., 2020].

In Chapters 5 and 6 we will deal with a specific class of white-box evasion attacks, i.e., ℓ_p-bounded attacks. The necessary background on ℓ_p-bounded attacks will be overviewed in Chapter 4.

1.2.2 Interpretability

Most of the advances in AI in terms of predictive performance have come at the price of reduced interpretability[1] of the models' inner workings. In other words, the best performing models are usually considered as *black-boxes*, being the underlying logic governing their behavior hardly understandable by humans. This does not necessarily represent a problem per se, indeed in many practical use cases, interpretability is not required. For example, in finance, one would surely prefer an extremely complex DL model over a fully interpretable linear model if the former guarantees 10× profits compared to the latter. The crucial point here is that the model can be tested in the wild with no risk of harm to people. This is simply not possible in many other domains, such as in healthcare, where a flawed model could put at risk human lives. In this case, we could only leverage historical data to test the generalization capabilities of a trained model, but we would not have the possibility to simulate every possible scenario that could occur in the real world. For instance, we could not anticipate the model behavior under distribution shifts [Quinonero-Candela et al., 2008]. This is an emblematic example where interpretability could be useful: if the model is equipped with some interpretability traits describing the logic behind its predictions, a clinician would have more information to decide whether or not to trust the model's outcomes. More generally, the need for interpretability stems from an incompleteness in the problem formalization [Doshi-Velez and Kim, 2017]. This means that ML models are optimized so as to minimize the prediction error, but in real-world applications, we usually seek additional feats that cannot be explicitly optimized during training. For this reason, eXplainable Artificial Intelligence (XAI) [Arrieta et al., 2020] has become a driving subfield for the advance of AI research. The general goal of XAI is to shed light on the inner workings of ML and DL models so that developers and users can easily inspect the models' inner structure and check whether other desiderata (such as fairness, robustness, etc.) have been successfully enforced.

With respect to the scope of interpretability methods, *local* methods provide explanations associated with individual predictions, while *global* methods explain the model as a whole. Depending on whether or not the peculiar structure of the ML model to be analyzed is exploited to produce explanations, we further distinguish between *model-specific* and *model-agnostic* methods, respectively. Model-specific methods are characterized by high *translucency* [Molnar, 2022], i.e., they heavily rely on the inherent structure of the specific ML model under

[1] Throughout this thesis, we will use the terms *interpretability* and *explainability* interchangeably. Some works highlighted subtle differences between the two [Gilpin et al., 2018, Rudin, 2019], that can, however, be overlooked for the purposes of this thesis.

examination, while model-agnostic methods earned remarkable interest due to their high *portability*, i.e., they can be applied to a wide range of models.

Model-specific methods A major focus in the XAI field is put on Deep Neural Networks (DNNs) [Che et al., 2016], and ensemble methods [Tolomei et al., 2017], two emblematic examples of algorithms classes that provide models that are highly accurate but hard to be understood by humans. As regards ensemble methods, we mainly relate our work to Random Forests (RFs) [Breiman, 2001], but several works on the interpretation of other ensembles (such as Gradient Boosting Decision Trees) can be found in literature [Valdes et al., 2016, Wang et al., 2018b]. RFs are ensembles of classification or regression trees leveraging *bagging* to reduce the variance of predictions. Compared to single Decision Trees, RFs significantly improve performance in terms of accuracy at the price of reduced interpretability. In this context, many works address the problem of improving standard feature importance score methods. Relevant examples are [Strobl et al., 2008], which proposes an improvement of the permutation importance measure based on a conditional permutation scheme, and [Li et al., 2019], in which the authors introduce a variant of the Mean Decrease Impurity (MDI) feature importance measure aimed at overcoming the problem of MDI feature selection bias. Besides single-feature importance measures, it is worth mentioning some recent works focused on the detection of interactions between features [Basu et al., 2018, Lundberg et al., 2018, Devlin et al., 2019]. Given the fact that DNNs achieve state-of-the-art performance on several complex tasks, it comes as no surprise that a considerable volume of research in the XAI field focused on the problem of DNNs interpretability. The latter can be tackled with the purpose of either providing explanations about the predictions (i.e., the outputs) produced by the model [Zeiler and Fergus, 2014, Simonyan et al., 2013, Murdoch et al., 2018], e.g., in the form of saliency maps as shown in Figure 1.3, or interpreting the internal representations of the processed data [Sharif Razavian et al., 2014, Bau et al., 2017]. It is worth highlighting a third promising line of research aimed at designing inherently interpretable DNNs [Melis and Jaakkola, 2018, Li et al., 2018, Oreshkin et al., 2019].

Model-agnostic methods Among the most prominent model-agnostic techniques used to explain individual predictions, popular approaches are Local Interpretable Model-agnostic Explanations (LIME) [Ribeiro et al., 2016], Anchors [Ribeiro et al., 2018] and SHapley Additive exPlanations (SHAP) [Lundberg and Lee, 2017]. Instead, Partial Dependence Plots [Friedman, 2001] and Accumulated Local Effects plots [Apley, 2016] represent examples of model-agnostic methods used to explain the model's behavior at a global level. While on one hand, high portability may appear as an attractive feature for interpretability methods, on the other hand, the interpretability problem is usually dealt with only once a specific model type has been chosen and the usefulness of

Figure 1.3: Saliency maps for the top-1 predicted class from ImageNet test samples (image from [Simonyan et al., 2013]).

model-agnostic methods simply lies in the lack of model-specific alternatives for several ML models classes. Moreover, model-agnostic methods usually exhibit not negligible shortcomings:

- Since the inner structure of the model being examined is not exploited, the user might suspect that the provided explanation is just a simplistic and coarse approximation of the true underlying relationship between the input and the output.
- The majority of model-agnostic methods are based on the manipulation of inputs and evaluation of the effects said manipulations induce on the corresponding predictions. This represents a delicate process as the artificially created input instances might not belong to the original data manifold, potentially causing stability issues and raising doubts about the actual information conveyed by the interpretability method.
- In light of the need for further restrictive assumptions and/or opaque methodological choices (e.g., independence between features, the creation of perturbed input instances), the user is asked to take a leap of faith and consider the method as reasonable while not fully understanding the theoretical underpinnings. This shifts the problem from the lack of trust in the model to the lack of trust in the interpretability method itself.

Exhaustive descriptions, analyses, and examples of model-specific and model-agnostic approaches can be found in [Guidotti et al., 2019, Molnar, 2022].

1.2.3 Fairness

Algorithms based on ML are increasingly being used in sensitive applications where their outcome may influence high-stake decisions, e.g., to determine whether a loan should be granted or not [Purificato et al., 2022], in justice to assess

the risk of recidivism [Hillman, 2019], or for organizations to screen job candidates [van Esch et al., 2021]. It is clear that any of the decisions mentioned above could affect the quality of someone's life for quite a long time. Unfortunately, countless episodes of discriminatory behaviors in similar scenarios have been reported. For example, the Correctional Offender Management Profiling for Alternative Sanctions (COMPAS) software, used to support decisions in U.S. courts by assessing the risk of recidivism, was found to be biased against African-American[2]. Decision support systems used for loan applications management leverage protected attributes such as race [Hardt et al., 2016]. An algorithm designed to promote job opportunities in the Science, Technology, Engineering, and Math (STEM) fields favored males over women for the visualization of advertisements, supporting a known gender stereotype regarding STEM careers [Lambrecht and Tucker, 2019]. Some products based on AI have been retired from the market because of improper behaviors, like Microsoft's Tay chatbot experiment, which was shut down only 24 hours after deployment since it was generating tweets with racist, sexist, and anti-Semitic language [Wolf et al., 2017]. The major problem with regard to fairness in ML is represented by the fact that ML models are trained on data that are not free of biases, especially when the data are generated by humans (e.g., tweets, reviews of products, news). As a result, without specific countermeasures to control the data collection and training processes, we would end up with models that not only inherit but also amplify undesired biases. To avoid this, a growing body of research works is currently addressing these issues, with the objective of formalizing novel practices in the engineering of AI systems to account for fairness principles by design. Since the fairness dimension is not the main focus of this thesis, we refer the curious reader to [Mehrabi et al., 2021, Verma and Rubin, 2018].

1.2.4 Privacy

Trustworthy AI systems should guarantee the safety of private information encoded in the data exploited in the training process and avoid the risk of privacy leakage. In the past few years, researchers have explored the extent to which sensitive and private information can be extracted from trained models and potential countermeasures to alleviate these issues, initiating a novel strand of research dubbed *Privacy-preserving Machine Learning*. If we assume to have access to the outcomes of a trained ML model, a great deal of information can be retrieved even though training data are no longer available. *Membership inference attacks* can infer whether or not a specific data point was exploited to train the model [Shokri et al., 2017]. This can be done even without access to the model parameters. The model is fed with fake data points whose features are tuned until the outputs produced by the model have very high confidence. In this way, the attacker can reconstruct data points that are, likely, similar to the original

[2] https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing

training samples and with them can train an auxiliary model to predict whether a data point was used or not to train the target model. Similar techniques can be leveraged to craft *model inversion attacks* to retrieve information about the input data, which may include sensitive and private information, from the output of the model [Fredrikson et al., 2015] or from gradient information [Zhu et al., 2019]. Another malicious attack that can reveal basically everything about a trained model is represented by the so-called *model extraction attack* [Tramèr et al., 2016]. This consists in mimicking the input-output behavior of a target model from queries so that a substitute model can be trained and used to infer sensitive information. To minimize the effect of the abovementioned attacks, a number of privacy-preserving techniques have been proposed. They can be partitioned into three main categories: confidential computing, federated learning, and differential privacy. For more details on privacy-preserving techniques see [Al-Rubaie and Chang, 2019, Boulemtafes et al., 2020].

1.2.5 Accountability

The concept of accountability is broad and extends beyond the context of ML models. According to Bovens, *'a relationship qualifies as a case of accountability when there is a relationship between an actor and a forum in which the actor is obliged to explain and justify his conduct, the forum can pose questions, pass judgement, and the actor may face consequences'* [Bovens, 2007]. Starting from Bovens' notion of accountability, in [Wieringa, 2020] the authors specialize the concept to the case where the request for explanation and justification regards a general algorithmic system - of which an AI system is a particular case. In this scenario, accountability can be viewed as the problem of making clear the responsibility distribution amongst all the parties involved in the functioning of the AI system. We can distinguish between *system designers* who design the system so that it meets the users requirements, *system deployers* who are in charge of the proper deployment of the system, *decision makers* who decide whether or not the system should be adopted and the characteristics it should have to comply with regulations and meet performance specifications, *end users* who directly interact with the system, and *system auditors* who are responsible for assessments of the system. Accountability is intimately connected to the interpretability of the AI system [Kim and Doshi-Velez, 2021]: interpretable models can provide explanations associated with their predictions and their inner logic is less opaque compared to that of black-box models. This helps making sense of the model's behavior, so as to ease the allocation of responsibility when an AI system does not work as expected.

1.2.6 Environmental Well-being

The new trend in AI is to train general-purpose models with billions of parameters that can be then adapted to solve multiple downstream tasks. These

models are called *foundation models* [Bommasani et al., 2021] and are trained on huge amounts of training samples, with sophisticated training routines that can last months on hundreds of Graphics Processing Units. Examples include BERT [Devlin et al., 2018], GPT-3 [Brown et al., 2020], and CLIP [Radford et al., 2021], but the list is set to grow in the years to come as companies and organizations are literally racing against each other to push the limits of AI. So far, most of the work on foundation models focused on Large Language Models, but it is reasonable to expect the design of foundation models for other domains, such as computer vision and healthcare. One of the main drawbacks of foundation models is the huge carbon footprint associated with the training and fine-tuning processes, along with inference elaborations. Even worse, since the issues discussed above, such as the lack of robustness or the presence of discriminatory biases, could affect the reliability of foundation models, it might be necessary to retrain them from scratch incorporating novel countermeasures to improve their trustworthiness. Model compression techniques could help reduce the energy consumption of AI algorithms. These include knowledge distillation [Gou et al., 2021], pruning [Anwar et al., 2017, He et al., 2017], and quantization [Han et al., 2015, Cai et al., 2017]. Another promising line of research investigates novel hardware designs that explicitly optimize training and inference energy efficiency [Chen et al., 2020c].

1.3 Thesis Outline

This thesis aims to profoundly investigate the properties of popular models that have been used for years in academia and industry, and provide tools to improve their trustworthiness. This is done along two dimensions in the space of Trustworthy AI: interpretability and robustness. Instead of focusing on the design of brand-new architectures or learning algorithms that deliver an infinitesimal improvement in accuracy, we take a step back and analyze existing models widely adopted in the field of Anomaly Detection and Computer Vision.

On one hand, we recognize the value of proposing new approaches to solve challenging problems, so as to minimize the risk of allocating all the efforts to a research direction that might prove unprofitable. On the other hand, we believe that thorough analyses of what has been done so far by generations of scholars are necessary to get a complete picture of a specific problem before moving forward with the design of novel strategies to solve it. For example, Vision Transformers (ViT) [Vaswani et al., 2017] achieve amazing performance on image classification tasks, and nowadays, the vast majority of papers in top AI conferences revolve around ViT, but we still do not know enough about Convolutional Neural Networks (CNNs) [Gu et al., 2018] to consign them to oblivion. Moreover, CNN is still the best model architecture for robustness against natural and system noises [Tang et al., 2021]. From a practical perspective, when a trained model has been deployed, and potentially integrated into a complex pipeline, the

improvement derived from retraining or the replacement with some brand-new state-of-the-art model is always weighted by the costs associated with these operations. As a result of this trade-off, it is often more convenient to equip less recent models with novel feats post-hoc, instead of re-running the training and deployment operations from scratch.

The thesis is outlined as follows. In Part I, we address the lack of interpretability in Anomaly Detection problems and provide novel tools to equip a popular Anomaly Detection model with interpretability traits. More specifically:

- In Chapter 2, we describe the Anomaly Detection task, discuss the related challenges and quickly overview traditional models. We also review the concepts at the core of the Isolation Forest algorithm.

- In Chapter 3, we introduce novel model-specific methods to explain the Isolation Forest model at both global and local scales. Specifically, we introduce i) a global interpretability method, dubbed *Depth-based Isolation Forest Feature Importance (DIFFI)*, to provide global feature importance scores which represent a condensed measure describing the macro-behavior of the Isolation Forest model on training data; ii) a local version of the DIFFI method, called *Local-DIFFI*, to provide local feature importance scores aimed at interpreting individual predictions made by the Isolation Forest model at test time; iii) a simple and effective procedure to perform unsupervised feature selection for Anomaly Detection problems based on the DIFFI interpretability method. We also discuss an industrial case study in the field of semiconductor manufacturing. The work presented in this Chapter is based on articles [J1] and [C1].

The Part II of the thesis is devoted to the interpretability and robustness of Convolutional Neural Networks for image classification problems. More specifically:

- In Chapter 4, we revise basic notions on DL models, with a special focus on CNNs for image classification and Adversarial Training, a learning paradigm that enforces robustness against adversarial examples.

- In Chapter 5, we investigate the Adversarial Training paradigm for robust image classification and analyze the structural and functional properties of adversarially-trained models. The analysis can be broadly construed as a step toward increasing the interpretability of adversarially-trained models. Indeed, our findings shed light on their inner workings, elucidating mechanisms through which robustness may be enforced. Specifically, we show that: i) adversarially-trained CNNs have a greater number of spatially dense feature maps than natural models, reducing model expressivity; ii) feature maps of adversarially-trained CNNs are more redundant, reducing the effective number of active channels in hidden layers; iii) the latent space of adversarially-trained CNNs offers representations with varying

levels of robust and natural accuracy, demonstrating (contrary to common belief) that latent features of robust models are not inherently robust; iv) adversarially-trained models are trivially biased towards color, potentially leading to undesired behaviors under simple content-preserving color perturbations that do not compromise other robust features such as shape. Taken together, our findings show that adversarially-trained models do not exploit efficiently the model capacity and that the simplicity biases induced by Adversarial Training, while useful to improve robustness to adversarial examples, might be harmful in other contexts. The work presented in this Chapter is based on article [S1].

- In Chapter 6, we tackle the problem of robustness against adversarial examples and introduce a novel defense based on the manipulation of the input image. Specifically, we introduce an iterative filtering framework, dubbed *Image-Graph Extractor (IGE)*, designed to explicitly remove textures from images while preserving the semantic content. Based on the IGE framework, we define a defense called *Filtering As a Defense (FAD)* that successfully improves the robustness of pre-trained CNNs. Additionally, filtered images produced by IGE can be leveraged as data augmentation during training to improve robustness against common data corruptions. The work presented in this Chapter is based on article [S2].

We conclude by summarizing the contributions of this thesis in Chapter 7, where we also highlight the potential impact the proposed methods could have on future research in the field of Trustworthy AI.

1.4 List of Publications

The work presented in this thesis has appeared in the publications listed below.

Journal Papers

[J1] **Carletti, M.**, Terzi, M., Susto, G. A. (2023), "Interpretable Anomaly Detection with DIFFI: Depth-based feature importance of Isolation Forest." Engineering Applications of Artificial Intelligence, 119, 105730.

Conference Papers

[C1] **Carletti, M.**, Maggipinto, M., Beghi, A., Susto, G. A., Gentner, N., Yang, Y., Kyek, A. (2020, December), "Interpretable anomaly detection for knowledge discovery in semiconductor manufacturing." In 2020 Winter Simulation Conference (WSC) (pp. 1875-1885). IEEE.

Submitted Papers

[S1] **Carletti, M.**, Terzi, M., Susto, G. A., "On the Limitations of Adversarial Training for Robust Image Classification" submitted to Neurocomputing, Elsevier.

[S2] Terzi, M., **Carletti, M.**, Susto, G. A., "Improving Robustness with Image Filtering." submitted to Neurocomputing, Elsevier.

Other Publications

The author has also contributed to the following publications, which are not discussed in this thesis.

[C2] Gentner, N., Kyek, A., Yang, Y., **Carletti, M.**, Susto, G. A. (2020, December), "Enhancing scalability of virtual metrology: a deep learning-based approach for domain adaptation." In 2020 Winter Simulation Conference (WSC) (pp. 1898-1909). IEEE.

[C3] Fabris, A., Mishler, A., Gottardi, S., **Carletti, M.**, Daicampi, M., Susto, G. A., Silvello, G. (2021, July), "Algorithmic Audit of Italian Car Insurance: Evidence of Unfairness in Access and Pricing." In Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society (pp. 458-468).

[J2] Gentner, N., **Carletti, M.**, Kyek, A., Susto, G. A., Yang, Y. (2021), "DBAM: Making virtual metrology/soft sensing with time series data scalable through deep learning." Control Engineering Practice, 116, 104914.

[J3] Mancin, L., Rollo, I., Mota, J. F., Piccini, F., **Carletti, M.**, Susto, G. A., Valle, G., Paoli, A. (2021). "Optimizing Microbiota Profiles for Athletes." Exercise and Sport Sciences Reviews, 49(1), 42-49.

[J4] Arena, S., Bodrov, Y., **Carletti, M.**, Gentner, N., Maggipinto, M., Yang, Y., Beghi, A., Kyek, A., Susto, G. A. (2021), "Exploiting 2D Coordinates as Bayesian Priors for Deep Learning Defect Classification of SEM Images." IEEE Transactions on Semiconductor Manufacturing, 34(3), 436-439.

[J5] Dandolo, D., Masiero, C., **Carletti, M.**, Dalle Pezze, D., Susto, G. A. (2023), "AcME—Accelerated model-agnostic explanations: Fast whitening of the machine-learning black box." Expert Systems with Applications, 214, 119115.

Part I

Interpretability in Anomaly Detection

Chapter 2

Background on Anomaly Detection

2.1 The Anomaly Detection Task

Anomaly Detection (AD) techniques aim at automatically identifying patterns, within a given collection of data, that do not conform to an expected regular behavior [Chandola et al., 2009]. Such patterns are usually referred to as *anomalies* or *outlier* in the AD literature, but also *discordant observations*, *exceptions*, *aberrations*, *surprises*, *peculiarities* or *contaminants*, depending on the application domain and community. Throughout this thesis, we will use the terms 'outlier' and 'anomaly' interchangeably and refer to Hawkins' definition given in Definition 2.1.1.

Definition 2.1.1. According to Hawkins' definition [Hawkins, 1980], an outlier is an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism.

As stated in [Ben-Gal, 2005], an exact definition of anomaly depends on assumptions on the data structure and on the specific class of AD methods adopted. Nevertheless, Hawkins' definition is general and broad enough to cope with different scenarios in terms of the characteristics of both the data and the AD method. The concept of an anomaly is related but different in nature from another common phenomenon in data analysis, i.e., *noise*. Noisy data are usually the results of measurement or labeling errors during the data collection process, or other sources of interference that worsen the quality of the true signal that is being recorded. As such, noise carries no information and is considered an undesired effect the data analyst wants to get rid of [Chandola et al., 2009]. Anomalies, instead, may be extremely interesting and informative data points to be analyzed, as they may also capture the natural variability within a group of data points sharing some semantically relevant characteristics [Suri et al., 2019]. For example, in healthcare AD models can be leveraged to spot anomalous

physiological readings, leading to prompt response to medical emergencies and deeper knowledge of one's health status [Ukil et al., 2016, Šabić et al., 2021].

The effectiveness of AD algorithms is of paramount importance in a wide array of application domains, ranging from wireless sensor networks [Miao et al., 2018] and industrial cyber-physical systems [Puggini and McLoone, 2018, Yang et al., 2017] to healthcare [Meneghetti et al., 2018], driving systems [Zhang et al., 2017], and biology [Zhang et al., 2020]. Their high usability is mainly due to the fact that most AD algorithms can be trained and deployed in unsupervised settings. This is particularly useful in environments where the data labeling process by human experts is prohibitively expensive and time-consuming, calling for a human-centered design principle that guarantees the minimization of human efforts.

2.1.1 Types of Anomalies

Based on the categorization proposed in [Chandola et al., 2009], we can distinguish between three main types of anomalies:

- **Point anomalies:** an individual data point is different and isolated from the rest of the data. For example, in a dataset where each data point represents an adult person, described by age, gender, and height, a person who is 220 centimeters tall is a point anomaly. This is the simplest and most common type of anomaly in AD problems.

- **Contextual anomalies:** a data point is considered as an anomaly in a specific context, while it is considered a regular data point in other contexts. For example, a high number of sold turkeys can be considered normal on the day before Thanksgiving, while it may be an anomaly on a random day in August. The notion of context can be specified only if so-called *contextual attributes* are available. In the example above, time is the contextual attribute. The number of sold turkeys, instead, is called a *behavorial attribute*. More generally, behavioral attributes are non-contextual characteristics of data points.

- **Group anomalies:** there is no single anomalous data point, but anomalous aggregated behaviors of groups of data points that are related to each other. This type of anomaly is common in sequential data, e.g., time series, where subsequent data points within a time interval may not be anomalies by themselves, but taken together they represent an anomalous behavior. For example, in a human electrocardiogram (ECG) signal, a prolonged sequence of constant low values may represent a group anomaly - in light of the quasi-periodic nature of ECG - while a single data point with a low value does not necessarily represent an anomaly by itself.

2.1.2 Challenges in Anomaly Detection

Although it may seem trivial to tell apart anomalies from regular points, the AD task is challenging and far from being fully solved. This makes it exciting from the perspective of both practitioners and researchers. Indeed, many aspects of AD problems - that stem mostly from the limitations of the data collection process - translate into interesting methodological challenges, including the following [Chandola et al., 2009]:

- In the vast majority of practical applications, AD datasets are not labeled. This means that we do not know which are the anomalous points and unsupervised ML models are the only option to solve the task. Moreover, the lack of labeled data points poses a significant problem in assessing the performance and generalization capabilities of AD models.

- If labeled datasets are available, supervised or semi-supervised ML models (if labels are available only for regular points) can be leveraged. However, the AD task cannot be cast as a binary classification problem due to the substantial class imbalance (anomalies are usually far fewer than regular points). Moreover, the anomalous class would be more heterogeneous than the regular class, and it might not represent all the possible types of anomalies.

- In some applications, it may be hard to collect anomalous data points because they correspond to the occurrence of undesirable events that should be avoided. For example, in the healthcare domain, an anomaly could represent a patient diagnosed with cancer, or, in the manufacturing domain, it may correspond to a mechanical fault causing a stop in the production line. As a result, it may be difficult to increase the number of anomalous data points by collecting more data in the wild.

- As mentioned above, the exact definition of anomaly depends on the specific application. Therefore, a precise formulation of the AD problem usually requires domain knowledge that may be costly or hard to get.

- If anomalies are produced by an adversarial agent with malicious intent, they may be subtle and hard to spot because their characteristics are intentionally made similar to those of regular data instances.

The development of a general and comprehensive framework to solve the AD task is even more challenging, given the fact that the problem could be formalized in extremely different ways depending on the nature of data, availability of labels, and so on. For this reason, it is usually difficult to adapt techniques developed for a specific problem setup to a more general scenario.

2.2 Anomaly Detection Algorithms

The format of the input in AD algorithms depends on the application domain and can be either a vector in the case of tabular data, an image, a sequence of images, or even a graph. Independently of the nature of the raw data, the format of the output is one of the following [Chandola et al., 2009]:

- An *anomaly score*: each data point is associated with a real value that reflects its 'degree of outlierness'. Based on the anomaly scores, one can set a threshold value to discriminate between anomalies and regular points. The threshold is usually tuned by taking into account priors from domain knowledge or time constraints if the output of the AD model should be revised by humans.

- A binary label: the model directly outputs a hard label, but it is usually possible to indirectly control the implicit threshold value by adjusting the hyperparameters of the model.

Anomaly scores provide a richer description of a given dataset as they can be used to define a ranking over anomalies. This could be useful in practice, e.g., to focus the attention of end users on ambiguous predictions so as to optimize the workforce when the AD model is part of a system where interaction with humans plays a key role.

Deep Learning vs traditional methods In recent years, a growing volume of research has been focusing on approaches based on DNNs to tackle the AD task [Pang et al., 2021], especially for applications involving graphs [Ma et al., 2021, Chen et al., 2021, Yuan et al., 2015], and videos [Nayak et al., 2021, Georgescu et al., 2021, Sabokrou et al., 2017]. Despite the high performance, DNNs cannot be considered the ultimate solution to every AD problem as they exhibit a number of drawbacks in several real-world scenarios: i) depending on the complexity of the task and the dimensionality of the data, the training process of a DNN might last many hours or even days; ii) state-of-the-art DNN models are implemented (and trained) on expensive Graphics Processing Units, that might not be affordable in environments/applications characterized by limited budget or resource-constrained devices; iii) typically, a huge number of data points are required for the DNN to get satisfying generalization capabilities. For these reasons, there still persists a countless number of applications where traditional AD techniques are preferred over solutions based on DNNs.

These traditional methods can be categorized into the following main subgroups [Wang et al., 2019, Chandola et al., 2009]:

- Statistical-based methods: these methods are based on the assumption that regular data points occur in high-probability regions of an underlying stochastic model, while anomalies occur in low-probability regions. The

statistical model is learned from the training data, and predictions for unseen data are obtained by means of statistical inference tests to check whether or not they belong to the learned model. Parametric methods assume knowledge about the true underlying distribution that generated the data and the task is to estimate the parameters of the distribution. Non-parametric methods, instead, are distribution-free.

- Distance-based methods: these methods are based on the assumption that anomalies are far away from their neighbor(s). Examples include Index-based algorithms and Nested-loop algorithms[Knorr and Ng, 1998], and methods based on K-nearest neighbors [Ramaswamy et al., 2000].

- Density-based: these methods are based on the assumption that anomalies lie in low-density regions of the feature space. Examples of density-based AD methods are the Local Outlier Factor [Breunig et al., 2000] and the Connectivity-based Outlier Factor [Tang et al., 2002] algorithms.

- Clustering-based: these methods rely on a preliminary clustering of data points. Depending on the algorithm, anomalies could be data points that do not belong to any of the learned clusters, data points that lie far away from the closest cluster centroid, or data points that belong to sparse and small clusters. The Cluster-Based Local Outlier Factor algorithm [He et al., 2003] and the FindOut algorithm [Yu et al., 2002] are examples of cluster-based methods.

- Ensemble-based: these methods are based on the concept of ensemble, i.e., to combine the predictions of multiple weak learners to improve the detection capabilities of the resulting model. The eXtreme Gradient Boosting Outlier Detection algorithm (semi-supervised) [Zhao and Hryniewicki, 2018], and the Isolation Forest [Liu et al., 2008,Liu et al., 2012] are examples of ensemble-based methods.

Many other alternative categorizations of AD models can be found in literature, and there may exist overlap among some of the abovementioned families of algorithms. For example, many cluster-based techniques rely on distance measures. Another promising research direction for AD is represented by recent works based on granular computing [Kiersztyn et al., 2021, Zhou et al., 2022]. In the next Section, we will focus on the Isolation Forest, a seminal algorithm that inspired most of the recent tree-based AD models [Barbariol et al., 2022], and quickly overview the principles behind its functioning.

2.3 Isolation Forest

The Isolation Forest (IF) [Liu et al., 2008, Liu et al., 2012] is an unsupervised AD algorithm leveraging an isolation procedure to infer a measure of outlierness

- the anomaly score - for each data point. The isolation procedure is based on recursive partitioning and aims at defining a region in the data domain where only the data point under examination lies. The underlying mechanism of IF is based on the reasonable hypothesis that the isolation procedure for outliers requires a limited number of iterations, while the isolation of inliers generally needs a larger number of recursive partitions. We will provide a formal description of the IF algorithm in the following.

2.3.1 The Isolation Forest Algorithm

The IF is an ensemble of *Isolation Trees (ITs)* $\{t_1, \ldots, t_T\}$, i.e., base anomaly detectors characterized by a tree-like structure. ITs are data-induced random trees in which each internal node v is associated with a randomly chosen splitting feature (denoted $f_{sp}(v)$) and a randomly chosen splitting threshold (denoted $\tau_{sp}(v)$). Data points associated with node v undergo a split test: points for which the value of $f_{sp}(v)$ is less than $\tau_{sp}(v)$ are sent to the left child of v, the others to the right child.

Given a dataset $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ of p-dimensional data points, each IT t is assigned a subset $\mathcal{D}_t \subset \mathcal{D}$ (usually called *bootstrap sample*) sampled from the original set and carries out an isolation procedure based on the split tests associated to the internal nodes. Bootstrap samples have the same predetermined size, i.e.

$$|\mathcal{D}_t| = \psi \quad \text{for } t = 1, \ldots, T.$$

Data points in $\mathcal{D}_t$ (called *in-bag samples*, from the perspective of tree t) are recursively partitioned until either all points are isolated or the IT reaches a predetermined depth limit $h_{max} = \lceil \log_2(\psi) \rceil$, function of the bootstrap samples size ψ. As a result, each data point $\mathbf{x}_i$ ends up in a leaf node, denoted $l_t(\mathbf{x}_i)$. We will denote with $h_t(\mathbf{x}_i)$ the number of edges that $\mathbf{x}_i$ passes through in its path from the root node to the corresponding leaf node, which is equivalent to the depth of the leaf node $l_t(\mathbf{x}_i)$.

The procedure described above is iterated over all ITs, each of which is assigned a different bootstrap sample. The anomaly score for a generic data point $\mathbf{x}_i$ is then computed as

$$z(\mathbf{x}_i) = 2^{-\frac{\bar{h}(\mathbf{x}_i)}{c(\psi)}} \tag{2.1}$$

where $c(\psi)$ is a normalization factor given by

$$c(\psi) = \begin{cases} 2H(\psi - 1) - \frac{2(\psi-1)}{\psi} & \text{if } \psi > 2, \\ 1 & \text{if } \psi = 2, \\ 0, & \text{otherwise} \end{cases} \tag{2.2}$$

and $H(k)$ is the harmonic number which can be estimated as $H(k) \approx \ln(k) + 0.5772156649$. $\bar{h}(\mathbf{x}_i)$ is the average path length associated with $\mathbf{x}_i$ and it is computed as

$$\bar{h}(\mathbf{x}_i) = \frac{1}{T} \sum_{t=1}^{T} h_t(\mathbf{x}_i). \tag{2.3}$$

In the last step of the IF algorithm, anomalous data points are flagged through a thresholding operation on the anomaly scores. In this way, it is possible to partition the original set $\mathcal{D}$ as follows:

- the subset of predicted inliers $\mathcal{P}_I = \{\mathbf{x}_i \in \mathcal{D} \,|\, \hat{y}_i = 0\}$,
- the subset of predicted outliers $\mathcal{P}_O = \{\mathbf{x}_i \in \mathcal{D} \,|\, \hat{y}_i = 1\}$,

where $\hat{y}_i \in \{0, 1\}$ is the binary label produced by the thresholding operation, indicating whether the corresponding data point $\mathbf{x}_i$ is anomalous ($\hat{y}_i = 1$) or not ($\hat{y}_i = 0$).

For further details on the IF algorithm and its properties, we refer the reader to the original paper [Liu et al., 2008] and to the extended work [Liu et al., 2012]. To conclude, it is worth highlighting that the IF, as a tree-based ensemble model, shares an inherent structure similar to that of the Random Forest (RF) [Breiman, 2001]. Nonetheless, random choices in IF have a far greater impact since, differently from RF, attributes associated with internal nodes are not selected according to specific splitting criteria but, indeed, randomly. This may be daunting to researchers interested in making the IF interpretable, but, as we shall show in Chapter 3, finding a solution to such a challenge is feasible.

2.3.2 Partitioning in Isolation Trees: A Toy Example

In Figures 2.1 and 2.2 is illustrated a toy example of recursive partitioning performed in a single IT. In this case, the dataset comprises 7 regular data points (blue) and 1 anomaly (red), described by two features f_1, f_2. In Figure 2.1, the isolation procedure for an anomaly x_O (left panel) is compared to the isolation procedure for a regular data point x_I (right panel). While the anomaly can be easily isolated with a single partitioning of the f_1-f_2 plane, the regular data point requires 5 partitioning steps. The same scenario is depicted in Figure 2.2 from the perspective of the IT associated with the partitioning performed in Figure 2.1. Notice that each internal node of the IT is associated with a split test, characterized by a splitting feature f_{sp} and a splitting threshold τ_{sp}. Moreover, each node (either internal or leaf) is associated with a subset of the data points, which is determined by the result of the split test performed by its parent node. The number of partitioning steps required to isolate a specific data point is equivalent to the depth of the leaf node where the data point ends up (the root node is at depth 0). The anomaly x_O is isolated with just one partitioning step, and the associated leaf node is indeed at depth 1. The regular data point x_I,

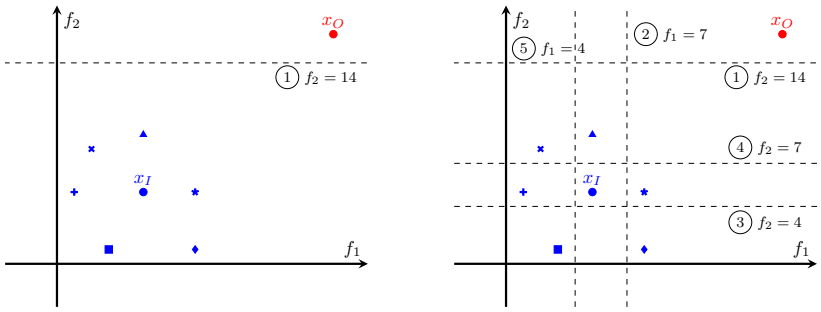

Figure 2.1: Split tests in the $f_1 - f_2$ plane. Isolation of an anomaly (left) and isolation of a regular point (right).

instead, is isolated in 5 partitioning steps, and the depth of the associated leaf node is indeed equal to 5.

In the IF algorithm, this procedure is performed for each IT, and the anomaly scores are obtained by averaging the depths according to Equation (2.1).

2.3.3 Variants of the Isolation Forest model

The widespread use and effectiveness of the IF algorithm have been driving factors for the design of many variants of the original implementation, adapting it to challenging application scenarios and integrating novel methodological principles. The Extended Isolation Forest [Hariri et al., 2019] exploits non-axis-parallel hyperplanes with random slopes for splitting the data. The Functional Isolation Forest [Staerman et al., 2019] extends the use of the original model from finite-dimensional observations to functional data. In k-means-based Isolation Forest [Karczmarek et al., 2020], k-means clustering is used to predict the number of divisions on each decision tree node. Other methods, such as iForestASD [Ding and Fei, 2013], RS-Forest [Wu et al., 2014], AHIForest [Ding et al., 2015], Isolation Mondrian Forest [Ma et al., 2020], adapt the IF for use with streaming data. For analyses and comparisons of other variants of the IF model, we refer the interested reader to [Barbariol et al., 2022].

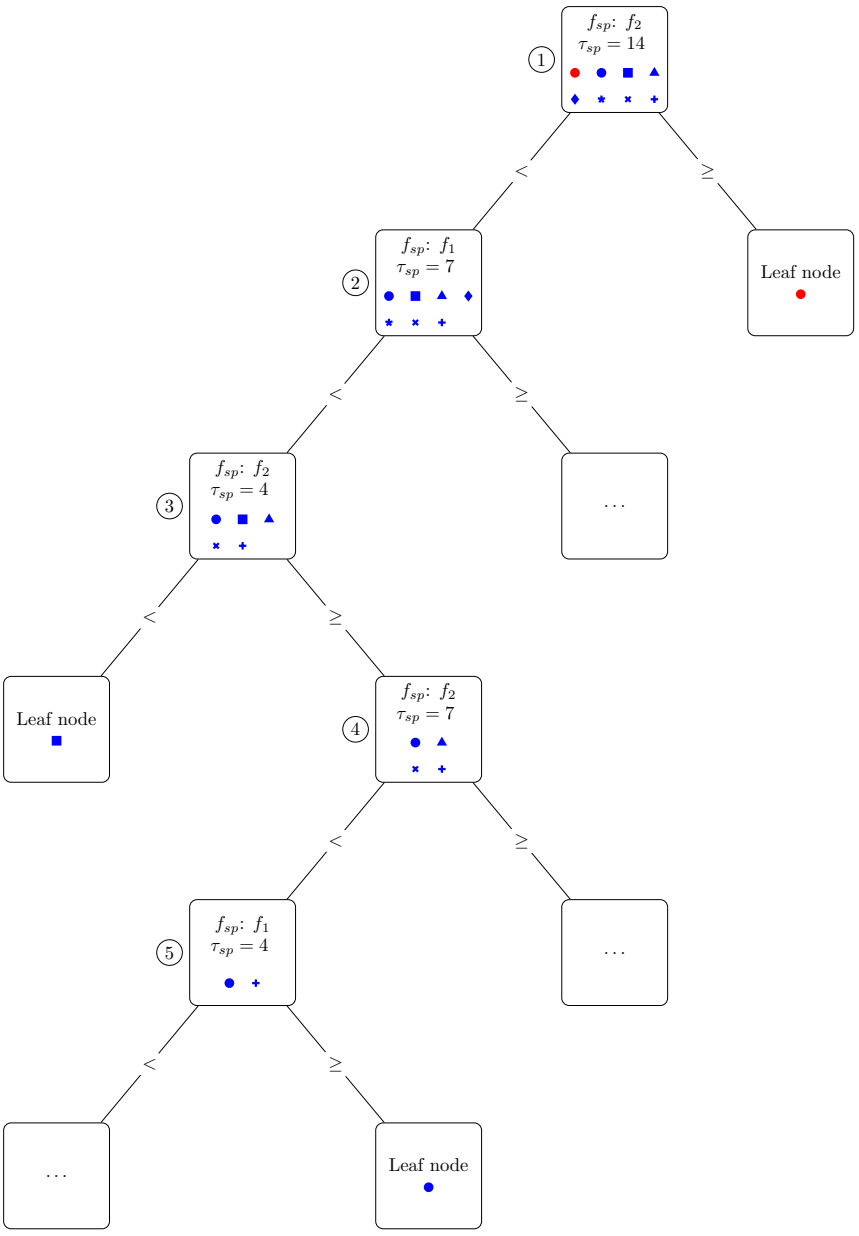

Figure 2.2: Split tests in an Isolation Tree.

Chapter 3

Depth-based Isolation Forest Feature Importance

3.1 Introduction

Although AD algorithms have proved to be extremely useful and effective, their widespread adoption is far from being a reality even in industries and organizations with adequate infrastructures. This is actually a more general problem affecting any technology based on ML, and it is mainly due to two 'soft' factors: (i) lack of confidence/trust from the users in the outcomes of AD algorithms and (ii) no immediate association between outcomes and root causes. The first issue arises from the lack of labeled data points (that, on the other hand, is one of the main reasons why AD algorithms are appealing in the first place), which makes it impossible to set up an adequate testing procedure. This leads either to blindly trusting the algorithm or not using it at all, both cases being undesirable. The second question, instead, investigates the possibility of gaining additional knowledge about the task at hand, which may translate into actionable insights for troubleshooting or root cause analysis. The aforementioned issues can be addressed following the principles of XAI [Gunning and Aha, 2019], whose objective is to make black-box ML models easily understandable by human beings.

In this Chapter, we focus on the interpretation of the Isolation Forest [Liu et al., 2008,Liu et al., 2012], one of the most popular and effective approaches for AD. As discussed in Section 2.3, the IF is inherently governed by randomness, which represents a major challenge for the design of interpretability techniques based on its inner structure. The methods proposed in Section 3.5 provide an effective and computationally inexpensive solution to this problem.

In the remainder of this Chapter, we review the relevant literature in the field of interpretable AD in Section 3.2, we give an overview of the motivations at the core of the proposed methods in Section 3.3 and the main contributions in Section 3.4. Section 3.5 is devoted to the description and analysis of the global

and local DIFFI methods for the interpretation of the IF, along with a novel technique for unsupervised feature selection. In Section 3.6 the experimental results and a discussion thereof are provided, while in Section 3.7 we present the results of a case study in the field of semiconductor manufacturing. Finally, in Section 3.8 we draw conclusions and identify some interesting research directions for future works.

3.2 Related Work

Regression and classification problems (and, consequently, the models commonly used to tackle them) attract the vast majority of research interest in the XAI community. The same cannot be said for AD unless standard ML or DL models are employed: this is typically the case where AD tasks are performed on time-series data through forecasting and monitoring the mismatch between forecast and real data, a formalization typically adopted for univariate AD tasks with highly non-stationary data. In the context of multivariate AD, interpretability is still a challenging topic: in the remainder of this Section, we briefly describe some of the most relevant works facing the problem of interpretability in AD.

In [Macha and Akoglu, 2018], authors aim to explain anomalies in group. The authors introduce a new algorithm called x-PACS (which stands for eXplaining Patterns of Anomalies with Characterizing Subspaces), producing interpretable rules which are also discriminative (i.e., they can separate anomalies from the normal points) and can, thus, be used also to perform AD.

The problems of outlier detection and outlier interpretation are addressed simultaneously also in [Dang et al., 2014]. The proposed algorithm is based on spectral graph embedding theory and can learn an optimal subspace in which an outlier is well discriminated from normal data points while maintaining the intrinsic geometrical structure of the data for high-quality outlier explanations.

In [Tang et al., 2015], a new algorithm for contextual outlier detection on categorical data is proposed. A contextual outlier is here defined as a small group of data points that share strong similarities with a larger reference group of data points on some attributes but deviate considerably on some other attributes. In contextual outlier detection, the goal is to identify the outliers, along with the associated contextual information, which is represented by: i) the corresponding reference group, ii) the attributes defining the abnormal behavior w.r.t the reference group, iii) the population of similar outliers sharing the same context and iv) the outlier degree (the ratio between the cardinality of the reference group and the cardinality of the outlier group).

Differently from the approaches described above, which rely on the design of new AD algorithms, in [Gupta et al., 2018], the authors propose a model-agnostic method to explain outliers either detected by an AD algorithm or dictated, i.e., reported to the analyst externally. They provide compact visual explanations (respecting the limited attention of human analysts) consisting of a set of pairwise

feature plots (which they call *focus-plots*). Among all possible focus-plots, given a fixed budget for explanations, they select the subset of focus-plots that "explain away" the outliers to the largest possible extent. Following a similar logic, also [Liu et al., 2017] leverages traditional AD algorithms for the detection stage. Then, a model-agnostic Contextual Outlier INterpretation (COIN) framework is set up to provide explanations. By exploiting the isolation property of outliers, the overall problem of detector interpretation is decomposed into multiple local classification tasks aimed at explaining individual outliers. In this way, the original task turns into the problem of explaining each outlier w.r.t. its local context.

The approach proposed in Section 3.5, as it will be clear later on, share the same "analyst-centered" approach of [Gupta et al., 2018] and, like in [Gupta et al., 2018, Liu et al., 2017], the focus is not on the design of a new detector. The only difference is in the scope of the proposed interpretability solutions: while the methods described in [Gupta et al., 2018, Liu et al., 2017] cover a wider range of models, the methods proposed in Section 3.5 are specifically tailored to the IF detector.

3.3 Motivations

If we consider the design of the evaluation procedure as part of the problem formalization process, the need for interpretable algorithms in the context of AD is consistent with the connection between the notions of interpretability and incompleteness evidenced in [Doshi-Velez and Kim, 2017] and mentioned in Section 1.2.2. Indeed, due to the lack of labeled datasets in AD problems, AD algorithms are practically rarely testable in unsupervised settings. To fill this gap that may prevent the adoption of such automated systems, we need to provide proxies to assess their trustworthiness, i.e., tools to interpret the model's inner workings and to assess whether it is aligned with the expected behavior.

DIFFI is, to the best of our knowledge, the first model-specific method addressing the need for interpretability for the IF detector. The global DIFFI method is inspired by the preliminary work [Carletti et al., 2019] but differs entirely in the information is supposed to convey. While in [Carletti et al., 2019] the goal is to get additional knowledge on the specific AD problem at hand (which is extremely useful, especially in contexts where no domain expertise is available), in this work, we focus on providing additional information about a trained instance of the IF model, with the main goal of increasing users' trust. Indeed, if the estimated feature importance scores aligned well with human prior knowledge, users would be more prone to lessen the supervision and safely give more autonomy to the machine (at least in non-critical scenarios). In this way, we could promote the adoption of the IF in fields where the professionals' skepticism towards intelligent algorithms is still a major obstacle to massive use.

The model-specific nature of DIFFI is motivated by the will to reflect the

actual logic governing the IF behavior and this may not be feasible with some model-agnostic techniques. For example, when exploiting interpretable surrogate models [Molnar, 2022] trained to approximate the predictions of a black box model, we need to make sure that the surrogate model fits the predictions of the original model with a satisfactory level of accuracy. Such a requirement represents an undesirable source of suspicion. Moreover, as argued in [Molnar, 2022, Lipton, 2018], models commonly considered as universally interpretable, such as decision trees or linear regression, may lose their transparency advantage as they are asked to fit complex relations: very deep decision trees do not offer simple and intuitive visualizations, while linear regression is not suitable to model highly non-linear mappings.

DIFFI is a *post-hoc method*: we decided to preserve the performance of an established and effective AD algorithm and focus on providing global and local feature importance measures computed a posteriori. The design of an intrinsically interpretable model would have required sacrificing some predictive power in light of the trade-off between accuracy and interpretability [Molnar, 2022].

The introduction of a local variant of the original algorithm for the interpretation of individual predictions serves a two-fold objective: i) it enables the interpretation of single data points in online settings when the model has already been deployed, and ii) it helps in enhancing trust as the user can check not only whether the model tends to make mistakes on those kinds of inputs where humans also make mistakes [Lipton, 2018], but also whether the misclassified inputs are being misinterpreted in the same way a human would.

It should be noted that by providing both a global and a local interpretability method, we can guarantee maximum flexibility: based on the required granularity or the amount of time that can be invested in the analysis of the results, the users have the possibility to choose the solution better suited to the specific scenario in which they operate.

The choice to focus on the seminal IF algorithm rather than on any newer variant is motivated by the fact that IF is still the most representative and most used method in the family of tree-based approaches for AD [Barbariol et al., 2022].

3.4 Contributions

Motivated by its ability to attract the attention of a growing and heterogeneous community of researchers and practitioners, we directed our efforts to the interpretation of the IF [Liu et al., 2008, Liu et al., 2012]. The IF model is particularly appreciated and widely used thanks to its high detection performance (often even with default hyperparameters values, with no tuning required) and its computational efficiency. Despite that, just like all ensemble learning methods, it might trigger perplexities and doubts as far as interpretability is concerned:

indeed, no information about the logic behind the mechanism producing the predictions is available, and neither an indication about which are the most relevant features to solve the AD task. In this Chapter, we propose for the first time *model-specific* methods (i.e., methods based on the particular structure of the IF model) to address the mentioned issues. Specifically, we introduce:

- A global interpretability method, called *Depth-based Isolation Forest Feature Importance (DIFFI)*, to provide Global Feature Importances (GFIs), which represent a condensed measure describing the macro-behavior of the IF model on *training* data.
- A local version of the DIFFI method, called *Local-DIFFI*, to provide Local Feature Importances (LFIs) aimed at interpreting individual predictions made by the IF model at *test* time.
- A simple and effective procedure to perform unsupervised feature selection for AD problems based on the DIFFI method.
- A suitable proxy task, i.e., unsupervised feature selection, for *functionally-grounded evaluation* of interpretability methods [Doshi-Velez and Kim, 2017] for AD problems with no prior knowledge about the feature importance.

Each contribution mentioned above complies with the human-centered principle aimed at matching the user's needs to the best extent possible. This translates into some characteristics we sought to prioritize, e.g., limited computational times and light and straightforward hyperparameters tuning procedures. Additionally, our approach does not require additional knowledge (e.g., game theory concepts necessary to grasp the rationale behind the functioning of SHAP fully) since it is based on basic computations on quantities that naturally emerge from the principles governing the IF model. Along these lines, the proposed methods are consistent with the simplicity that characterizes the IF model, thus avoiding the risk of developing an interpretability framework that is more complex than the model itself. As we shall discuss in the next Sections, our method does not imply artificial manipulations of the data points and requires no fitting of interpretable local surrogate models. Put differently, the DIFFI method directly handles the original model and the original data points, with no local approximations and data perturbations. This results in an accurate and fully-transparent description of the IF inner structure, which cannot be achieved with model-agnostic interpretability methods.

3.5 Methods

In this Section, we extensively discuss the rationale behind the DIFFI method and thoroughly analyze each building block. We then propose a local variant of

the DIFFI approach, *Local-DIFFI*, for the interpretation of individual predictions. We conclude with the introduction of a novel method that is based on global DIFFI for unsupervised feature selection, which also serves as a 'proxy task' for evaluating the quality of attributed feature importance scores when prior knowledge about the problem is not available. This framework for evaluating feature importance scores can be leveraged as a *functionally-grounded evaluation* method [Doshi-Velez and Kim, 2017] in the context of unsupervised AD.

3.5.1 DIFFI

The DIFFI method relies on two simple hypotheses exploited to define feature importance for the AD task at hand. A split test associated with a feature deemed as important should:

I1) Induce the isolation of anomalous data points at small depths (i.e., close to the root) while relegating regular data points to the bottom end of the trees;

I2) Produce a higher imbalance on anomalous data points, while ideally being useless on regular points.

Notice that if contrary to the hypotheses above, the most relevant features induced isolation of regular data points at small depths and produced a higher imbalance on regular data points, the resulting IF model would try to isolate regular (rather than anomalous) data points. Therefore, we would obtain a model whose objective is not consistent with that of AD.

Let us consider a trained instance of the IF detector and the corresponding training set $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$. For each tree t we partition, based solely on the predictions produced by tree t, the assigned bootstrap sample $\mathcal{D}_t$ into the subset of predicted inliers $\mathcal{P}_{I,t}$ and the subset of predicted outliers $\mathcal{P}_{O,t}$, where

$$\mathcal{P}_{I,t} = \{\mathbf{x}_i \in \mathcal{D}_t \,|\, \hat{y}_{i,t} = 0\},$$
$$\mathcal{P}_{O,t} = \{\mathbf{x}_i \in \mathcal{D}_t \,|\, \hat{y}_{i,t} = 1\},$$

and $\hat{y}_{i,t}$ denotes the prediction produced by tree t associated to $\mathbf{x}_i$. Predictions are obtained, as usual, through a thresholding operation on the anomaly scores, which are now computed by replacing $\bar{h}(\mathbf{x}_i)$ with $h_t(\mathbf{x}_i)$ in (2.1). The choice to consider only bootstrap samples for each tree, rather than the entire training set, is motivated by the need to reduce the computational cost and the desire to decouple the evaluation of feature importance scores from the generalization capability of the trained model. Indeed, each tree is trained only on the corresponding bootstrap sample, and by considering the whole training set, the performance of the tree on unseen data (other than the bootstrap sample) might affect the resulting feature importance scores. In general, this is not a problem since training data are supposed to be drawn from the same distribution and, thus, the performance on in-bag and out-of-bag samples should be similar. Nonetheless,

considering only in-bag samples for each tree, the computational cost can be made independent of the training set size.

We will define *Cumulative Feature Importances (CFIs)* for inliers and outliers, real-valued quantities that will be then properly normalized and combined together to produce the final feature importance measures. CFIs are updated in an additive fashion by exploiting data points in $\mathcal{P}_{I,t}$ and $\mathcal{P}_{O,t}$, for $t = 1, \ldots, T$. The update rule depends on two quantities that reflect the two intuitions explained above: the depth of the leaf node where a specific data point ends up (intuition I1) and the *Induced Imbalance Coefficient (IIC)* associated with a specific internal node (intuition I2). In the remainder of this Section, we first explain how to compute IICs, then we describe the procedure for the update of CFIs for inliers and outliers and combine them to produce the final GFIs. An overview of the method and its building blocks is given in Figure 3.1.

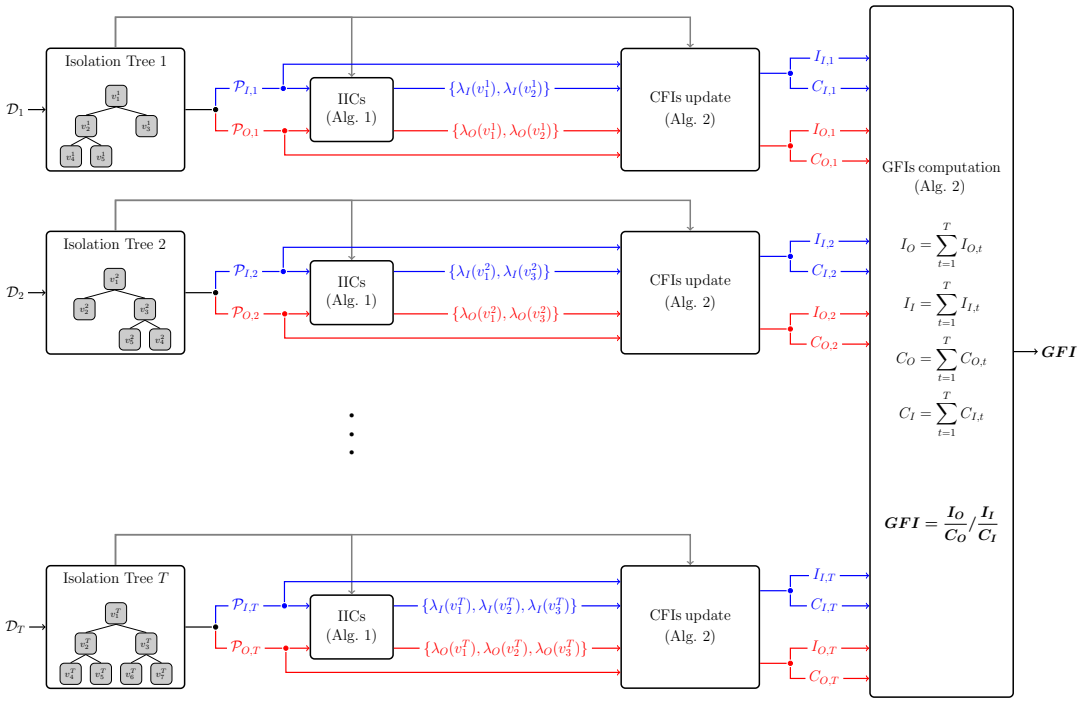

Figure 3.1: Overview of the DIFFI method for global feature importance scores. The notation v_i^t denotes the i-th node in the t-th tree. The quantities and the information flow related to predicted inliers/outliers are in blue/red, respectively. Boxes denote computation blocks: more details on IICs and CFIs/GFIs computation will be provided in Alg. 1 and Alg. 2 respectively.

3.5.1.1 IICs computation

Let us consider the generic internal node v in tree t. Let $n(v)$ represent the number of data points associated with node v, $n_l(v)$ the number of data points

associated with its left child and $n_r(v)$ the number of data points associated with its right child. The IIC of node v, denoted $\lambda(v)$, is obtained as follows

$$\lambda(v) = \begin{cases} 0, & \text{if } n_l(v) = 0 \text{ or } n_r(v) = 0 \\ \\ \tilde{\lambda}(v), & \text{otherwise} \end{cases} \tag{3.1}$$

where

$$\tilde{\lambda}(v) = \Gamma\left(\frac{\max(n_l(v), n_r(v))}{n(v)}\right) \tag{3.2}$$

and $\Gamma(\cdot)$ is a scaling function mapping its input into the interval $[0.5, 1]$, whose rationale is detailed below. In (3.1), the first case represents a *useless split*, in which all data points are sent either to the left or right child. The best possible split, instead, is what we call an *isolating split*: this happens when either the left or the right child receives exactly one data point. An isolating split is assigned the highest possible IIC, i.e., 1.

For the experiments, we use the following scaling function

$$\Gamma(a) = \frac{a - \lambda_{min}(n)}{\lambda_{max}(n) - \lambda_{min}(n)} \cdot 0.5 + 0.5, \tag{3.3}$$

where $\lambda_{min}(n)$ and $\lambda_{max}(n)$ denote the minimum and maximum scores, respectively, that can be obtained a priori[1] given the number $n(v)$ of data points associated to the specific node location v. We notice that by scaling the values, we can mitigate undesired effects due to the specific value of $n(v)$, which should not affect the computation of the induced imbalance. For instance, if $n = 10$ data points are associated with a specific node, the worst non-useless split (5 points to each child) leads to $\lambda_{min} = 0.5$. Instead, if $n = 11$, the worst non-useless split (5 samples to one child and 6 to the other one) leads to $\lambda_{min} = \frac{6}{11} = 0.54$. With the introduction of the scaling function $\Gamma(\cdot)$, the two scenarios depicted above are equivalent at least for the extreme cases of isolating split and worst non-useless split.

As it will be clear later on, we need to distinguish between IICs for inliers, denoted $\lambda_I(v)$ and computed by exploiting the predicted inliers $\mathcal{P}_{I,t}$, and the counterpart for outliers, denoted $\lambda_O(v)$ and computed by exploiting the predicted outliers $\mathcal{P}_{O,t}$.

The computation of IICs is performed for each internal node v of each tree $t = 1, \ldots, T$ in the forest. The pseudocode is reported in Algorithm 1.

3.5.1.2 CFIs update

We distinguish between CFI for inliers, denoted I_I, and the counterpart for outliers, denoted I_O. Both I_I and I_O are p-dimensional vectors, where the j-th component represents the CFI (for inliers or outliers) for the j-th feature.

[1]Here "a priori" means before applying the scaling function $\Gamma(\cdot)$.

Algorithm 1 DIFFI - IICs computation.

Input: Isolation Tree t, set of data points $\mathcal{S}$
Output: IICs vector $\Lambda \in \mathbb{R}^{|t|}$ ($|t|$ number of internal nodes in tree t)

function IIC($t, \mathcal{S}$)
  for internal node v in t **do**
    Compute $\lambda(v)$ as in (3.1), using samples in $\mathcal{S}$
    $\Lambda[v] \leftarrow \lambda(v)$
  end for
  return Λ
end function

Let $Path(\mathbf{x}, t)$ be the path from the root node to the corresponding leaf node associated with data point $\mathbf{x}$ in tree t. For simplicity we restrict the attention to the CFI update rule for inliers (i.e., I_I), as the extension to I_O is immediate. First we initialize $I_I = \mathbf{0}_p$, where $\mathbf{0}_p$ denotes the p-dimensional vector of zeros. Then we update I_I in an additive fashion. Specifically, we iterate over all trees in the forest and then, for each tree t, over the subset of predicted inliers $\mathcal{P}_{I,t}$. Finally, for the generic predicted inlier $\mathbf{x}_I \in \mathcal{P}_{I,t}$, we iterate over the internal nodes in its path $Path(\mathbf{x}_I, t)$. If the splitting feature associated with the generic internal node v is f_j, then we update the j-th component of I_I by adding the quantity

$$\Delta = \frac{1}{h_t(\mathbf{x}_I)} \cdot \lambda_I(v), \tag{3.4}$$

where we recall that $h_t(\mathbf{x}_I)$ denotes the depth of the leaf node (in tree t) associated with data point $\mathbf{x}_I$. In (3.4), we can notice the contributions of two factors. The right-hand side factor characterizes the "local" effect of the split through the induced imbalance at that specific node location. The left-hand side factor, instead, characterizes the "global" effect of the split taking into account potential situations in which an apparently bad (from the "local" perspective) split actually re-organizes the data points in a way that makes it easier for subsequent split tests to isolate them.

As regards the update rule for I_O, the only differences with respect to the procedure detailed above are that we iterate over $\mathcal{P}_{O,t}$ rather than $\mathcal{P}_{I,t}$ and that we replace $\lambda_I(v)$ with $\lambda_O(v)$ in (3.4).

3.5.1.3 GFIs computation

Recalling that in the IF, differently to what happens in the RF algorithm, the splitting features are selected randomly, the careful reader should perceive a potential issue: if the generic feature f was sampled more frequently than others, it would unfairly receive a higher CFI. We define the *features counter* for inliers, denoted C_I, and the counterpart for outliers, denoted C_O, as p-dimensional vectors where the j-th component represents how many times the j-th feature

appeared while updating the CFIs. C_I is updated iterating over $\mathcal{P}_{I,t}$, while C_O is updated iterating over $\mathcal{P}_{O,t}$, for $t = 1, \dots, T$. In order to filter out the effect of random splitting feature selection, we simply normalize the CFIs by their corresponding feature counters, C_I and C_O respectively. The GFIs are then obtained as

$$GFI = \frac{I_O/C_O}{I_I/C_I}, \tag{3.5}$$

where divisions are performed element-wise. Notice that higher feature importance for inliers (i.e., a high value of the denominator in (3.5)) implies lower overall feature importance. This is consistent with intuition I1: important features isolate outliers closer to the root and simultaneously do not contribute to the isolation of inliers. Algorithm 2 summarizes the whole procedure at the core of the DIFFI method. IICs are computed according to the function described in Algorithm 1. The function Feat$(\cdot)$ returns the feature associated with the internal node passed as argument.

3.5.2 Local-DIFFI

For the interpretation of individual predictions produced by the IF, we exploit a procedure similar to the one described in Section 3.5.1 with differences due to the impossibility of computing some quantities in the local case (i.e., when considering one sample at a time). Specifically:

- the Induced Imbalance Coefficients cannot be computed since we consider only one sample;
- all quantities referred to predicted inliers cannot be computed since the focus is on the interpretation of predicted outliers.

Given a predicted outlier $\mathbf{x}_O$, the corresponding *Local Feature Importance (LFI)* is computed as

$$LFI(\mathbf{x}_O) = \frac{I_O^{loc}}{C_O^{loc}}, \tag{3.6}$$

where, similarly to the global case, C_O^{loc} is the features counter and I_O^{loc} is the CFI associated with $\mathbf{x}_O$. Both C_O^{loc} and I_O^{loc} are updated in an additive fashion by iterating over the ITs in the forest and over the internal nodes in the path $Path(\mathbf{x}_O, t)$ in each IT. Differently from the global case, if the splitting feature associated with the generic internal node v (in tree t) is f_j, then we update the j-th component of I_O^{loc} by adding the quantity

$$\Delta_{loc} = \frac{1}{h_t(\mathbf{x}_O)} - \frac{1}{h_{max}}. \tag{3.7}$$

The correction term $-\dfrac{1}{h_{max}}$ in (3.7) makes sure that CFI is not updated when the leaf node associated with the predicted outlier $\mathbf{x}_O$ is at the maximum depth of the

Algorithm 2 DIFFI - CFIs update and GFIs computation.

Input: Isolation Forest F, bootstrap samples $\{\mathcal{D}_1, \ldots, \mathcal{D}_T\}$
Output: global feature importance scores $GFI \in \mathbb{R}^p$

Initialize counter and cumulative importance variables:
for $j = 1, \ldots, p$ **do**
    $C_I(f_j), I_I(f_j), C_O(f_j), I_O(f_j) \leftarrow 0$
end for
for Isolation Tree t in Isolation Forest F **do**
    Get predicted inliers and outliers according to tree t:
    $\mathcal{P}_{I,t}, \mathcal{P}_{O,t} \leftarrow \text{Predict}(\mathcal{D}_t, t)$
    Get IICs associated with inliers and outliers:
    $\Lambda_I \leftarrow \text{IIC}(t, \mathcal{P}_{I,t})$
    $\Lambda_O \leftarrow \text{IIC}(t, \mathcal{P}_{O,t})$
    Update CFIs:
    for $\mathbf{x} \in \mathcal{P}_{I,t}$ **do**
        for internal node v in $\text{Path}(\mathbf{x}, t)$ **do**
            $f = \text{Feat}(v)$
            $C_I(f) \leftarrow C_I(f) + 1$
            $I_I(f) \leftarrow I_I(f) + \frac{1}{h_t(\mathbf{x})} \cdot \Lambda_I[v]$
        end for
    end for
    for $\mathbf{x} \in \mathcal{P}_{O,t}$ **do**
        for internal node v in $\text{Path}(\mathbf{x}, t)$ **do**
            $f = \text{Feat}(v)$
            $C_O(f) \leftarrow C_O(f) + 1$
            $I_O(f) \leftarrow I_O(f) + \frac{1}{h_t(\mathbf{x})} \cdot \Lambda_O[v]$
        end for
    end for
end for
Compute FI for each feature:
for $j = 1, \ldots, p$ **do**
    $\text{GFI}(f_j) = \frac{I_O(f_j)}{C_O(f_j)} / \frac{I_I(f_j)}{C_I(f_j)}$
end for

tree. If the correction term were not used, the CFI of each feature in $Path(\mathbf{x}_O, t)$ would also be updated in cases where the data point under examination is not isolated (i.e., when it ends up in leaf nodes at the maximum depth) as Δ_{loc} would always be greater than zero. The pseudocode of the Local-DIFFI method is reported in Algorithm 3. The function $\text{Feat}(\cdot)$ returns the feature associated with the internal node passed as argument.

Algorithm 3 Local-DIFFI.

Input: Isolation Forest F, predicted outlier $\mathbf{x}_O$
Output: local feature importance scores $LFI \in \mathbb{R}^p$

Initialize counter and cumulative importance variables:
for $j = 1, \dots, p$ **do**
    $C_O^{loc}(f_j), I_O^{loc}(f_j) \leftarrow 0$
end for
for Isolation Tree t in Isolation Forest F **do**
    Update CFIs:
    for internal node v in Path$(\mathbf{x}_O, t)$ **do**
        $f = \text{Feat}(v)$
        $C_O^{loc}(f) \leftarrow C_O^{loc}(f) + 1$
        $I_O^{loc}(f) \leftarrow I_O^{loc}(f) + \frac{1}{h_t(\mathbf{x}_O)} - \frac{1}{h_{max}}$
    end for
end for
Compute FI for each feature:
for $j = 1, \dots, p$ **do**
    $\text{LFI}(f_j) = \frac{I_O^{loc}(f_j)}{C_O^{loc}(f_j)}$
end for

3.5.3 Unsupervised feature selection with global DIFFI

The DIFFI method outlined in Section 3.5.1 can be effectively exploited to perform feature selection in the context of AD problems when labels associated with training data points are not available. The procedure consists in training N_{fs} different instances of IF obtained with the same training data but different random seeds in order to filter out effects due to the stochasticity inherently present in the model. The global DIFFI scores associated with each instance of IF are then aggregated to define a ranking on the features as follows:

1. We define the p-dimensional vector of aggregated scores $S_{agg} \in \mathbb{R}^p$ initialized as a p-dimensional vector of zeros, where p is the number of features.

2. For each of the N_{fs} IF instances:

    - we rearrange the global DIFFI scores in decreasing order, thus obtaining a ranking of the features (for the specific IF instance) from the most important one to the least important one;
    - we update S_{agg} by adding, for each feature, a quantity that is a function of the estimated rank $\hat{r}$, namely

$$\Delta_{fs} = 1 - \frac{\log(\hat{r})}{\log(p)}. \tag{3.8}$$

Figure 3.2: Update function for aggregated scores: an example with $p = 100$.

Notice that in (3.8), we differentiate more the scores among the most important features, while for the least important ones the scores are similar and very small (see Figure 3.2).

3. The resulting vector of aggregated scores S_{agg} is then used to define a ranking over the features: the higher the aggregated score, the more important the feature.

The motivations behind this strategy stem from the human-centered design principle adopted in this work and can be summarized as follows:

- Users can spend only a limited amount of time on preprocessing operations since, especially in production environments, the deployment of novel algorithmic solutions is usually meant to react to emerging issues promptly. The light computational cost of the DIFFI method, thanks to the in-bag samples trick, is particularly appealing in such time-constrained applications.

- In order to minimize the effort of users, the choice of the IF (combined with DIFFI) as a proxy model to produce a ranking on the features is attractive as it introduces just a few hyperparameters to be tuned; in addition, it is worth mentioning that the IF is often preferred over other AD algorithms due to its good performance with the default hyperparameters values suggested in the original paper.

- The proposed strategy for unsupervised feature selection takes into account the nature of the task, while most other methods do not. This is particularly

important for AD tasks as relevant features for classification might not be relevant for AD. The user interested in solving an AD problem may trust more a method specifically suited for such a purpose than other task-agnostic methods.

As anticipated in Section 3.4, in addition to the unquestionable usefulness of the unsupervised feature selection task, the procedure outlined above also represents an excellent proxy to indirectly assess the quality of the feature importance scores provided by the global DIFFI method described in Section 3.5.1. Indeed, good feature importance scores would lead to a good ranking of the features, which in turn can be considered as a good solution to the unsupervised feature selection problem. Therefore, given a pre-specified number of features k that an AD model could exploit, we would expect an effective interpretability method to provide the most informative subset of k features to solve the AD task (i.e., the subset of k features delivering the best possible performance).

3.6 Experimental Results

We report in this Section experimental results on synthetic and real-world datasets to assess the effectiveness of both the global DIFFI method (to provide global feature importance scores and to perform unsupervised feature selection) and its local variant to provide feature importance scores associated with individual predictions. We also provide analyses of the global and local DIFFI methods on synthetic data to assess the relevance of each component. All experiments were run on a standard consumer laptop equipped with an Intel Core i7-8750H 2.20GHz CPU and 16 GB RAM.

We make the code publicly available[2] to enhance the reproducibility of our experimental results and to foster research in the field.

3.6.1 Global interpretation of the Isolation Forest model

We first evaluate the global DIFFI method on controlled experiments on synthetic data. The synthetic dataset was created by initially considering 2-dimensional data points whose dimension is then augmented by adding noise features, similar to what was done in [Carletti et al., 2019]. Specifically, the generic data point $\mathbf{x}$ is represented by the p-dimensional vector

$$\mathbf{x} = [\varrho\cos(\vartheta), \varrho\sin(\vartheta), e_1, \ldots, e_{p-2}]^\intercal, \tag{3.9}$$

where $e_j \sim \mathcal{N}(0,1)$ for $j = 1, \ldots, p-2$ are white noise samples. Parameters ϱ and ϑ are random variables drawn from continuous uniform distributions. For regular data points, we have

$$\vartheta \sim \mathcal{U}(0, 2\pi), \qquad \varrho \sim \mathcal{U}(0, 3), \tag{3.10}$$

[2] https://github.com/mattiacarletti/DIFFI

while for anomalous data points, we have

$$\vartheta \sim \mathcal{U}(0, 2\pi), \qquad \varrho \sim \mathcal{U}(4, 30). \tag{3.11}$$

For our experiments, we consider a training set composed of 1000 6-dimensional data points (thus 4 noise features), with 10 % anomalies. We trained 10 instances of IF with 100 trees and $\psi = 256$ (typical choices for the IF hyperparameters [Liu et al., 2008]), obtaining an average F1-score on training data equal to 0.71 (standard deviation 0.03). Since the models we obtain achieve satisfactory detection performance, it is reasonable to assume that the informative features (i.e., $\varrho\cos(\vartheta)$ and $\varrho\sin(\vartheta)$ in (3.9)) are effectively exploited to solve the task. Based on this observation, for each trained instance of the IF model, we consider the feature importance scores provided by the global DIFFI method and rank the features accordingly. As expected, the coordinates of the points are always ranked as the two most important features. If we do not consider the IIC contribution (i.e., if we set $\lambda_I(v) = 1$ in (3.1)), the gap between the normalized importance scores of the informative features and the noise features is significantly reduced, making it harder to tell them apart. Similar issues arise when we only consider the contribution of predicted outliers (i.e., when setting $I_I/C_I = 1$ in (3.5)). In this latter case, $\varrho\cos(\vartheta)$ does not appear amongst the top-2 most important features for 3 (out of 10) models, and the same happens with $\varrho\sin(\vartheta)$ for 2 (out of 10) models.

3.6.2 Interpretation of individual predictions

For the interpretation of individual predictions provided by the IF model, we exploit the local variant of the DIFFI method described in Section 3.5.2. We assess the effectiveness of the Local-DIFFI method on a synthetic dataset and a real-world dataset. On both datasets, we have prior knowledge about the most relevant features for the AD task to be solved, which would be fundamental for evaluating the performance of DIFFI. We remark how finding real-world data for AD problems with a priori knowledge of the relevant features is not a trivial task. The experimental setup adopted here simulates a real scenario of remarkable interest in several application domains: given a trained instance of the IF, the user is interested in deploying the model in online settings to get the prediction and the corresponding local feature importance scores associated with the individual data point being processed. Typical applications include (but are not limited to) the monitoring of smart manufacturing systems and the detection of abnormal patterns in healthcare data: in both examples, the promptness of responses may be crucial to ensure quick and effective corrective actions for the industrial processes/machines or the well-being of patients.

3.6.2.1 Synthetic dataset

As done in Section 3.6.1 for the global DIFFI method, we first analyze the performance of the Local-DIFFI method in controlled experiments on synthetic

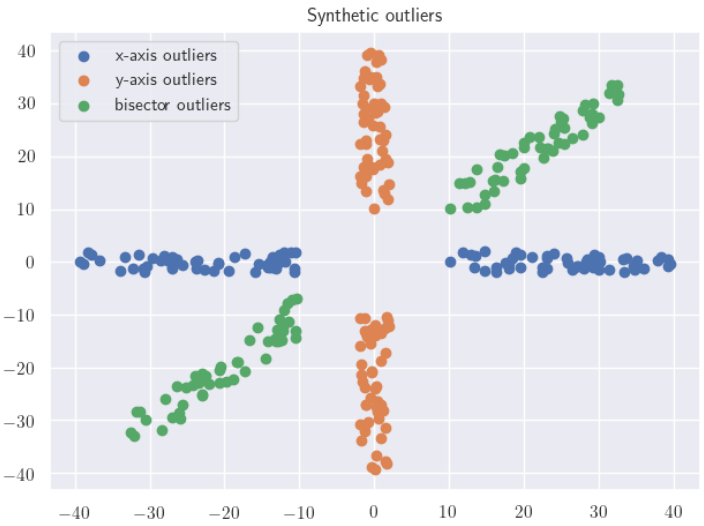

Figure 3.3: Synthetic outliers projected on the $f_1 - f_2$ plane.

data. The training set is the same considered in Section 3.6.1 and is exploited to train an instance of IF with 100 trees and $\psi = 256$ (that are typical choices for the IF hyperparameters [Liu et al., 2008]), obtaining an F1-score on training data equal to 0.76. For the testing phase, we generated 300 additional ad-hoc anomalies, displayed in Figure 3.3 (projected on the subspace of relevant features): 100 lying on the x-axis (blue points), 100 on the y-axis (orange points) and 100 on the bisector (green points). The prior knowledge for this AD task is represented by the fact that only feature f_1 is relevant for outliers on the x-axis, only feature f_2 is relevant for outliers on the y-axis and both f_1 and f_2 are relevant for outliers on the bisector (all the other features, being white noise samples, are irrelevant in all cases). Once the predictions associated with the generated test outliers are obtained, we run the Local-DIFFI algorithm to get the local feature importance scores and the corresponding feature rankings. We compared the performance of the Local-DIFFI method with SHAP and LIME. In Figure 3.4, features are color-coded, the columns represent the ranks, and the height of the bar associated with each feature represents the fraction of predicted anomalies for which the feature has a specific rank. For example, in the top left plot in Figure 3.4, feature f_1 has rank 1 for all predicted anomalies. Instead, feature f_2 has rank 2 for $\sim 40\%$ of predicted anomalies, rank 3 for $\sim 30\%$ of predicted anomalies, rank 4 for $\sim 10\%$ of predicted anomalies, and so on. As can be seen, both Local-DIFFI and SHAP perfectly identify the actual important feature(s): in the first two rows, for all correctly predicted outliers, the first column (representing the most important feature as estimated by the interpretability method) is always associated with the correct feature, namely f_1 and f_2 for outliers on the x-axis and on the y-axis, respectively; in the third

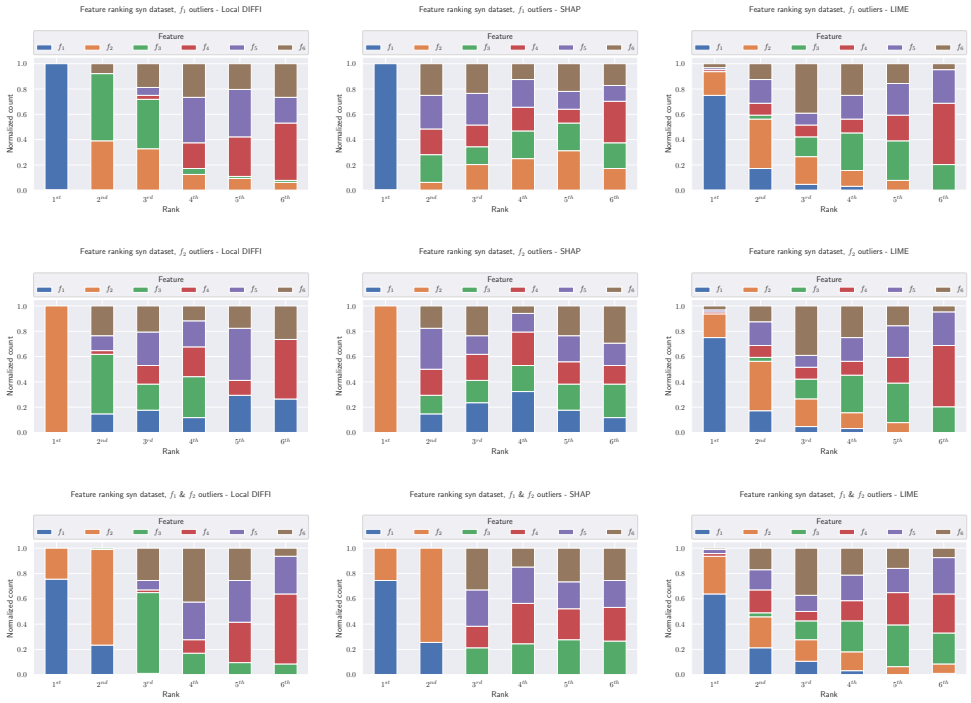

Figure 3.4: Feature rankings for the synthetic dataset based on Local-DIFFI scores (left column), SHAP scores (central column) and LIME scores (right column): outliers on the x-axis (first row), on the y-axis (second row) and on the bisector (third row).

row (referred to the points on the bisector), instead, both f_1 and f_2 are deemed important by Local-DIFFI and SHAP, thus aligning with prior knowledge. In this latter case, we also observed that feature importance scores provided by Local-DIFFI for feature f_1 and f_2 are comparable, while the same rightly does not happen for outliers on the axes. Feature rankings provided by LIME do not align well with prior knowledge. This is probably due to issues related to the sampling strategy used to produce perturbed data points, which might not belong to the data manifold, and/or to the well-known instability of the explanations [Alvarez-Melis and Jaakkola, 2018]. A major advantage of Local-DIFFI over SHAP and LIME is the computational time: while SHAP and LIME have an average execution time of 0.221 and 0.291 seconds per sample, respectively, Local-DIFFI runs in 0.023 seconds per sample on average. Along the lines of the experiments described in Section 3.6.1, we performed an ablation study to assess the role of the correction term in (3.7). Similarly to what happens with the global DIFFI method, when we ignore the effect of the correction term (i.e., when setting $-\frac{1}{h_{max}} = 0$ in (3.7)), the gap between the normalized importance scores of the informative features and the noise features is significantly reduced. This is due to the fact that, without the modulation of the correction term,

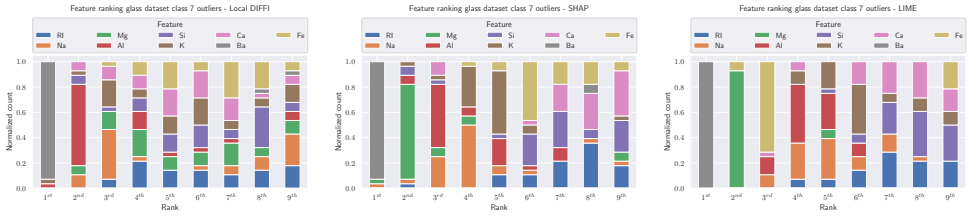

Figure 3.5: Feature rankings for the glass dataset based on Local-DIFFI scores (left column), SHAP scores (central column), and LIME scores (right column): class 7 outliers (headlamps glass).

the CFI is also being updated for useless splits, i.e., splits that do not induce isolation of the data point before the maximum depth of the tree is reached.

3.6.2.2 Real-world dataset

We consider a modified version of the Glass Identification UCI dataset[3] originally intended for multiclass classification tasks. The dataset consists of 213 glass samples represented by a 9-dimensional feature vector: one feature is the refractive index (RI), while the remaining features indicate the concentration of Magnesium (Mg), Silicon (Si), Calcium (Ca), Iron (Fe), Sodium (Na), Aluminum (Al), Potassium (K) and Barium (Ba). Originally the glass samples were representative of seven categories of glass type, but for our experiments, we grouped classes 1, 2, 3, and 4 (i.e., window glass) to form the class of regular points, while the other three classes contribute to the set of anomalous data points (i.e., non-window glass): containers glass (class 5), tableware glass (class 6) and headlamps glass (class 7). We assess the performance of Local-DIFFI on predicted outliers belonging to class 7, considered as test data points. Similarly to [Gupta et al., 2018], we exploit prior knowledge on headlamps glass: the concentration of Aluminum, used as a reflective coating, and the concentration of Barium, which induces heat-resistant properties, should be important features when distinguishing between headlamps glass and window glass. We trained an instance of IF with 100 trees and $\psi = 64$ and obtained an F1-score on training data equal to 0.55. On the test data (class 7), the IF identified 28 out of 29 anomalies. As for the synthetic dataset, we ran the Local-DIFFI algorithm to get the local feature importance scores and compared the performance with those obtained with SHAP and LIME. As shown in Figure 3.5, Local-DIFFI identifies the concentration of Barium and Aluminum as the most important features in the vast majority of predicted anomalies, well aligned with a priori information about the task. The same cannot be said for SHAP and LIME: while the most important feature is still the concentration of Barium, the second most important feature for almost all predictions is the concentration of Magnesium.

[3] https://archive.ics.uci.edu/ml/datasets/Glass+Identification

Table 3.1: AD datasets used for unsupervised feature selection experiments.

Dataset	Num samples	Num features	Contamination
satellite	6435	36	32%
cardio	1831	21	9.6%
ionosphere	351	33	36%
lympho	148	18	4.1%
musk	3062	166	3.2%
letter	1600	32	6.25%
mnist	7603	100	9.2%
optdigits	5216	64	3%
vowels	1456	12	3.4%
annthyroid	7200	6	7.42%

Additionally, also in this case, the Local-DIFFI exhibits way smaller execution time (0.019 seconds per sample on average) than SHAP (0.109 seconds per sample on average) and LIME (0.301 seconds per sample on average).

3.6.3 Unsupervised feature selection

According to the procedure outlined in Section 3.5.3, we exploit the global DIFFI scores to define a ranking over the features representing the data points in the problem at hand. In all experiments described below, we run $N_{fs} = 5$ instances of IF. To verify the quality of the selected features, we perform experiments on popular AD datasets from the Outlier Detection DataSets (ODDS) database[4], whose characteristics are summarized in Table 3.1. Once the ranking is obtained, we train an instance of IF by exploiting only the top k most important features (according to the ranking), with $k = 1, \dots, p-1$. We repeat the procedure 30 times (with different random seeds) and compute the median F1-score. We provide comparisons with two other commonly used unsupervised feature selection techniques, i.e., Laplacian Score [He et al., 2006], and SPEC [Zhao and Liu, 2007]. We did not consider other techniques such as Nonnegative Discriminative Feature Selection [Li et al., 2012], and $l_{2,1}$-Norm Regularized Discriminative Feature Selection [Yang et al., 2011] due to their prohibitive computational cost, which makes extensive usage practically cumbersome. The hyperparameters values for the final IF model are tuned separately for each dataset through grid search, exploiting all the available features, and then kept fixed for all the experiments involving the same dataset. For the unsupervised feature selection methods, instead, we set the hyperparameters to the default values to be consistent with the goal of minimizing user efforts in time-consuming operations. Furthermore, this approach perfectly fits real-world applications

[4] http://odds.cs.stonybrook.edu/

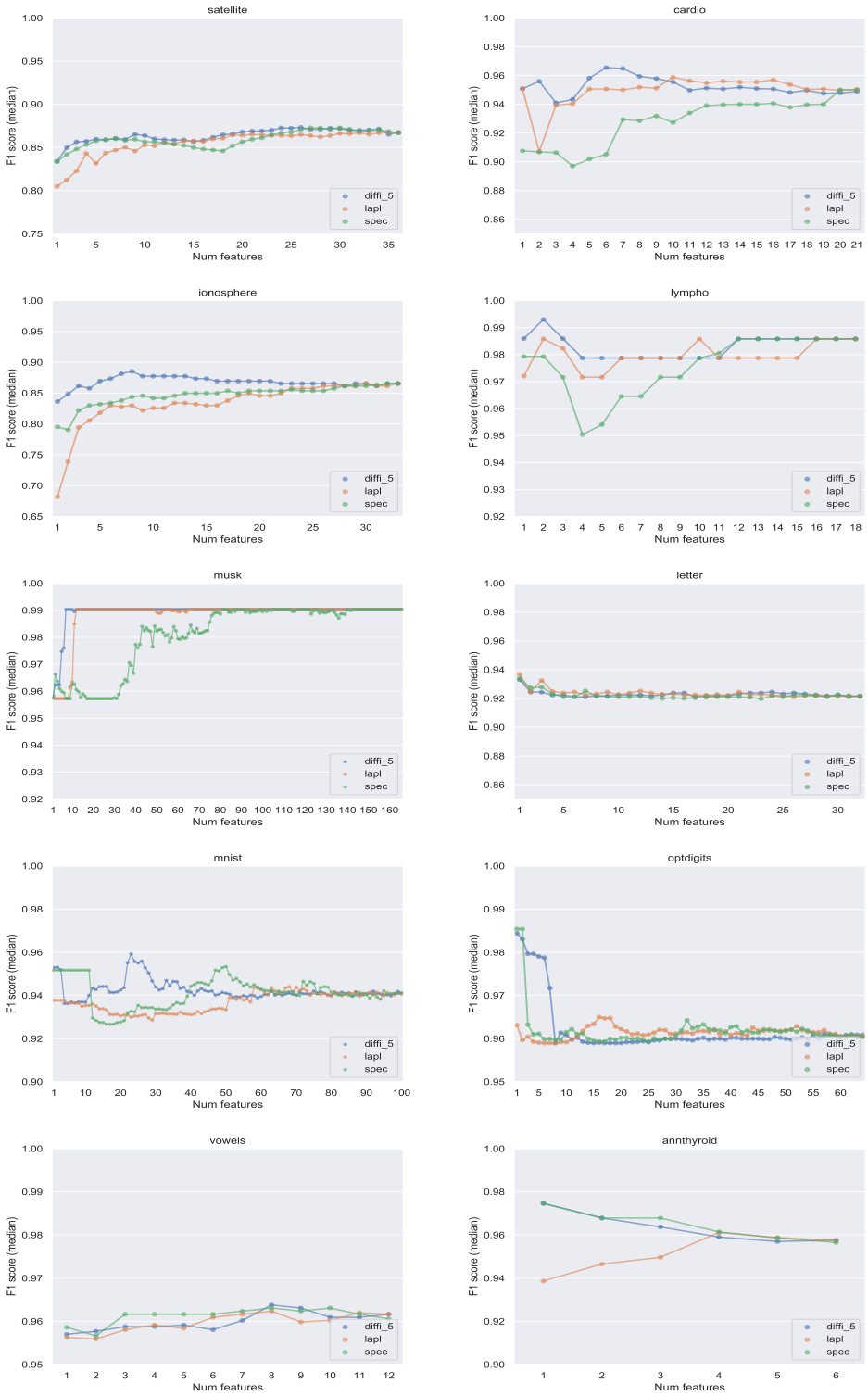

Figure 3.6: Evaluation of the global DIFFI method (`diffi_5`) for unsupervised feature selection, compared with Laplacian Score (`lapl`) and SPEC (`spec`) methods.

in which the lack of ground truth labels prevents the design of any principled hyperparameters tuning procedure, leading users to rely on existing heuristic rules.

As can be seen in Figure 3.6, the performance of DIFFI is comparable with those of the Laplacian Score and SPEC methods and consistently outperform them for a wide range of k values (i.e., number of exploited features) for the `cardio`, `ionosphere`, and `musk` datasets. Also, in most cases, DIFFI can identify the optimal combination of features, i.e., the subset of features leading to the highest (median) F1-score value.

We believe that task-specific methods such as DIFFI are preferable over task-agnostic methods (like Laplacian Score and SPEC) as the features that are actually relevant to solve a classification problem might not be relevant to solve an AD problem [Puggini and McLoone, 2018]. This comes as no surprise in light of the different nature of the two tasks, and it may unconsciously affect the user preference when reasoning about the most appropriate approach. Additionally, as mentioned in Section 3.5.3, the procedure based on DIFFI requires minimal - if any - hyperparameters tuning: the only hyperparameters are inherited from the underlying proxy model, i.e., an instance of IF, which has proved to provide satisfactory performance with the default hyperparameters values on a broad spectrum of applications.

3.7 Case Study: Semiconductor Manufacturing

In this Section, we assess the effectiveness of the global DIFFI method in the context of semiconductor manufacturing for a Chemical Vapor Deposition (CVD) [Susto et al., 2011] monitoring problem. Specifically, we consider 4 datasets representing 4 different recipes, with different characteristics in terms of cardinality (see Table 3.2). Data points are represented as 25-dimensional feature vectors, where each feature is derived from sensor measurements installed on the equipment. It is worth highlighting that none of the considered datasets is equipped with ground truth labels indicating whether a specific data point is anomalous or not, therefore the detection performance of AD algorithms cannot be assessed. Nevertheless, the formulation of an AD problem with this experimental setup is reasonable and could be useful in many respects:

- AD algorithms (such as the IF) which associate an anomaly score to each data point are particularly useful in flagging potentially problematic situations, even in cases where the data point under examination does not clearly represent an anomaly.
- Unsupervised AD algorithms offer a multivariate alternative to simpler univariate control charts for monitoring purposes.

In addition to the above considerations, for the specific case study considered in this Section, we can also rely on prior knowledge due to the collaboration with

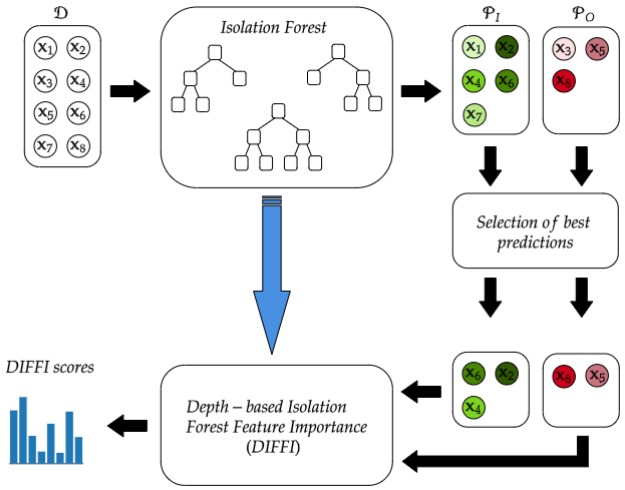

Figure 3.7: Overview of the approach adopted for the experiments on CVD datasets. Circles represent data points, where green indicates predicted inliers and red predicted outliers. The darker the shade, the more confident the prediction.

domain experts. More in detail, thanks to the physical interpretation that can be associated with most of the features, we have additional information about which are the most relevant features for the CVD process at hand. This can be considered as a (noisy) ground truth for the feature importance ranking and can be exploited to assess how well the feature importance ranking based on the global DIFFI scores is aligned with domain knowledge. The prior knowledge is represented by the claim that the indices of the most important features are $\{15, 16, 17, 18, 19, 20, 21, 22, 23\}$.

For our experiments, we train an instance of the IF with the default hyperparameters setting suggested in the original paper, i.e., $T = 100$ trees and sub-sampling size $\psi = 256$. For the DIFFI method, we do not exploit bootstrap samples as described in Section 3.5, but we consider a fixed number (50) of predicted inliers and predicted outliers with the lowest and highest anomaly scores, respectively. In this way, the global DIFFI scores are computed based on the data points whose associated predictions are the most confident. This reveals to a greater extent the inner logic of the IF in real-world scenarios, where less confident predictions may inject noise into the computation of the feature importance scores if the whole dataset was used. An overview of the adopted approach is given in Figure 3.7. For the sake of simple visualizations, we report experimental results referred to a single experiment, which is representative of a set of experiments performed with different random seeds, all leading to similar results.

We compare the DIFFI scores with the feature importance scores obtained

Table 3.2: Cardinality of CVD datasets.

Dataset	Num samples
Recipe 1	956
Recipe 2	34716
Recipe 3	3580
Recipe 4	13459

with the Permutation-based Importance (PIMP) method [Fisher et al., 2018] (based on the measure of variable importance introduced in [Breiman, 2001] for RFs). We adapted the PIMP method to unsupervised settings by evaluating the deviation from original predictions rather than the drop in detection performance under permutations of feature values. Results are reported in Figure 3.8. It is evident that both DIFFI and PIMP successfully identify the correct important features, i.e., those expected by domain experts. However, the relative difference between the most important features and the least important features obtained with PIMP is not as pronounced as that obtained with DIFFI.

3.8 Conclusions

This Chapter introduces Depth-based Isolation Forest Feature Importance (DIFFI), a method to provide interpretability traits to the IF model, one of the most popular and effective AD algorithms. By providing a quantitative measure of feature importance in the context of the AD task, DIFFI allows describing the behavior of IF at the global and local scales, providing insightful information that can be exploited by final users to get a better understanding of the underlying process and to enable root cause analysis.

DIFFI is as effective as the current state-of-the-art methods SHAP and LIME, with significantly smaller computational costs, making it really appealing for real-world production applications and even amenable to real-time scenarios. Moreover, we show that DIFFI can be employed to perform unsupervised feature selection, allowing the development of computationally parsimonious (and potentially more accurate) AD solutions. In a semiconductor manufacturing case study, the feature importance scores provided by the global DIFFI method aligned well with prior domain knowledge about the problem at hand (Chemical Vapor Deposition).

We believe that, given the exponentially growing interest in IF, DIFFI will be of paramount importance to enhance its usability and diffusion in real-world applications. We also believe that DIFFI would lead to an increase in the adoption of IF, as models naturally equipped with interpretability traits are usually preferred over black-box models [Rudin, 2019]. Concerning the latter aspect, future interdisciplinary studies could fruitfully explore the user

perspective through dedicated experiments.

With the formulation proposed in this work, the DIFFI method can only be leveraged to interpret the original IF model. To overcome this limitation, we envision DIFFI to be possibly extended to other tree-based models for AD, such as the Extended Isolation Forest [Hariri et al., 2019], the SCiForest [Liu et al., 2010], or the Streaming HSTrees [Tan et al., 2011]). In particular, the low computational costs open up the opportunity to exploit DIFFI in the flourishing field of online AD applications with streaming data, where time efficiency is crucial [Miao et al., 2018, Zhang et al., 2019b].

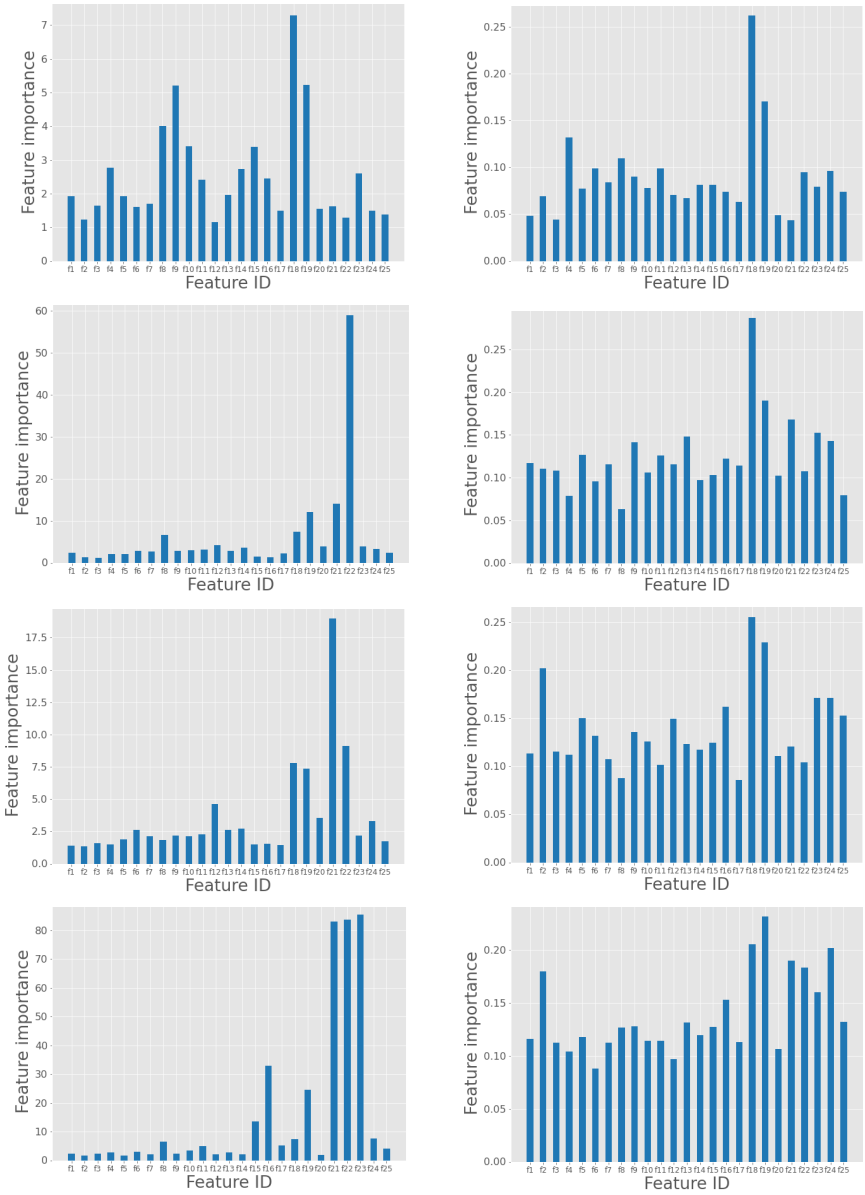

Figure 3.8: Feature importance scores for DIFFI (left) and PIMP (right) for the recipe 1 dataset (first row); recipe 2 dataset (second row); recipe 3 dataset (third row); and recipe 4 dataset (fourth row).

Part II

Interpretability and Robustness in Computer Vision

Chapter 4

Background on Computer Vision Models

4.1 Artificial Neural Networks

Artificial Neural Networks (ANNs) are the models at the heart of the extraordinary results of DL technologies in the past decade [Goodfellow et al., 2016]. They are loosely inspired by the structure and functioning of biological neural networks and consist of a collection of nodes (or *artificial neurons*) connected to each other. In Feedforward Neural Networks (FNNs), these nodes are organized in layers, and the information flows from the input to the output with no feedback loops, as shown in Figure 4.1. The first layer of the network is represented by the input variables and is dubbed *input layer*, while the last layer computes the final output of the model and is called *output layer*. Intermediate layers are called *hidden layers* since training samples do not provide information about the output for each hidden neuron of the network, but just the model input-output pair of values. In Figure 4.1, the network has only one hidden layer, but state-of-the-art models used in practice have several hidden layers to encode more complex functions. Models with more than one hidden layer are called Deep Neural Networks.

Each neuron of the network computes an affine transformation of its inputs and applies a non-linearity usually called *activation function*. In other words, if we denote with $\mathbf{x}^{k-1}$ the output of the $(k-1)$-th layer, the k-th layer computes the following function

$$\mathbf{x}^k = \sigma(\mathbf{W}^k \mathbf{x}^{k-1} + \mathbf{b}^k), \tag{4.1}$$

where $\mathbf{W}^k$ and $\mathbf{b}^k$ are the matrix of weights and the vector of biases of the k-th layer, respectively. The function $\sigma(\cdot)$ represents the activation function. Throughout this thesis, we will consider models with ReLU activation functions $\mathrm{ReLU}(x) = \max\{0, x\}$ (see Figure 4.2). Multilayer FNNs are universal approximators, in the sense that in principle they have the capability of approximating any measurable function to any desired degree of accuracy [Hornik et al., 1989].

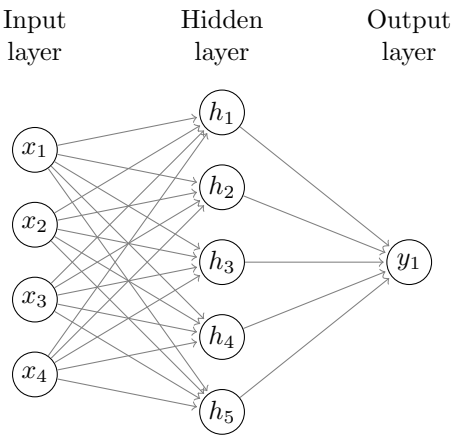

Figure 4.1: Feedforward Neural Network with one hidden layer.

The goal of the training process is, indeed, to optimize the weights and biases of the model - collectively denoted by $\boldsymbol{\vartheta}$ - so that the resulting function

$$y = f_{\boldsymbol{\vartheta}}(\mathbf{x}) \tag{4.2}$$

is a good approximation of the true underlying data-generating function

$$y = f^*(\mathbf{x}), \tag{4.3}$$

where in Equations (4.2) and (4.3) $(\mathbf{x}, y)$ represents a generic input-output pair. This is done by minimizing a suitable cost function $J(\boldsymbol{\vartheta})$. Typically, $J(\boldsymbol{\vartheta})$ can be written as

$$J(\boldsymbol{\vartheta}) = \mathbb{E}_{(\boldsymbol{x},y)\sim\hat{p}_{data}}\big[\mathcal{L}(f_{\boldsymbol{\vartheta}}(\boldsymbol{x}), y)\big], \tag{4.4}$$

where $\mathcal{L}$ is the per-sample loss function and $\hat{p}_{data}$ is the empirical distribution. Actually, the ultimate goal would be to reduce the expectation of the loss function $\mathcal{L}$ over the true data-generating distribution p_{data}, i.e., to minimize the so-called *risk*

$$R(f) = \mathbb{E}_{(\boldsymbol{x},y)\sim p_{data}}\big[\mathcal{L}(f_{\boldsymbol{\vartheta}}(\boldsymbol{x}), y)\big]. \tag{4.5}$$

Since the true data-generating distribution p_{data} is unknown and we only have a finite training set, we minimize the *empirical risk*

$$\widehat{R}(f) = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}(f_{\boldsymbol{\vartheta}}(\boldsymbol{x}^{(i)}), y^{(i)}), \tag{4.6}$$

where N is the number of training samples. In Chapters 5 and 6, we consider models trained by optimizing the cross-entropy loss via Stochastic Gradient Descent (SGD) [Bottou et al., 2018].

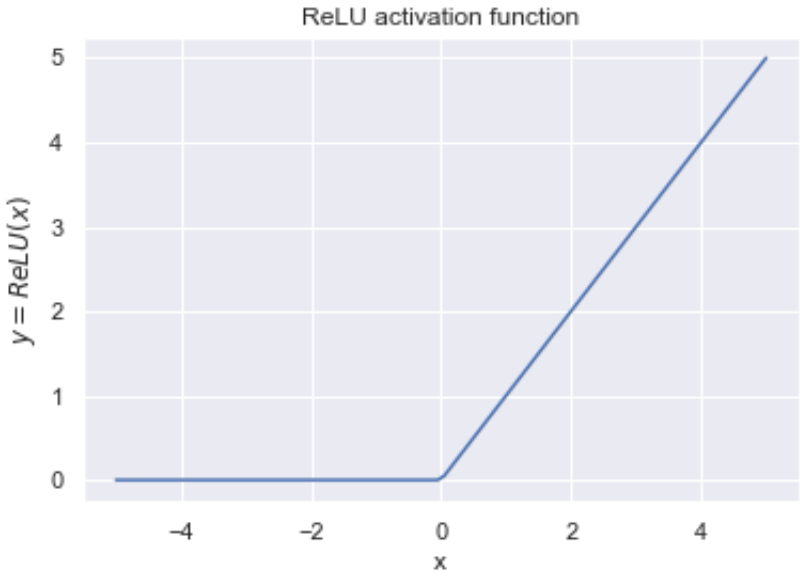

Figure 4.2: ReLU activation function.

4.2 Convolutional Neural Networks

Convolutional Neural Networks are sophisticated neural architectures to handle data with a grid-like topology [Goodfellow et al., 2016] that have earned tremendous success, especially for Computer Vision tasks [Krizhevsky et al., 2017, He et al., 2016a, Simonyan and Zisserman, 2014]. They are characterized by the fact that, in at least one of the hidden layers, *convolution*, a special kind of linear operation, is used in place of the usual matrix multiplication operation Equation (4.1). As can be seen in the example in Figure 4.3, the convolution operation has two arguments: the *input*, which is a multidimensional array of data, and a *convolutional kernel*, which is a multidimensional array of parameters. For the image classification tasks that we will consider in the next Chapters, both the input and the convolutional kernel have dimension 2. The output of the convolution operation is usually called *feature map* and is produced as follows. The kernel slides over the input array and performs an element-wise multiplication between its parameters and the values of the input data at the spatial locations selected by the area of the kernel. Finally, the results of this element-wise multiplication operation are summed together to produce a scalar value to be assigned to the corresponding spatial location in the feature map at the output. This procedure is repeated until the area of the whole input array is covered, and the feature map at the output is filled up. In Figure 4.3, the colored areas highlight how the value for a single location of the feature map is produced. Notice that the feature map produced by the 2D convolution operation has a reduced dimension compared to the input. In some practical

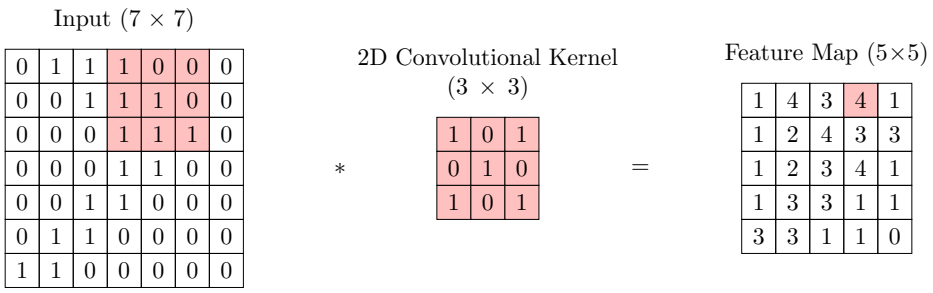

Figure 4.3: 2D convolution operator.

applications, we may want to preserve the original dimensions of feature maps, and this is done by adding extra fake pixels at the borders of the input array. This technique is called *padding*. The value of the added pixels could be 0 (*zero padding*) or replicate the value of pixels at the borders of the original input array. In modern CNNs, convolutional layers consist of a collection of convolutional kernels dubbed *filters*, so that both the input and the output of convolutional layers have an additional dimension, usually called the *channel dimension*. The function implemented by the k-th convolutional layer (with a single channel in input and output, for simplicity) can be written as

$$\mathbf{x}^k = \sigma(\mathbf{W}^k * \mathbf{x}^{k-1} + \mathbf{b}^k), \tag{4.7}$$

where the symbol $*$ denotes the convolution operator. The huge success of CNNs stems from three main properties of convolution, namely *sparse connectivity*, *parameter sharing*, and *equivariance to translation*. In particular, equivariance to translation means that if the input is shifted, the output of the convolutional layer will be shifted in the same way. See [Goodfellow et al., 2016] for more details.

Another useful property of CNNs is the invariance to small translations of the input, which is enforced by *pooling layers*. Pooling functions downsample feature maps by replacing groups of nearby activations with a scalar value that represents a summary statistics of the neighborhood. Popular choices for pooling functions are *max pooling* and *average pooling*.

Finally, CNNs usually employ *batch normalization layers* [Ioffe and Szegedy, 2015] to cope with the problem of internal covariate shift, namely the change in the distribution of a layer's inputs as a result of the update of parameters in the previous layer during training. If we denote with $\mathbf{y}^k$ the output of the batch normalization layer, we have that

$$\mathbf{y}^k = \gamma^k \hat{\mathbf{x}}^k + \beta^k, \tag{4.8}$$

where γ^k and β^k are learnable parameters, and $\hat{\mathbf{x}}^k$ is the whitened input to the layer

$$\hat{\mathbf{x}}^k = \frac{\mathbf{x}^k - \mathbb{E}[\mathbf{x}^k]}{\sqrt{\mathrm{Var}[\mathbf{x}^k]}}. \tag{4.9}$$

In practice, the expectation and variance in Equation (4.9) are estimated over mini-batches of the training set and updated in an online fashion.

Throughout the thesis we consider image classifiers encoded by CNNs of the form $f \circ g$, where f is a feature extractor with learnable parameters $\boldsymbol{\vartheta}_f$ and g is a fully-connected linear layer with learnable parameters $\boldsymbol{\vartheta}_g$. We refer to the outputs of f as *latent features*. CNNs considered in this thesis are composed of stacked sequences of convolutional layers, batch normalization layers, ReLU activation functions, and pooling layers.

4.3 Adversarial Examples

The discovery of adversarial examples [Szegedy et al., 2013] has been arguably the most severe setback regarding the trustworthiness of DL models. Adversarial examples are artificially modified images that are perceptually almost identical to their original counterpart but can cause DL models to misclassify them with high confidence. For instance, the adversarial example in Figure 4.4 (central panel) is very similar to the original image (left panel), but while the latter is correctly classified ('tree frog') by a ResNet50 model, the predicted class for the former is 'pizza' with confidence 99.9%. It is evident that the existence of adversarial examples poses huge problems to the reliability of models and represents a substantial difference between the performance of human beings and machines regarding the robustness to imperceptible perturbations. Indeed, a human observer would never be fooled by slight changes in the input image that do not alter the semantics of its content.

Over the last decade, several methods to craft adversarial examples have been proposed. In [Goodfellow et al., 2014], the authors introduce the Fast Gradient Sign Method (FGSM), which leverages the gradient sign of the loss function with respect to the input as follows

$$\mathbf{x}' = \mathbf{x} + \varepsilon \cdot \mathrm{sign}(\nabla_{\mathbf{x}} \mathcal{L}(f_{\boldsymbol{\vartheta}}(\mathbf{x}, y)), \tag{4.10}$$

where the norm of the perturbation $\delta = \varepsilon \cdot \mathrm{sign}(\nabla_{\mathbf{x}} \mathcal{L}(f_{\boldsymbol{\vartheta}}(\mathbf{x}, y))$ is bounded to guarantee that the adversarial is similar to the original image. Other popular adversarial attacks are: DeepFool [Moosavi-Dezfooli et al., 2016], a simple and effective algorithm based on an iterative linearization of the classifier to generate minimal perturbations that are sufficient to change the prediction of a classifier; the one-pixel attack [Su et al., 2019], a black-box attack that perturbs a single pixel in the input image using differential evolution; universal adversarial perturbations [Moosavi-Dezfooli et al., 2017], i.e., small universal perturbations

Figure 4.4: Adversarial example for an ImageNet test image (index 1551). Original image (left); targeted ℓ_2-bounded adversarial example crafted with PGD, $\varepsilon_{te} = 3$, 20 steps predicted as 'pizza' with confidence 99.9% (center); magnified distance between the original image and the corresponding adversarial example background (right).

vectors that can impair the classification of a great portion of images in a given set; adversarial patch [Brown et al., 2017], that constructs image-independent adversarial patches to make classifiers output a target class. For a complete survey on the topic, please refer to [Chakraborty et al., 2018, Akhtar and Mian, 2018].

4.4 Adversarial Training

Adversarial Training (AT) is arguably the most prominent method to make models robust to adversarial examples. In their seminal paper [Madry et al., 2017], the authors study the problem of adversarial robustness through the lens of robust optimization and propose a comprehensive theoretical framework that encompasses both attacks and defenses. The first step is to define a set of allowed perturbations $\mathcal{M}$, which is usually an ℓ_2-ball or ℓ_∞-ball of radius ε around the input image $\mathbf{x}$ for the problems of interest in this thesis. Then, instead of computing the loss directly on original training samples, we let the adversary perturb them first, leading to the following saddle problem:

$$\min_{\boldsymbol{\vartheta}} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\big[\max_{\delta\in\mathcal{M}} \mathcal{L}(f_{\boldsymbol{\vartheta}}(\mathbf{x},y)\big]. \tag{4.11}$$

The saddle problem in Equation (4.11) can be viewed as the composition of an inner maximization problem, aimed at finding an adversarial version of the original input so that the loss is maximized, and an outer minimization problem, aimed at finding a configuration of the model's parameters that correctly classifies the adversarial example. As a result, solving the saddle problem in Equation (4.11) leads to a model that is robust against adversarial perturbations within the set $\mathcal{M}$.

Notice that the FGSM attack given in Equation (4.10) can be interpreted as a single-step procedure to solve the inner maximization problem in Equation (4.11).

According to [Madry et al., 2017], a more powerful adversary is represented by a multi-step variant which is Projected Gradient Descent (PGD) on the negative loss function. Given a norm $\|\cdot\|_p$ and a perturbation budget $\varepsilon > 0$, let B_ε be the ℓ_p-ball of radius ε centered at $\mathbf{x}$. A PGD-based adversarial example for the original input $\mathbf{x}$ is initialized at a random point in B_ε and iteratively updated (for a given number of steps) according to the following rule:

$$\mathbf{x}'_{t+1} = \Pi_{B_\varepsilon}\left[\mathbf{x}'_t + \alpha \cdot \operatorname{sign}(\nabla_{\mathbf{x}} \mathcal{L}(f_{\boldsymbol{\vartheta}}(\mathbf{x}', y)))\right], \tag{4.12}$$

where α is the attack step-size, and $\Pi_{B_\varepsilon}(\cdot)$ projects an input onto B_ε.

The robust models considered in Chapters 5 and 6 are trained/evaluated on attacks bounded in either ℓ_2- or ℓ_∞-norm. We denote with ε_{tr} the perturbation budget used during the training and with ε_{te} the perturbation budget to craft adversarial examples for robustness evaluation.

Chapter 5

Properties and Limitations of Adversarial Training

5.1 Introduction

DNN architectures have enjoyed massive success in image classification tasks [LeCun et al., 2015, Krizhevsky et al., 2009a, Krizhevsky et al., 2012]. However, since the discovery of adversarial examples [Szegedy et al., 2013, Goodfellow et al., 2014], the trustworthiness of their predictions has begun to be questioned. To date, many defenses have been proposed to ameliorate the robustness against adversarial attacks, but only a few proved to be still effective after improvement of existing attacks [Athalye et al., 2018,Tramer et al., 2020]. As mentioned in Section 4.4, AT [Madry et al., 2017] is arguably the most prominent amongst such successful defenses, and has become a cornerstone and the primary benchmark for robustness in DNNs. As such, countless efforts have been made toward understanding the principles underlying the tremendous success of AT. Nonetheless, this long-standing problem is far from being solved. In this Chapter, we build on well-known facts about adversarially-trained models[1] - such as the accuracy-robustness trade-off [Tsipras et al., 2018, Yang et al., 2020] or the shape bias [Zhang and Zhu, 2019] - and unveil previously unnoticed behaviors that improve our understanding of AT. The aim of the work outlined in this Chapter is twofold. Firstly, to challenge some common beliefs on robust models as a first step toward establishing more sound empirical evidence of their limitations. Secondly, to reveal properties of adversarially-trained models that could serve as a base for future research to achieve robustness without AT. Our analysis focuses on CNN architectures commonly adopted in image classification tasks. Based on our findings, we conclude that current CNNs trained with AT are not the best choice for robust classification in Computer Vision. Indeed, the experiments described in this Chapter empirically demonstrate that adversarially-trained

[1]The expressions *robust models* and *adversarially-trained models* will be used interchangeably hereinafter.

CNNs do not exploit efficiently the model capacity and that the simplicity biases induced by AT may lead to undesired behaviors. We first overview prior works on AT and biases of CNNs in Section 5.2, then describe the motivations behind our analyses in Section 5.3. In Section 5.4 we summarize our contributions, while in Section 5.5 we introduce the methodological tools needed for our experiments. Experimental results are discussed in Section 5.6. In Section 5.7, we recap our findings and point to some interesting directions for future works.

5.2 Related Work

Adversarial Training It is widely accepted that AT represents the current state-of-the-art in defending against adversarial attacks. Indeed, while many other defenses have been proposed [Guo et al., 2017, Dhillon et al., 2018, Xie et al., 2017, Song et al., 2017], they have been systematically evaded by refined attacks [Athalye et al., 2018, Tramer et al., 2020]. Being the de-facto standard for enforcing robustness, AT has attracted a great deal of research interest. Many studies analyzed structural properties [Ilyas et al., 2019, Engstrom et al., 2019, Shaham et al., 2018] and peculiar behaviors [Salman et al., 2020a, Terzi et al., 2020, Utrera et al., 2020, Yang et al., 2020] of adversarially-trained models. In particular, a well-known problem affecting robust models is the so-called accuracy-robustness trade-off [Tsipras et al., 2018, Yang et al., 2020]. Initially, this tension was thought to be inherent when training robust models [Tsipras et al., 2018], but subsequent analysis [Yang et al., 2020] proved that commonly used image datasets are separable, conjecturing that a perfectly robust and accurate classifier can, in principle, exist. Other studies focused on improving the performance of AT. In [Xie et al., 2020], the authors propose to use distinct batch norm layers for clean and adversarial examples during training to improve natural accuracy. In [Zhang et al., 2019a], a novel formulation of adversarial defense, dubbed TRADES, is proposed. The method is based on the optimization of a loss taking into account both natural accuracy and adversarial robustness. Finally, another line of research is devoted to analyzing and estimating the Lipschitz constant of DNNs, as a guarantee of stability and robustness to be enforced during training [Scaman and Virmaux, 2018, Huang et al., 2021, Liang and Huang, 2020]. The present work complements prior studies on the properties of robust models and provides arguments to resolve contrasting results found in the literature.

Biases of CNNs In [Ding et al., 2019], the authors analyze the impact of semantics-preserving transformations of the input data distributions on clean accuracy and adversarial robustness. The experiments show that robust accuracy under PGD training is much more sensitive than clean accuracy under standard training to the differences in input data distribution. In [De and Pedersen, 2021], the impact on natural accuracy of color distortions is assessed. Specifically,

the authors propose a variant of the ImageNet dataset where a set of color distortions is applied to the original images. The aim of [Hermann et al., 2020] is to investigate the source of the texture bias in models trained on ImageNet. It shows that random-crop augmentation biases the models toward texture and proposes more naturalistic forms of data augmentation as a simple way to mitigate texture bias. Surprisingly, they could extract and decode shape information from hidden layers with high accuracy. This suggests that classification layers might play an essential role in removing shape information. In [Zhang and Zhu, 2019], the authors exploit saliency maps to interpret the inner workings of adversarially-trained models and compare them to standard models. They also evaluate these models on distorted test sets preserving either shape or textures and verify that adversarially-trained models rely more on global features such as shape and edges. They finally show that standard models are biased toward textures, as previously observed by [Geirhos et al., 2018]. Along these lines, our experiments focus on the simplicity biases of robust models. Unlike previous studies, which primarily focused on the bias toward shape and textures, our analysis addresses the bias toward color - decoupled from shape and textures.

5.3 Motivations

The experiments described in Section 5.6 are aimed at highlighting interesting properties and potentially harmful limitations of robust models. This is done by characterizing their behavior along four dimensions: i) analysis of the sparsity and frequency of activations of their feature maps; ii) analysis of the redundancy of their feature maps; iii) analysis of the inherent robustness of their latent representations; iv) assessment of the bias toward color and consequences on their predictive performance. In the following, we summarize the motivations behind each of the abovementioned directions of scrutiny.

Densely Active Feature Maps Many works discussed the interplay between model sparsity and robustness to adversarial attacks [Xiao et al., 2018, Wong and Kolter, 2018, Guo et al., 2018, Ye et al., 2019, Özdenizci and Legenstein, 2021, Wang et al., 2018a], often leading to opposite conclusions [Guo et al., 2018]. Since the parameters of a model provide a convenient static representation of the task, these studies primarily focus on weight sparsity. Conversely, we address the sparsity of activations. Indeed, the interaction between the input and the model - rather than a structural representation of the latter - is the critical factor to be investigated to characterize the model's behavior.

Feature Maps Redundancy We focus on an underexplored direction to understand how adversarially-trained CNNs work internally, namely the redundancy of feature maps. We draw inspiration from three facts: i) redundant signals are widely exploited in contexts where robustness to noise is a primary concern,

e.g., in the theory of frames [Kovacevic and Chebira, 2008] or in error-correcting codes [Guruswami and Rudra, 2008]; ii) the trade-off between accuracy and robustness in DNNs is conjectured to be due to limitations of existing architectures and training methods [Yang et al., 2020], rather than class separability under adversarial attacks [Tsipras et al., 2018]; iii) AT benefits more from larger capacity than standard training [Madry et al., 2017,Xie and Yuille, 2019]. Higher internal redundancy (inspired by point i)) in adversarially-trained models would provide a valid argument to justify empirical observations ii) and iii). Indeed, if AT used feature maps redundancy to average out adversarial noise, that would be a limitation leading to reduced model capacity and potentially contributing to the drop in natural accuracy (point ii)). This could also be one of the reasons why robust models benefit more from larger capacity than natural models (point iii)): increasing the width or depth of CNNs consists in increasing the number of layers and the number of feature maps in convolutional layers, respectively, compensating for the reduced capacity in robust models with redundant feature maps.

Latent Space A recent work [Shafahi et al., 2019] demonstrated that robustness can be preserved in transfer learning settings, where the feature extractor - trained on the source domain - is kept frozen and the linear classifier is retrained on natural examples from the target domain. The authors also showed that retraining the linear classifier on natural examples from the source domain does not affect robust accuracy. However, this behavior has been assessed for a single model, and whether this claim can be generalized should be investigated. This is the goal of our experiments, where we analyze the predictive power and robustness of latent representations in adversarially-trained models. We draw inspiration from the surprising fact that representations in robust models can be inverted [Engstrom et al., 2019]. This property might appear incompatible with the accuracy-robustness trade-off observed in practice: if the representation is invertible, then information about the data should be preserved in the latent space, and the natural accuracy of robust models should not drop. In [Terzi et al., 2020], the authors resolve such discrepancy by proving that AT preserves information about the data, but the information that is *accessible* to the classifier does not contain all the details about the input. Our results corroborate this finding and show that latent representations of adversarially-trained models offer varying levels of robust and natural accuracy, dependent on the feature combinations selected by the classifier.

Color Bias It is a known fact that adversarially-trained models are more biased toward simple features - such as shape - rather than textures [Utrera et al., 2020, Chen et al., 2020a]. While these so-called *simplicity biases* are typically welcomed since they help improve robustness, they could be harmful in other contexts (as we shall show in Section 5.6). In our experiments, we

study the color bias of adversarially-trained models independently from the shape and texture information, a property that has been hitherto overlooked and not adequately analyzed. Indeed, most of the studies in the literature have been focusing on the texture-shape dichotomy [Geirhos et al., 2018, Utrera et al., 2020, Chen et al., 2020a].

5.4 Contributions

The contributions of the work outlined in this Chapter can be summarized as follows:

- We show that adversarially-trained models have a greater number of spatially dense feature maps that activate for all data points in the dataset compared to natural models. This always-active subnetwork decreases the model expressivity.
- We show that feature maps in adversarially-trained models are more redundant than in natural models, thus reducing the effective number of active channels in hidden layers.
- We show that the latent space of adversarially-trained models offers representations with varying levels of robust and natural accuracy. Specifically, by retraining the classifier on natural images, the natural accuracy can be significantly improved at the price of a decrease in robustness. This demonstrates that AT spends capacity to preserve extra information about the input that is then ignored by the robust classifier.
- We assess the color bias of adversarially-trained models by decoupling the color information from the texture and shape information. We also point out subtle failure modes that may undermine the deployment of robust models in practice.

5.5 Methods

Before describing the analytical tools employed in our experiments, we introduce the necessary notation. Let us partition the feature extractor f of a generic CNN in *blocks* along the depth dimension and denote with $\mathbf{x}^k$ the activations (corresponding to input image $\mathbf{x}$) at the output of the k-th block. The definition of *block* depends on the architecture. By way of example, for models in the ResNet family [He et al., 2016a, Zagoruyko and Komodakis, 2016] we collect activations at the output of each ResNet block, any of whom is composed of multiple sequences of the form convolutional layer - batch normalization layer - ReLU activation function. In general, $\mathbf{x}^k$ is a 3-D tensor with dimension $C_k \times H_k \times W_k$, where C_k, H_k and W_k are the number of channels, feature maps

height and width at the output of the k-th block. For our purposes, we consider its 2-D version $\tilde{\mathbf{x}}^k \in \mathbb{R}^{C_k \times (H_k \times W_k)}$ with vectorized feature maps. Let $\tilde{\mathbf{x}}^k[i]$ (or equivalently, $\mathbf{x}^k[i]$) represent the i-th feature map at the output of the k-th block, with $i \in \{1, \ldots, C_k\}$.

Definition 5.5.1. We say that a feature map is *active* if at least one of its activations is greater than 0.

Definition 5.5.2. We say that a feature map is *densely active at level* τ_{dens} if at least τ_{dens} of its activations are greater than 0, for a given density threshold $\tau_{dens} \in [0, 1]$.

Definition 5.5.3. Let $\mathcal{I}_+^k$ be the subset of active feature maps indices at the output of the k-th block. We define the tensor of the active feature maps at the output of the k-th block as $\tilde{\mathbf{x}}_+^k \stackrel{\text{def}}{=} \tilde{\mathbf{x}}^k[\mathcal{I}_+^k]$.

Notice that $\tilde{\mathbf{x}}_+^k$ has dimension $|\mathcal{I}_+^k| \times H_k \times W_k$, where $|\mathcal{I}_+^k|$ is the cardinality of $\mathcal{I}_+^k$ and $|\mathcal{I}_+^k| \leq C_k$.

Measuring Densely Active Feature Maps Sparsity of activations can be assessed by counting the number of densely active feature maps for each data point in the dataset and each block in the model. We combine this information with the frequency of activation of densely active feature maps across the dataset to measure the model's expressivity.

Measuring Feature Maps Redundancy Given a block index k, the starting point is the computation of the cosine similarity matrix S_+^k between active feature maps

$$S_+^k(i,j) = \frac{< \tilde{\mathbf{x}}_+^k[i], \tilde{\mathbf{x}}_+^k[j] >}{\|\tilde{\mathbf{x}}_+^k[i]\| \cdot \|\tilde{\mathbf{x}}_+^k[j]\|} \tag{5.1}$$

where $< \cdot, \cdot >$ represents the inner product. The next step consists in clustering together active feature maps whose cosine similarity is above a given threshold τ_{sim}. As a result, we obtain a set of clusters $\mathcal{C}_1^k, \ldots, \mathcal{C}_{n_k}^k$, with $n_k \leq C_k$. Completely uncorrelated feature maps would produce $n_k = C_k$ clusters with one element each, while high correlation ($> \tau_{sim}$) amongst some of the feature maps would result in a more limited number of clusters, some of whom would have cardinality > 1. The number of clusters at the output of a given block can be thought of as the *effective number of active channels* at the output of the block.

Definition 5.5.4. We say that an active feature map is *redundant* if it belongs to a cluster with cardinality > 1.

According to the definitions given above, for the k-th block, we analyze feature maps redundancy by counting the number of redundant feature maps C_k^R.

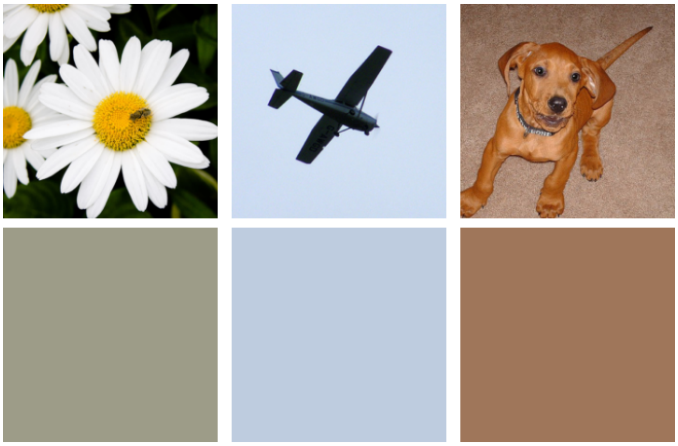

Figure 5.1: Original images (top) and their pixel-averaged counterparts (bottom) from the ImageNet test set.

Measuring Inherent Robustness of Latent Features The setup for this batch of experiments is as follows. We consider a model $f \circ g$ pretrained in adversarial settings. We keep the feature extractor f fixed and retrain from scratch the linear classifier g *on natural examples*. In other words, we re-initialize and then optimize $\boldsymbol{\vartheta}_g$, while $\boldsymbol{\vartheta}_f$ is kept unchanged. We find that this simple procedure can lead to non-trivial improvements in natural accuracy for robust models, but this occurs at the price of a significant decrease in robustness.

Measuring Color Bias We assess the color bias of adversarially-trained and natural models with three experiments.

Experiment #1. We measure the natural accuracy on pixel-averaged images. In other words, for each image in the test set, we consider its texture-less and shape-less monochromatic counterpart, where each pixel assumes the average value of all pixels (examples in Figure 5.1).

Experiment #2. We consider transformed images where a colored contour of varying thickness is applied. We measure the natural accuracy drop with respect to the evaluation on clean images. The colored contours can be red, green, blue, or white. For the CIFAR-10 dataset, we consider contours of thickness 1, 2, 3, or 4 pixels. For the ImageNet dataset, we consider contours of thickness 1, 5, 10, 15, and 20 pixels (examples in Figure 5.2). Notice that the semantic content of images is minimally affected by the added contour, and humans would not be induced to make wrong predictions because of the frame element. Unlike humans, the performance of adversarially-trained CNNs can be significantly impaired in this scenario. We tested the effect of colored contours just as a prime example of how a content-preserving alteration of the original image - leveraging

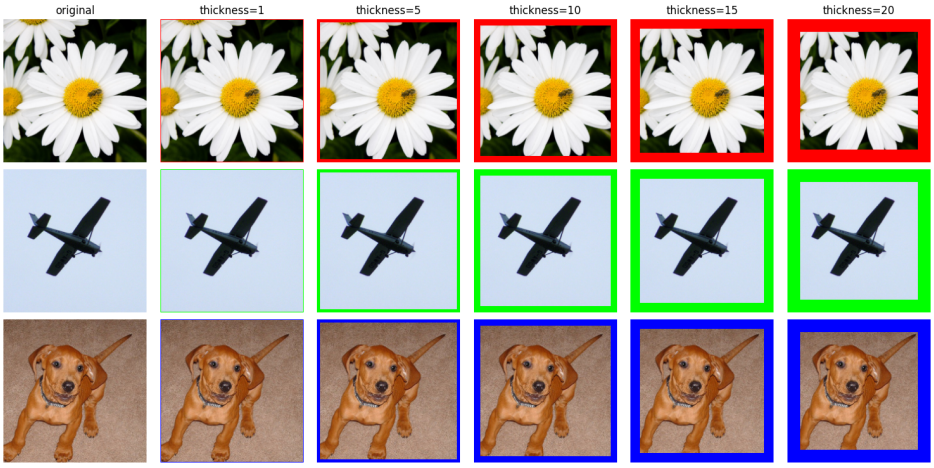

Figure 5.2: ImageNet images with colored contours.

the simplicity bias induced by AT - can lead to dramatic changes in the behavior of these models. Hence, the existence of even more subtle case studies is not excluded.

Experiment #3. Along the lines of experiment #1, we investigate the relevance of the foreground and background average colors. To do so, we leverage the ImageNet-9 dataset introduced in [Xiao et al., 2020], for which fine-grained foreground masks are available. The test set used for our analysis consists of 450 images per class. To evaluate the accuracy on ImageNet-9 of models pretrained on ImageNet, we map the predictions from the 1000 ImageNet classes into the 9 coarse-grained ImageNet-9 classes. We exploit the foreground mask to extract the average color of the foreground and background for each test image. Then, we apply transformations that result in monochromatic variants of each image where the color is given by either i) the average value of all pixels ('AVG'), which is equivalent to what is done in experiment #1; ii) the average value of pixels in the background ('AVG BG'); iii) the average value of pixels in the foreground ('AVG FG'). Differently from [Xiao et al., 2020], we focus on the average color information in the background and foreground and decouple it from other informative signals (e.g., shape and texture). Additionally, we consider a transformation that couples the shape and color signals ('SHAPE+COLOR'), where pixels in the foreground (background) share their corresponding average color. This allows the emergence of the shape of the main object. Examples of transformed ImageNet-9 images are shown in Figure 5.3.

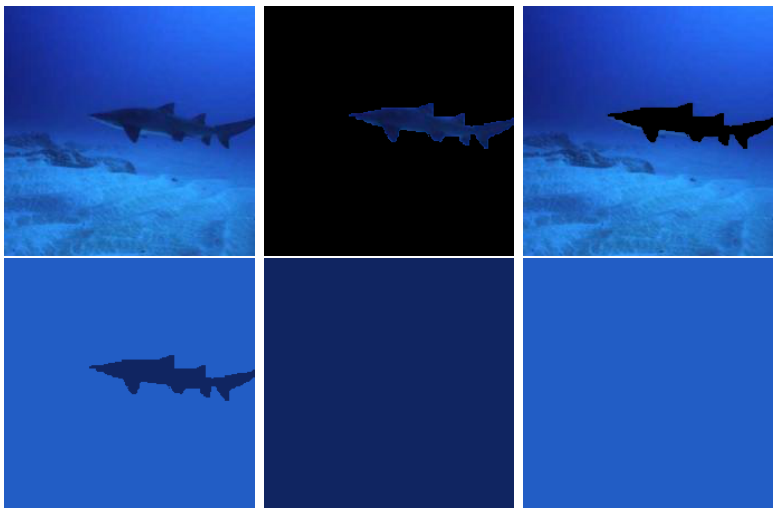

Figure 5.3: Transformations on ImageNet-9 images. Top row: original image (left), original foreground (center), original background (right). Bottom row: SHAPE+COLOR (left), AVG FG (center), AVG BG (right).

5.6 Experimental Results

Models and datasets We conduct our analyses on three public datasets for which models are easy to train (from a computational point of view), or pretrained models are available.: CIFAR-10 [Krizhevsky et al., 2009b], CIFAR-100 [Krizhevsky et al., 2009b], and ImageNet [Deng et al., 2009].

For CIFAR-10 and CIFAR-100, we consider ResNet18 models trained with $\varepsilon_{tr} \in \{0, 0.5, 1, 2, 3, 4\}$ and ℓ_2-norm. The case $\varepsilon_{tr} = 0$ represents standard training. All CIFAR-10 and CIFAR-100 models are trained for 150 epochs, batch size 128, weight decay $5 \cdot 10^{-4}$, initial learning rate $\eta = 0.1$ and a drop of η by a factor 10 every 50 epochs. For robust models trained with AT, adversarial examples are crafted with 7 PGD steps and attack step size $\alpha = 1$.

For ImageNet, we consider ResNet18, ResNet50 [He et al., 2016a], Wide ResNet50 $\times 2$, Wide ResNet50 $\times 4$ [Zagoruyko and Komodakis, 2016] models trained with $\varepsilon_{tr} \in \{0, 0.5, 3, 5\}$ for ℓ_2-norm and with $\varepsilon_{tr} \in \{0, 0.5, 1, 2, 4, 8\}$ for ℓ_∞-norm. Moreover, we consider VGG16 [Simonyan and Zisserman, 2014] models trained with $\varepsilon_{tr} \in \{0, 3\}$ and ℓ_2-norm. All ImageNet models are pretrained models retrieved from [Salman et al., 2020a]. Please refer to [Salman et al., 2020a] for further details.

Some experiments are based on a subset of the available models for ease of visualization and/or computational efficiency. Results on robust models trained with ℓ_∞-norm, along with results with different values of τ_{dens} and τ_{sim} are presented in Appendix A.

5.6.1 Densely Active Feature Maps

We analyze densely active feature maps on natural images from the test set. The reasons why we focus on the original images rather than on adversarial examples are: i) the higher computational cost required to craft adversarial examples; ii) the fact that AT produces feature maps that are stable under adversarial attacks so that the differences in the activations between natural and adversarial images are negligible. To provide experimental evidence on point ii), we check the number of densely active feature maps for adversarial examples (crafted with the same specifications that were used during training) instead of natural images in Figure 5.6. The results are almost identical in the two cases. In Figures 5.4 and 5.5 is shown the number of densely active feature maps at level $\tau_{dens} = 0.95$ for ImageNet models and CIFAR-10/CIFAR-100 models, respectively, where N_k is the number of always densely active feature maps for the k-th block. Similar results hold for other large enough values of τ_{dens} (see Appendix A.1). By examining how the number of densely active feature maps varies under different training conditions, a remarkable result emerges: compared to natural models, robust models have a higher number of feature maps that are *spatially dense* and *active for all samples in the dataset*. This holds true across *all architectures and datasets* analyzed in this work. Notice that in most cases, the larger ε_{tr}, the higher the value of N_k. This means that, as the robust accuracy of the model increases, spatially dense feature maps in intermediate blocks make less use of ReLUs. The fact that these feature maps are spatially dense and active for all data points in the dataset suggests that they convey information that is always important/necessary. The model is partitioned into an always active subnetwork - that gets larger as the robustness of the model increases - and another subnetwork whose feature maps activate only for a subset of the dataset. This partition recalls the structure of gating mechanisms where one branch controls the flow of information in another branch, like the Gating Units proposed in [Liu et al., 2021]. These results suggest that AT induces, as ε_{tr} increases, a larger portion of feature maps to have a qualitatively different behavior compared to the rest of the model, reducing its expressivity.

5.6.2 Feature Maps Redundancy

Similarly to the experiments on densely active feature maps, we compute the number of redundant feature maps on natural images from the test set. Almost identical results can be obtained when considering adversarial examples instead of natural images (see Figure 5.9). In Figures 5.7 and 5.8 is shown the number of redundant feature maps C_k^R across layers for different architectures and training conditions. We set $\tau_{sim} = 0.95$ to get clusters whose elements are highly correlated to each other, but similar results hold for other values of τ_{sim} (see Appendix A.2). Notice that there exists a strong correlation between the robust accuracy of adversarially-trained CNNs and the redundancy of their active

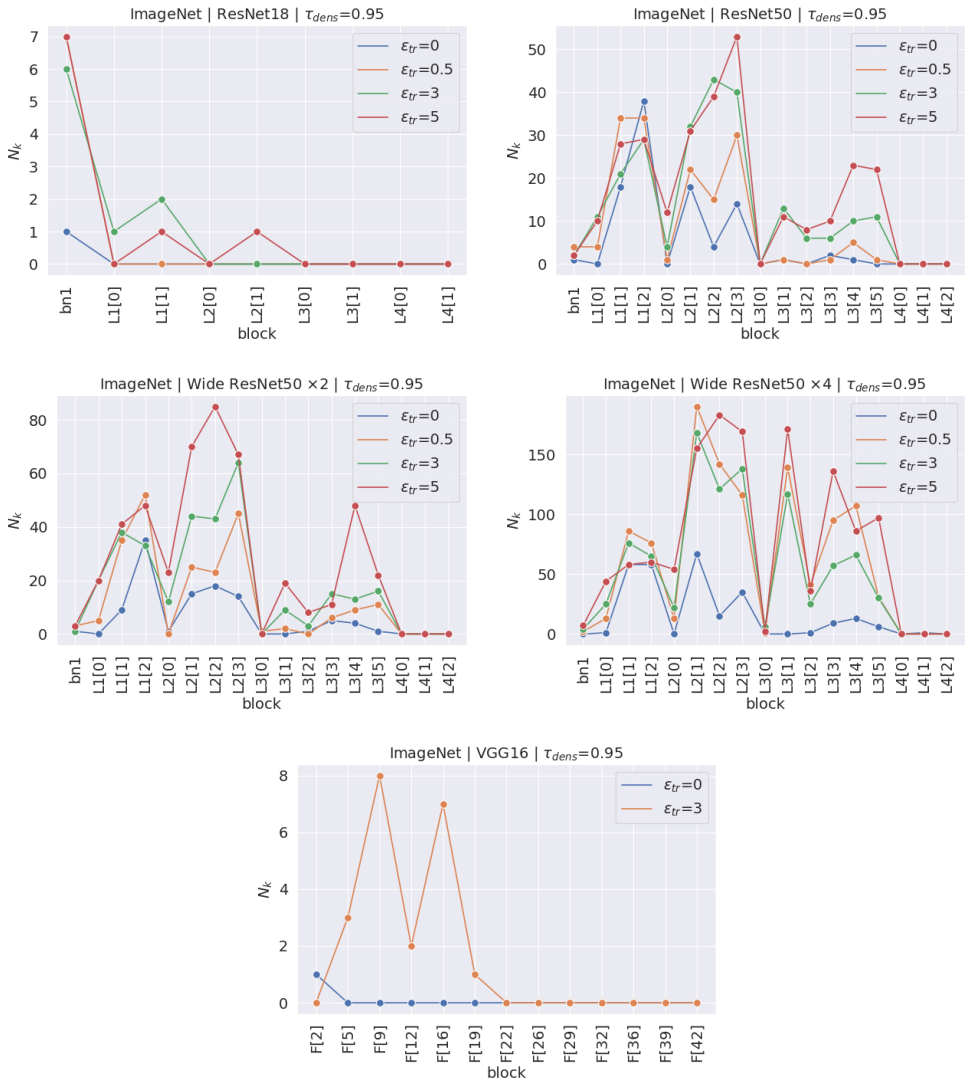

Figure 5.4: Number of always densely active feature maps (at level $\tau_{dens} = 0.95$) for ImageNet models. Robust models are trained with ℓ_2-norm.

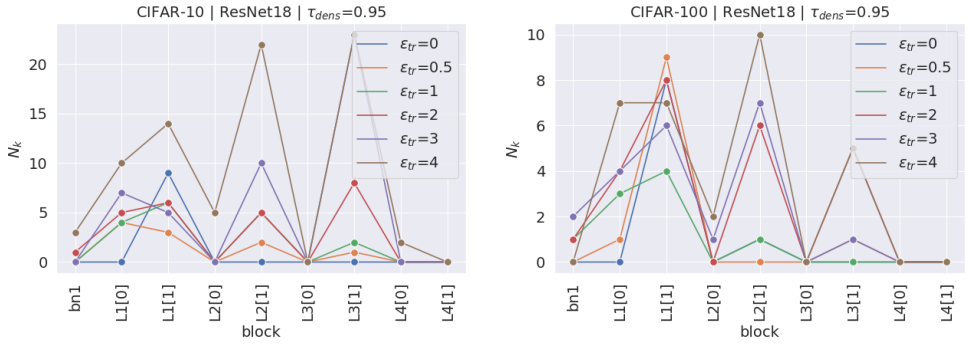

Figure 5.5: Number of always densely active feature maps (at level $\tau_{dens} = 0.95$) for CIFAR-10 and CIFAR-100 models. Robust models are trained with ℓ_2-norm.

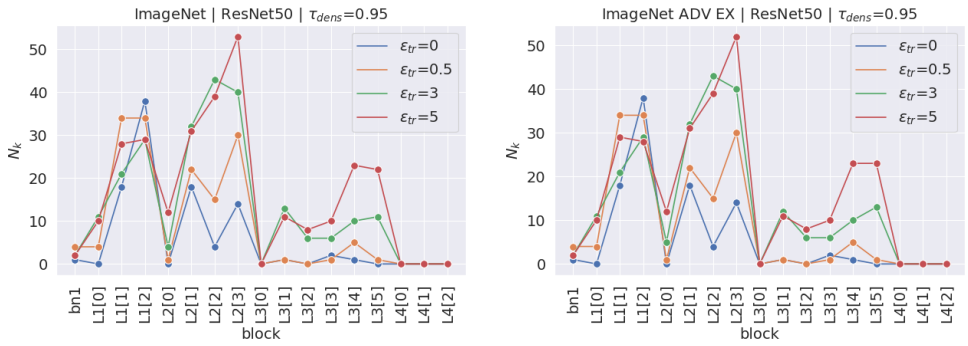

Figure 5.6: Number of always densely active feature maps (at level $\tau_{dens} = 0.95$) for ResNet50 on ImageNet: natural images (left) and adversarial examples (right). Robust models are trained with ℓ_2-norm.

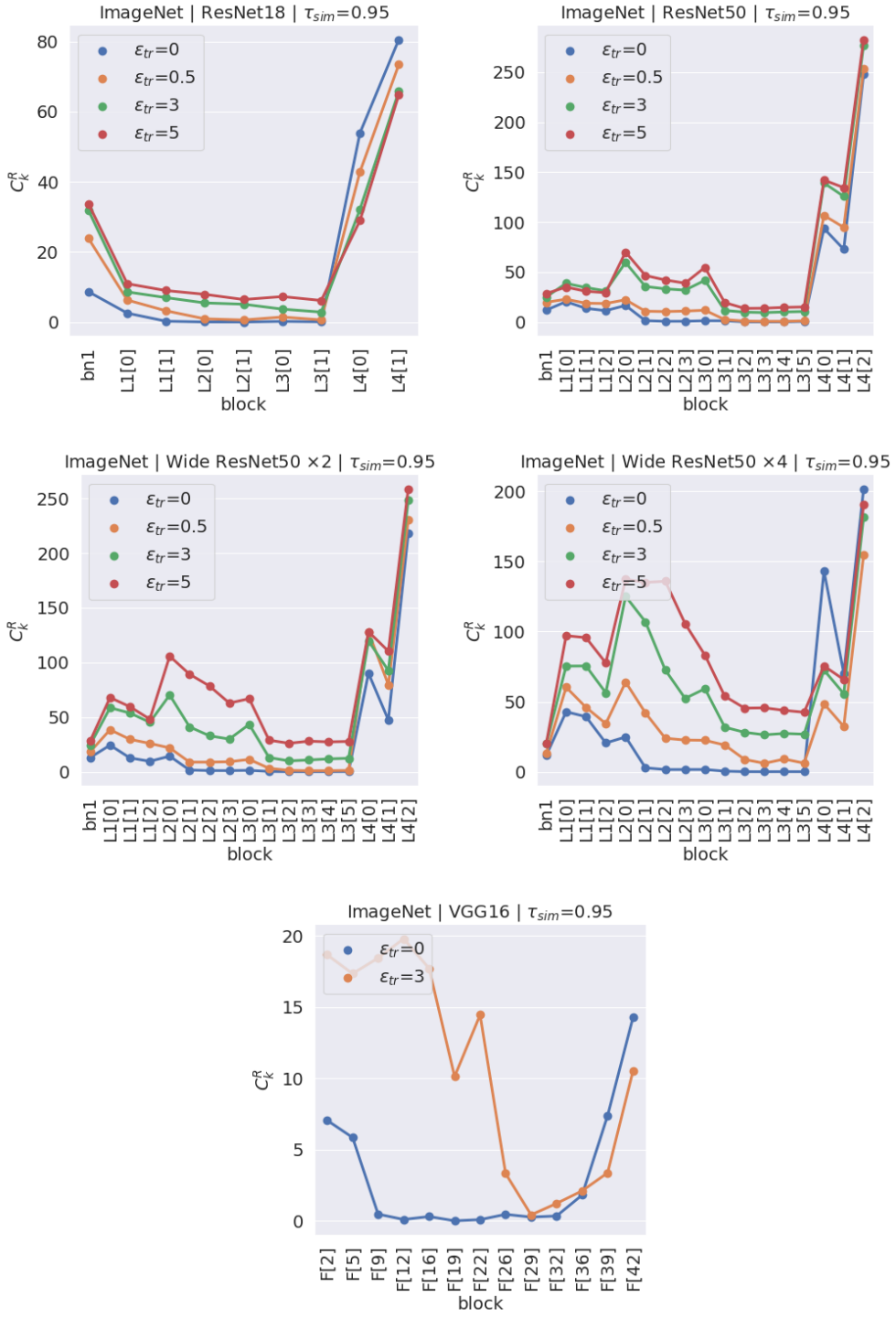

Figure 5.7: Number of redundant feature maps (with $\tau_{sim} = 0.95$) for ImageNet models. Robust models are trained with ℓ_2-norm. Values are averaged over 5000 images randomly sampled from the test set.

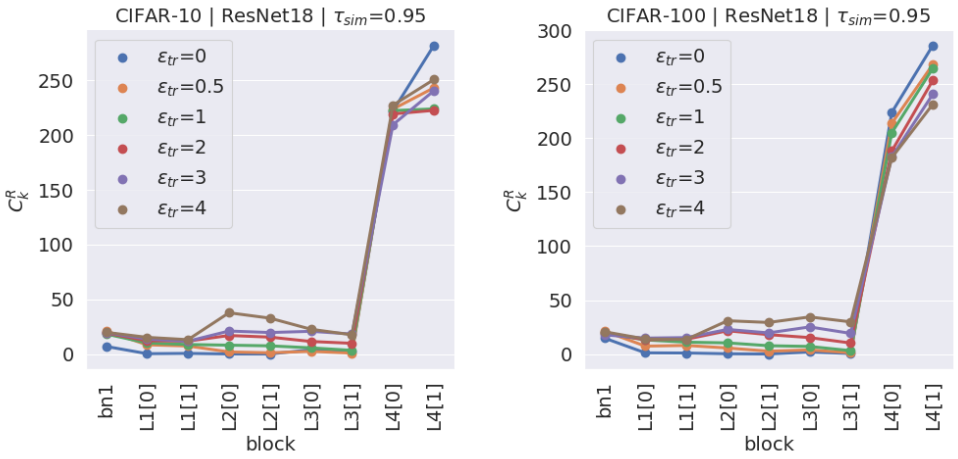

Figure 5.8: Number of redundant feature maps (with $\tau_{sim} = 0.95$) for CIFAR-10 and CIFAR-100 models. Robust models are trained with ℓ_2-norm.

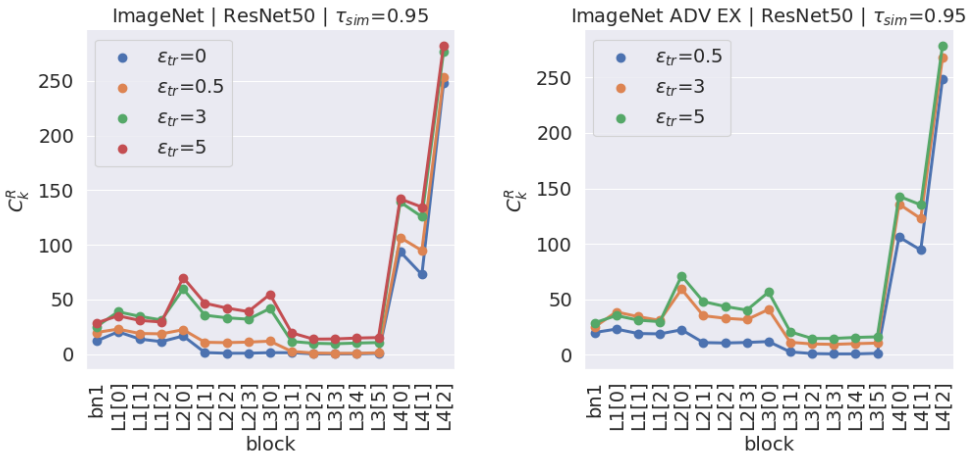

Figure 5.9: Number of redundant feature maps ($\tau_{sim} = 0.95$) for ResNet50 on ImageNet: natural images (left) and adversarial examples (right). Robust models are trained with ℓ_2-norm. Values are averaged over 5000 images randomly sampled from the test set.

feature maps. This effect is most prominent for models with higher capacity. For example, for ImageNet, Wide ResNet50 $\times 4$ has more redundant feature maps than Wide ResNet50 $\times 2$, which in turn has more than ResNet50[2]. The only exception is represented by the deeper layers, whose behavior is substantially different if compared to others. This is not surprising, as these layers are deputed to combine features to fit and/or memorize classes, as demonstrated by [Stephenson et al., 2021]. It is worth mentioning that wide natural models exhibit a larger number of redundant feature maps compared to their thinner counterparts. This is consistent with the results in [Casper et al., 2019], where the existence of so-called *redundant units* is proved and leveraged to explain implicit regularization in wide (natural) models.

Our findings suggest a novel direction for the investigation of the mechanism through which local robustness may be implemented by adversarially-trained CNNs, namely a coupling between feature maps. Notice that in [Liang and Huang, 2020], the authors propose - although for a very simple DNN - a coupling between subsequent layers as a viable solution for achieving small Lipschitz constants, regardless of their norm. The results presented above suggest that AT might exploit similar schemes. Since in robust models the activation of a specific feature map implies the activation of slightly different copies of it more frequently than in natural models, the model capacity is exploited less efficiently.

5.6.3 Latent Features

We retrain linear classifiers $\tilde{g}$ for 15 epochs on natural examples from the source domain on which the pretrained model $f \circ g$ was trained. Results for ImageNet, and CIFAR-10/CIFAR-100 are reported in Tables 5.1 and 5.2, respectively. This procedure leads to a clear improvement in natural accuracy with respect to the original classifier g, unveiling the availability of predictive latent representations that were not fully exploited. On the other hand, such enhanced predictive power implies a drastic drop in robustness. We acknowledge a correlation between the value of ε_{tr} and the entity of this behavior - the larger ε_{tr}, the larger the increase (decrease) in natural (robust) accuracy. As anticipated in Section 5.5, our results are in line with [Terzi et al., 2020]: the information about the input is preserved in the latent space, but it is not fully accessible for the robust classifier. This behavior would be expected if the latent features were given and a robust classifier was trained on top of them at a later stage. Indeed, a robust classifier would rightly discard features that are less robust - although potentially more predictive - since robustness is the objective being explicitly maximized. The surprising fact is that latent features are not given but learned during training, and robust models spend a fraction of their capacity to learn features that are then ignored by the classifier. A natural question arises: why are these features present if they are not exploited by the

[2]Notice that wide models have the same number of output channels as their thinner counterparts at the output of elementary blocks, but they have more channels *within* the block.

Table 5.1: Natural and robust accuracy of retrained classifiers on ImageNet. Feature extractors are pretrained robust models trained with ℓ_2-norm. Robust accuracy is evaluated against adversarial attacks crafted with 20 PGD steps, attack step size 1, and $\varepsilon_{te} = \varepsilon_{tr}$.

Model	**Metric**	**Clf**	$\boldsymbol{\varepsilon_{tr} = 0.5}$	$\boldsymbol{\varepsilon_{tr} = 3}$	$\boldsymbol{\varepsilon_{tr} = 5}$
ResNet18	nat acc	orig	65.48	53.12	45.59
		retr	65.85	55.07	49.18
			(+0.37)	(+1.95)	(+3.59)
ResNet18	rob acc	orig	55.15	31.05	21.85
		retr	54.20	27.17	16.93
			(-0.95)	(-3.88)	(-4.92)
ResNet50	nat acc	orig	73.16	62.83	56.13
		retr	73.19	64.06	58.70
			(+0.03)	(+1.23)	(+2.57)
ResNet50	rob acc	orig	63.41	38.94	27.78
		retr	62.31	34.53	22.10
			(-1.10)	(-4.41)	(-5.68)
Wide ResNet50 ×2	nat acc	orig	75.11	66.90	60.94
		retr	74.87	67.48	62.73
			(-0.24)	(+0.58)	(+1.79)
Wide ResNet50 ×2	rob acc	orig	65.85	41.70	30.61
		retr	64.85	37.69	25.28
			(-1.00)	(-4.01)	(-5.33)

Table 5.2: Natural and robust accuracy of retrained classifiers on CIFAR-10 and CIFAR-100. Feature extractors are pretrained robust models trained with ℓ_2-norm. Robust accuracy is evaluated against adversarial attacks crafted with 20 PGD steps, attack step size 1, and $\varepsilon_{te} = \varepsilon_{tr}$.

Model	Metric	Clf	$\varepsilon_{tr} = 0.5$	$\varepsilon_{tr} = 1$	$\varepsilon_{tr} = 2$	$\varepsilon_{tr} = 3$	$\varepsilon_{tr} = 4$
CIFAR-10							
ResNet18	nat acc	orig	88.34	80.14	64.43	59.25	46.54
		retr	88.78	82.53	72.23	67.69	56.48
			(+0.44)	(+2.39)	(+7.8)	(+8.44)	(+9.94)
	rob acc	orig	68.30	51.44	33.38	22.85	16.26
		retr	67.16	46.77	20.36	8.71	4.87
			(-1.14)	(-4.67)	(-13.02)	(-14.14)	(-11.39)
CIFAR-100							
ResNet18	nat acc	orig	63.68	57.97	49.73	37.41	27.42
		retr	64.16	57.60	52.34	46.03	39.56
			(+0.48)	(-0.37)	(+2.61)	(+8.62)	(+12.14)
	rob acc	orig	36.07	22.50	13.07	9.54	6.35
		retr	36.67	21.44	8.16	4.05	1.67
			(+0.6)	(-1.06)	(-4.91)	(-5.49)	(-4.68)

classifier? One hypothesis is that they are an artifact of the training process. At the initial stage of the training, a weak attacker might not be able to provide informative adversarial examples, and the learning process may be irreversibly biased toward solutions that are not optimal for robust classification.

5.6.4 Color Bias

We start this batch of experiments by evaluating natural accuracy on the extreme case of pixel-averaged images, as discussed in Section 5.5. In Tables 5.3 and 5.4, we observe that robust models can achieve better performance than natural models - and better than random guessing - on images that contain no information other than the average color. To the best of our knowledge, this is the first experiment demonstrating the color bias of adversarially-trained models in conditions where the shape and texture signals are zeroed-out. It is not surprising that adversarially-trained models rely more than natural ones on global color information as it is a robust feature. Problems arise when models *trivially* depend on color, i.e., when performance can be impaired by color-based perturbations that do not compromise other robust features such as the shape. We demonstrate that this is the case for robust models with the experiment on colored contours. Notice that our choice to place the frame element at the image's border is dictated by the need to preserve the shape of the main object, leveraging the center bias over the location of objects. In Figures 5.10 and 5.11 is shown the average (computed over different colors) drop in natural accuracy with respect to clean images when a colored contour is added. The performance

Table 5.3: Natural accuracy on pixel-averaged images for ImageNet. Robust models are trained with ℓ_2-norm.

Model	$\varepsilon_{tr} = 0$	$\varepsilon_{tr} = 0.5$	$\varepsilon_{tr} = 3$	$\varepsilon_{tr} = 5$
ResNet18	0.292	0.302	0.346	0.456
ResNet50	0.264	0.340	0.370	0.428
Wide ResNet50 ×2	0.204	0.318	0.398	0.420
Wide ResNet50 ×4	0.184	0.312	0.446	0.404
VGG16	0.298	-	0.390	-

Table 5.4: Natural accuracy on pixel-averaged images for CIFAR-10 and CIFAR-100. Robust models are trained with ℓ_2-norm.

Model	$\varepsilon_{tr} = 0$	$\varepsilon_{tr} = 0.5$	$\varepsilon_{tr} = 1$	$\varepsilon_{tr} = 2$
CIFAR-10				
ResNet18	8.00	16.66	15.57	17.57
CIFAR-100				
ResNet18	1.08	1.18	1.40	2.76

of robust models is remarkably impaired, even with relatively thin contours. For example, a 5-pixel contour for ImageNet images amounts to less than 9% of the entire image yet causes a drop in accuracy that is more than doubled compared to natural models. Besides the larger average drop, adversarially-trained models also present higher variance across different colors, indicating instability under color-based transformations. These results prove that robust models, while more 'stable' than natural models from the perspective of adversarial noise, rely on a delicate balance based on summary statistics of the dataset, such as global color. Once a perturbation in this sense is introduced, their stability is compromised, and performance decrease catastrophically. We expand our analysis on the color bias with experiments on the ImageNet9 dataset, as described in Section 5.5. Results for robust models trained with ℓ_2-norm are listed in Table 5.5. It is evident that robust models rely more on the background color than the foreground color. This behavior is more pronounced for more robust models (i.e., models trained with larger values of ε_{tr}), providing more evidence of the trivial reliance on color information. The SHAPE+COLOR experiment complements the above analysis on the color bias and brings a new perspective on the known simplicity biases that characterize adversarially-trained models (i.e., shape and color). When only the color and shape signals are available, just a small fraction of the accuracy on the original images is covered. This proves that known simplicity biases alone might not play a major role in the predictive power of robust models. Further investigations in this direction may

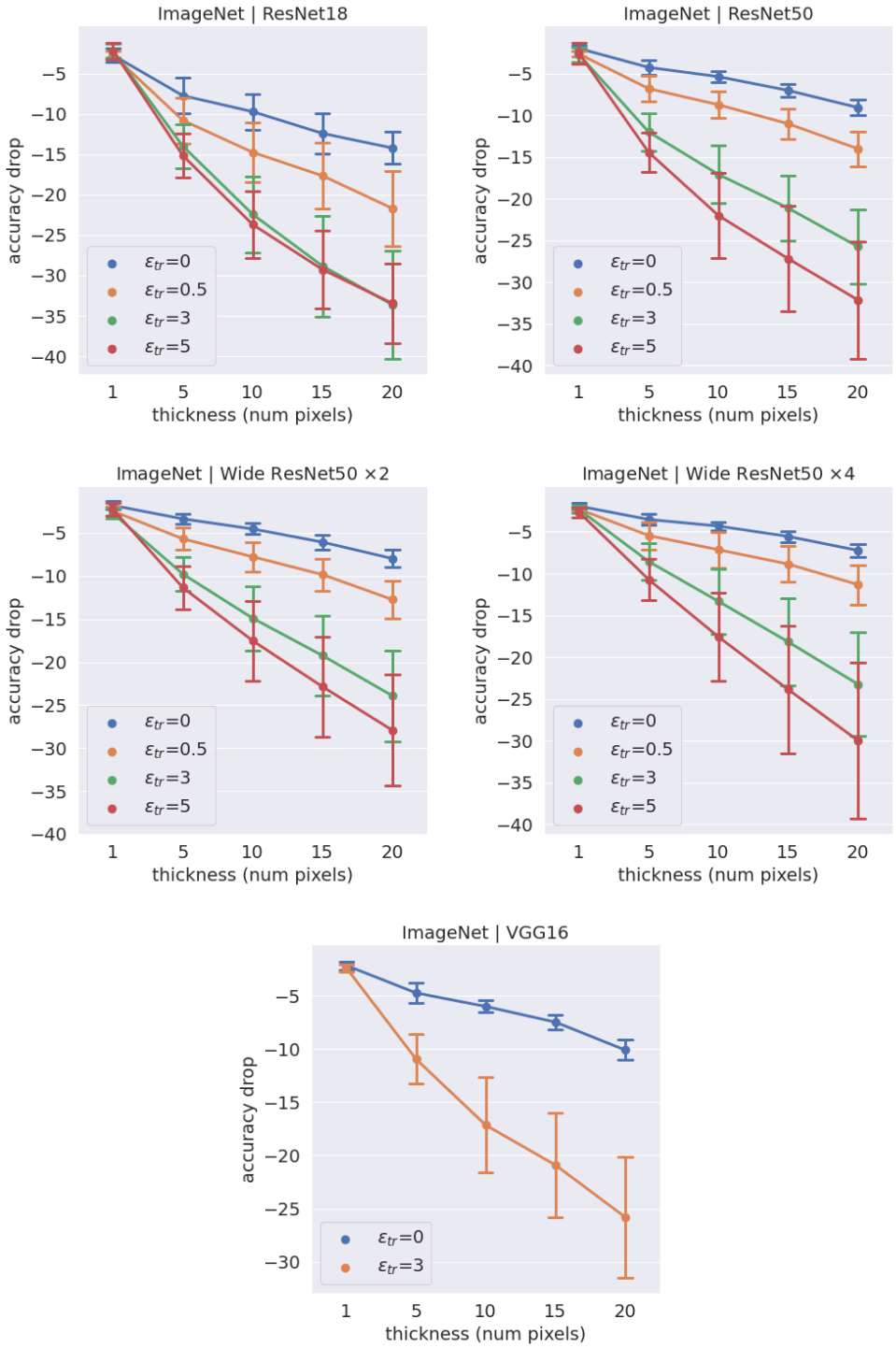

Figure 5.10: Colored contours for ImageNet models: natural accuracy drop with respect to the original test set. Circles represent the average accuracy drop over different colors (white, red, green, blue), and error bars indicate the corresponding standard deviation. Robust models are trained with ℓ_2-norm.

Table 5.5: Natural accuracy with different pixel-averaging methods for ImageNet-9. Accuracy is computed based on the 9 classes of the ImageNet-9 dataset. Robust models are trained with ℓ_2-norm. The first line of each block gives the accuracy on the original images.

Model	**Transform**	$\varepsilon_{tr} = 0$	$\varepsilon_{tr} = 0.5$	$\varepsilon_{tr} = 3$	$\varepsilon_{tr} = 5$
ResNet18	-	93.68	92.74	86.37	80.81
	avg	2.42	4.64	4.47	9.28
	avg bg	2.57	5.48	5.78	8.74
	avg fg	1.11	1.75	1.85	5.48
	shape+color	8.69	11.75	15.78	17.58
ResNet50	-	95.83	95.56	91.58	87.41
	avg	0.77	5.26	5.46	9.56
	avg bg	1.09	6.25	6.52	8.96
	avg fg	0.05	1.93	4.15	5.43
	shape+color	12.30	14.10	16.27	18.59
Wide ResNet50 ×2	-	95.83	96.37	92.86	89.85
	avg	2.27	4.57	8.52	10.22
	avg bg	2.52	5.46	8.64	9.70
	avg fg	1.93	1.53	5.26	7.11
	shape+color	17.58	10.86	17.83	18.49
Wide ResNet50 ×4	-	96.59	96.67	93.98	92.35
	avg	1.21	5.41	9.06	7.46
	avg bg	1.43	5.90	9.33	7.85
	avg fg	0.27	1.78	4.77	4.10
	shape+color	15.65	11.93	17.63	17.41
VGG16	-	94.20	-	88.12	-
	avg	1.65	-	6.44	-
	avg bg	2.22	-	7.11	-
	avg fg	0.42	-	3.16	-
	shape+color	8.52	-	15.80	-

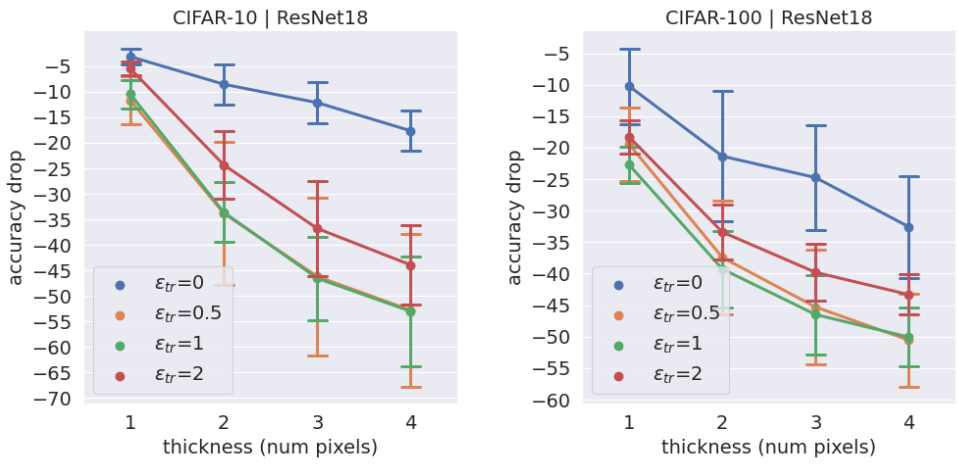

Figure 5.11: Colored contours for CIFAR-10 and CIFAR-100 models: natural accuracy drop with respect to the original test set. Circles represent the average accuracy drop over different colors (white, red, green, blue), and error bars indicate the corresponding standard deviation. Robust models are trained with ℓ_2-norm.

help improve our understanding of AT.

5.7 Conclusions

In this Chapter, we highlight previously unnoticed properties of the representations learned by adversarially-trained CNNs, improving our understanding of AT and highlighting the limitations it induces on modern CNN architectures. Specifically, we show that i) a larger portion of spatially dense feature maps in robust models activate for all data points in the dataset, thus reducing the model expressivity; ii) the information conveyed through the network is more redundant in robust models than in natural ones, limiting the effective number of active channels in hidden layers; iii) robust models spend capacity to learn predictive features that are not properly leveraged by the robust classifier. Taken together, these results suggest that model capacity is not exploited efficiently when CNNs are trained with AT. In addition, we demonstrate that robust models overly rely on global color information that could result in undesired behaviors under simple content-preserving perturbations that do not compromise other robust features such as the shape. Our findings are consistent with the analysis in [Yang et al., 2020], where the authors conjecture that the accuracy drop in adversarially-trained CNNs is likely due to limitations in current architectures and training routines. Our experiments investigate the root causes of such limitations and provide insights that may be useful for achieving robustness without resorting to AT.

Limitations and Future Works Our work is focused on CNN architectures. We believe it would be interesting to extend the use of our tools to unveil the properties of other architectures - such as Visual Transformers [Dosovitskiy et al., 2020] - or the impact of other training paradigms - such as multitask learning [Ruder, 2017, Mao et al., 2020] and self-supervised learning [Chen et al., 2020b, He et al., 2020, Kolesnikov et al., 2019] - in future works.

Moreover, our experiments on latent features show, contrary to common belief, that the latent space of adversarially-trained models is not inherently robust, but offers varying levels of robust and natural accuracy. On one side, this implies an additional source of robustness degradation in transfer learning settings, which adds to the decrease due to the domain shift. On the other side, this property can be leveraged to extract - from the same model - representations with different characteristics in terms of robustness and predictive power, enabling one to switch seamlessly between tasks that require different levels of robustness. Investigations along these lines could be a good avenue for future work.

Chapter 6

Improving Robustness with Image Filtering

6.1 Introduction

The methods described in this Chapter to improve the robustness of DNNs are inspired by the visual inspection of many adversarial examples. As can be appreciated in Figure 6.1, adversarial attacks seek to create spurious micro-patterns that, although perceptually negligible to the human observer, could dramatically impact the behavior of DNNs [Szegedy et al., 2013], as discussed in Section 4.3. These artifacts give rise to many tailored 'virtual edges' and micro-textures, which could leverage structural properties of the CNN architecture (specifically, the sum-aggregation operation in convolutional layers and the increasing receptive field size in deeper layers) to easily subvert the model's predictions.

In recent years, many defenses against adversarial examples have been proposed. However, the vast majority of the proposed approaches failed under stronger attacks or adaptive ones, as their false sense of robustness turned out to be related to different forms of obfuscated gradients [Athalye et al., 2018, Tramer et al., 2020]. In particular, all defenses based on transformations of the input image (such as Total Variation, JPEG compression, etc. [Guo et al., 2017]) proved to be ineffective [Athalye et al., 2018]. The most reliable method to truly enforce robustness in DNNs is AT [Madry et al., 2017], which consists of training DNNs on adversarial samples rather than natural ones, as discussed in Section 4.4. In these settings, the training process is induced to find a set of parameters so that the model is robust to adversarial examples. The exposure of the spurious textures characterizing adversarial examples during training lets the model infer that a particular configuration of a few distant pixels is, in principle, not correlated with the task or that a particular local texture is not predictive. As this process takes place during training, the model can adapt its parameters accordingly.

Figure 6.1: Micro-patterns created by ℓ_2-bounded adversarial attacks on CIFAR-10 images.

As anticipated above, many issues related to the correlation of spatially distant pixels and the accumulation of the malign effects of adversarial perturbations are due to the model architecture itself rather than the training procedure or the training data. Therefore, major enhancements in adversarial robustness should come from novel architectures taking into account the structural weaknesses of current CNNs. However, as we shall show in our experiments, significant improvements can still be achieved by acting on the input space. Indeed, based on the above intuitions, this Chapter presents a novel image filtering framework that improves robustness against adversarial attacks without resorting to AT.

In the remainder of this Chapter, we review existing research works on adversarial robustness in Section 6.2. In Section 6.3 we lay out the motivations that inspired the proposed method, while in Section 6.4 we summarize the contributions of this Chapter. In Section 6.5 the image filtering framework designed to improve the robustness of DNNs is introduced, along with a defense scheme based on it. In Section 6.6 we present the results of our experiments. In Section 6.7 we sum up our findings, discuss limitations and potential improvements of the proposed method and point to some possible research directions for future work.

6.2 Related Work

Since the discovery of the existence of adversarial examples [Szegedy et al., 2013], countless attempts to make image classifiers robust to adversarial perturbations have appeared in the literature. As anticipated in Section 6.1, current state-of-the-art methods to enforce robustness directly leverage adversarial ex-

amples during training [Goodfellow et al., 2014, Kurakin et al., 2016, Kannan et al., 2018, Madry et al., 2017]. A major limitation affecting AT is the high computational cost as adversarial examples are typically crafted through iterative optimization routines. For this reason, significant efforts have been made to improve the robustness of models without resorting to AT. Proposed solutions cover a wide range of techniques based on curvature regularization [Moosavi-Dezfooli et al., 2018], robust optimization to improve local stability [Shaham et al., 2015], the use of additional unlabeled data [Carmon et al., 2019], local linearization [Qin et al., 2019], Parseval networks [Cisse et al., 2017], defensive distillation [Papernot et al., 2016], model ensembles [Pang et al., 2019], channel-wise activations suppressing [Bai et al., 2021], feature denoising [Xie et al., 2019], self-supervised learning for adversarial purification [Shi et al., 2020], and input manipulations [Guo et al., 2017, Dziugaite et al., 2016, Lu et al., 2017]. All the listed techniques, except those based on input manipulations, require training the model or an auxiliary module from scratch. Our method, instead, can be used in combination with pretrained models.

As regards the techniques based on input manipulations, in [Lu et al., 2017] the authors analyze the effect of image rescaling on adversarial examples. [Dziugaite et al., 2016] explores the possibility of improving robustness through JPG compression based on the intuition that adversarial perturbations are unlikely to leave an image in the space of JPG images. In [Guo et al., 2017], the authors assess the effectiveness of input transformations based on image cropping and rescaling, bit-depth reduction, JPEG compression, total variance minimization, and image quilting. The main objective is to remove the adversarial perturbations from images while preserving sufficient information to correctly classify them. Unlike the mentioned input-based techniques that only implicitly (or not at all) induce mitigation of the bias toward textures, our method makes this effect explicit by directly removing textures from the input image. Differently from prior literature, we properly test our method according to guidelines for input-based defenses [Athalye et al., 2018] and show promising performance.

6.3 Motivations

By visual inspection of adversarial examples in Figure 6.1, we noticed that adversarial attacks introduce micro-patterns that, as discussed in Section 6.1, could fool image classifiers. An artificially created virtual edge could trigger particular edge detectors amongst the model's convolutional kernels, whose activation may be combined thanks to the sum-aggregation operation performed in convolutional layers. Moreover, the effects of the activation of these edge detectors at different - and potentially distant - spatial locations may be combined in deeper layers by leveraging the increasing receptive field size. As a result, the delicate balance of activations the model relies on could be broken, leading to a significant drop in generalization power [Tsipras et al., 2018, Ilyas et al., 2019].

In light of these observations, it is evident that the fundamental strength of the micro-patterns introduced by adversarial attacks lies in their simultaneous local and non-local nature. Locality stems from the fact that the perturbation can be very precise as the adversary could alter any spatial location and cause a direct effect on neighboring pixels. This phenomenon enables the creation of local virtual edges, that are defined by neighboring pixels with different intensities. Non-locality, instead, emerges from the possibility of inducing correlation among spatially distant pixels thanks to the increasing receptive field size in deeper layers. In view of this, the attacker is allowed to be very precise in choosing the optimal spatial locations so that the malign effects of perturbations can conveniently accumulate across layers, while affecting only minimally the semantics of the image.

The goal of the method proposed in this Chapter is to limit the action space of the adversary so as to make the creation of virtual edges and micro-textures more difficult and less precise. This is done by means of an iterative filtering procedure that merge together pixels that are perceptually similar at a given resolution.

6.4 Contributions

While it is not the only technique in the literature based on input manipulation to prevent adversarial attacks [Guo et al., 2017, Lu et al., 2017, Dziugaite et al., 2016], the method introduced in this Chapter differs significantly from prior works in the fundamental concepts at the basis of its functioning. The main difference between our method and others relying on input transformations is that we *explicitly* remove textures from input images while preserving the informative features. This is achieved through an iterative filtering procedure based on simple rules that do not require additional training routines.

Our image filtering strategy can also be exploited as a data augmentation, which we show to be beneficial also for robustness against common corruptions (e.g., shot noise, pixelate, glass blur). Additionally, our framework allows for high flexibility in the choice of its key components, which makes it basically independent of the specific implementation as long as its underpinning principles are not violated.

The contributions of this Chapter can be summarized as follows:

- We introduce a novel image filtering framework, dubbed *Image-Graph Extractor (IGE)*, whose main objective is to remove textures while preserving the image's semantic content. Our framework is based on an implicit graph representation of images, the *Image-Graph (IG)*.

- We propose an effective defense against ℓ_p-bounded adversarial attacks, called *Filtering As a Defense (FAD)*, leveraging the texture suppression effect of IGE.

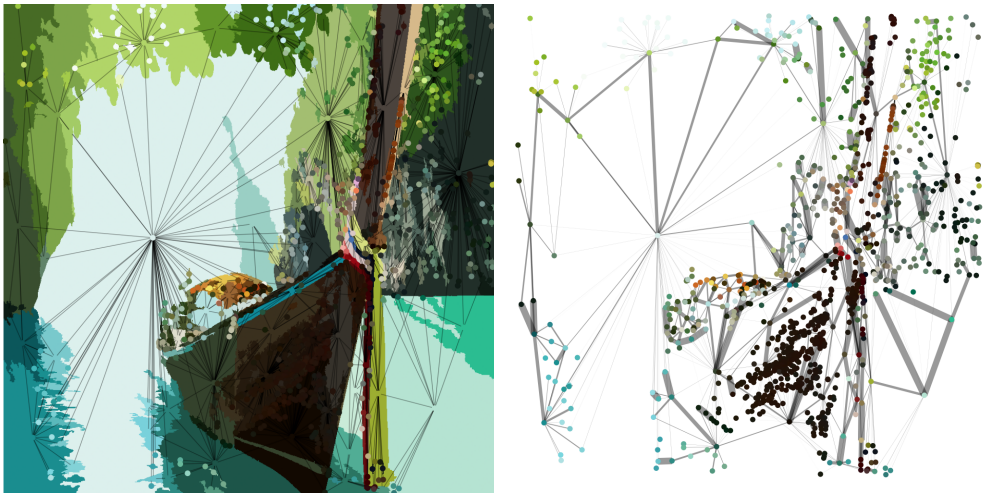

Figure 6.2: Example of IGE-based image filtering for a 1000×1000 image. Image-Graph over the filtered image with $r^* = 0.5$ (left), and Image-Graph (right). The thickness of edges is proportional to the fraction of the perimeter shared by neighboring nodes.

- We exploit filtered images produced through the IGE filtering framework as a data augmentation strategy. We show that models trained with our data augmentation strategy are more robust against common corruptions such as shot noise, pixelate, and glass blur.

While IGE has been primarily conceived and designed to enforce robustness, the intermediate graph representation necessary for the iterative filtering procedure might prove useful in many other applications. In fact, the IG associated with the input image enables the emergence of structure in an unstructured domain. The extracted structure, if adequately exploited, might boost performance in downstream tasks. In Figure 6.2 is shown an example of a filtered image and the resulting IG.

6.5 Methods

6.5.1 Image Graph Extractor

In this Section, we introduce the IGE framework. For the sake of readability, we first highlight the guiding principles at its core in Section 6.5.1.1 and describe thoroughly its key components, i.e., the merging rule (Section 6.5.1.2) and the iterative filtering procedure (Section 6.5.1.4). All the design choices characterizing the specific instance of the IGE method used for our experiments are discussed in Section 6.5.1.3. Implementation details are given in Appendix B.1. In order to satisfy parallel and efficient image processing, we developed IGE in `CUDA/C++` and NVIDIA `Thrust`.

6.5.1.1 Guiding Principles

The information contained in an image depends on the effective resolution r we see at: when 'zooming out' the image (i.e., decreasing r), pixels that are distinct at higher resolutions are no more distinguishable and merge together forming patches with larger size. Conversely, finer details emerge when 'zooming in' the image (i.e., increasing r) and patches split into multiple sub-patches with smaller sizes. Note that the merging process does not depend only on color similarity, but also on the relative size of patches. Based on these observations, an image $\mathbf{x}$ at a given resolution r can be represented as a graph, where nodes correspond to patches of similar pixels and edges encode information about connectivity of the different patches. The size of each node amounts to the number of pixels the corresponding patch is composed of. As a result, each image can be partitioned into a collection of patches according to an underlying graph representation.

6.5.1.2 Merging Rule

The objective of IGE is to emulate the perceptual effects described in Section 6.5.1.1. While the produced filtered images reveal complex elaborations that might be hard to model, IGE is driven by a simple merging rule, which we describe hereinafter.

Let $\mathbf{x}^r$ be an image at resolution r and let $\mathcal{G}^r = (\mathcal{V}^r, \mathcal{E}^r)$ be the corresponding IG, where $\mathcal{V}^r$ is the set of nodes and $\mathcal{E}^r$ the set of edges at resolution r. For the sake of simplicity, in the following, we implicitly assume the dependence on r and omit it from the notation (e.g., we use $\mathcal{V}$ in place of $\mathcal{V}^r$). We also assume edges to be binary-valued, with $e_{i,j} = 1$ indicating that v_i and v_j are neighboring nodes, where $e_{i,j} \in \mathcal{E}$ and $v_i, v_j \in \mathcal{V}$. In the image domain, this translates into the fact that patches corresponding to v_i and v_j are neighbors (according to some specified criterion). Let s_i and s_j be the size of nodes v_i and v_j, respectively. We define the color associated with node v_i, denoted with c_i, as the average value of pixels in v_i. Given an opportune color distance d_c, we define the *adjusted distance* d_a as

$$d_a(c_i, c_j; s_i, s_j, r) = \varphi(d_c(c_i, c_j), s_i, s_j, r) \tag{6.1}$$

for some function φ. Notice that d_a is function of the color distance $d_c(c_i, c_j)$, the size of nodes s_i and s_j, and the resolution r. Neighboring nodes v_i and v_j are merged if

$$d_a(c_i, c_j; s_i, s_j, r) < \tau(d_0, r) \tag{6.2}$$

where $\tau(d_0, r)$ is the threshold (function of the resolution r) that determines if two nodes are in the same patch at a given resolution r. The parameter d_0 represents the minimum color distance perceived at maximum resolution r_0, corresponding to the resolution of the original (unfiltered) image $\mathbf{x} = \mathbf{x}^{r_0}$. We assume by convention that $r_0 = 1$. In order to be consistent with the guiding

principles in Section 6.5.1.1, τ must be a monotonically decreasing function with respect to r, that is $\dfrac{\partial \tau(d_0, r)}{\partial r} < 0$. Indeed, as we zoom out the image (r decreases), pixels that are more and more distant from each other in color space become indistinguishable, resulting in a less strict threshold.

The adjustment of the color distance d_c modeled by Equation (6.1) takes into account the size of nodes. The rationale behind the choice to incorporate the size of nodes into the computation of the adjusted color distance can be better explained by an example: a black 1-pixel node surrounded by white pixels becomes perceptually indistinguishable from its neighbors at a far larger resolution if compared to a black 10-pixels node.

In summary, the core of the IGE framework is represented by the merging rule described in this Section. The key ingredients defining the rule are: the criterion used to determine neighboring patches, the color distance d_c, the adjusted distance d_a (specifically, the function φ), the minimum color distance perceived at maximum resolution d_0, and the threshold function τ. The choice of the latters is subject to the following constraints:

C1. If all the variables but r are kept fixed, it is required that $\frac{\partial \tau(r)}{\partial r} < 0$ and/or $\frac{\partial d_a(r)}{\partial r} > 0$.

C2. If all the variables but s_i are kept fixed, it is required that $\frac{\partial d_a(s_i)}{\partial s_i} > 0$.

C3. If all the variables but d_c are kept fixed, $\dfrac{\partial d_a}{\partial d_c} > 0$.

The IGE framework is general in that it allows flexibility in the choice of the key ingredients as long as the constraints listed above are satisfied.

6.5.1.3 Design Choices

As stated in Section 6.5.1.2, to define an instantiation of the IGE framework, we need to specify: (i) the criterion used to define neighboring patches; (ii) the color distance d_c; (iii) the adjusted distance d_a (specifically, we should define the function φ); (iv) the merging threshold function τ; (v) the minimum color distance perceived at maximum resolution d_0. In the following are discussed the design choices adopted for our experiments. These are the result of some trial and error, and a more comprehensive experimental setup would certainly lead to better design choices and improved performance. Nevertheless, our main goal here is to showcase the general effectiveness of the IGE framework to enforce robustness, with no special focus on the specific implementation.

- As for (i), we consider the most natural definition of neighboring patches, i.e., patches whose borders touch in at least one pixel. While simple, this choice could cause merges that are not optimal, and better tuning of the length of the shared border may lead to significant improvement in the filtering procedure.

- As regards the choice of the color distance d_c, we tested both the CIEDE2000 distance [Sharma et al., 2005] and the Euclidean distance. While CIEDE2000 is more aligned with human color perception, we did not notice significant differences compared to the Euclidean distance, and for the experiments presented in this Chapter, we used the latter. Given two RGB vectors $p = [p_R, p_G, p_B]^\top$ and $q = [q_R, q_G, q_B]^\top$ their Euclidean distance is given by
$$d_c(p, q) = \sqrt{(p_R - q_R)^2 + (p_G - q_G)^2 + (p_B - q_B)^2}. \tag{6.3}$$

- For the adjusted distance d_a, we adopted the following formulation
$$d_a = d_c(c_i, c_j) \cdot \tilde{\varphi}(s_i, s_j, r), \tag{6.4}$$
$$\tilde{\varphi}(s_i, s_j, r) = \left(1 + e^{(-\alpha(\min\{s_i, s_j\} - \frac{\beta}{r}))}\right)^{-1/r} \tag{6.5}$$
where the color distance d_c and adjustment $\tilde{\varphi}$ terms are decoupled. We set $\alpha = 0.04$ and $\beta = 10$. The term $\tilde{\varphi}$ is aimed at reducing the perceived color distance when the resolution is low and/or the size of nodes is small. The adjustment for the size of nodes is represented by the term $\min\{s_i, s_j\}$. Notice that the size adjustment depends only on the smallest node. This is to encode the fact, discussed in Section 6.5.1.2, that smaller patches become perceptually indistinguishable from their neighbors at larger resolutions. This rule may be refined by also considering the relative size of nodes s_i/s_j. The term $\frac{\beta}{r}$ helps incentivize the early merge of small nodes. It is worth noting that several functions φ have been tested and the quality of the filtered images is quite stable under different choices of φ.

- As for the threshold function τ, we use
$$\tau(d_0, r) = d_0 \frac{1 - (r - r_m)}{r_m}, \tag{6.6}$$
where $r_m = 0.1$.

- For the minimum color distance perceived at maximum resolution, we set $d_0 = 0.03$.

6.5.1.4 Iterative Filtering

The iterative filtering procedure consists in repeatedly performing merging operations after a preliminary phase for initialization. The functioning of the whole IGE framework is summarized in Algorithm 4. Let r^* be the *target resolution* and $r_0 = 1$ the initial resolution corresponding to the original image $\mathbf{x}$, as mentioned in Section 6.5.1.2. In our framework, the target resolution $r^* \in]0, r_0[$, together with the *filtering step size* Δ_r, defines the number of filtering steps $N_r = \frac{1-r^*}{\Delta_r}$ to be performed.

Algorithm 4 IGE pseudocode.

Input: original image $\mathbf{x}$
Output: filtered image $\mathbf{x}^{r^*}$
Parameters: r^*, d_c, d_a, d_0, τ

$\mathcal{G}^r = \texttt{img2graph}(\mathbf{x}^{r_0})$
$r = r_0 - \Delta_r$
while $r > r^*$ **do**
    $\mathcal{G}^r = \texttt{merge}(\mathcal{G}^r; d_c, d_a, d_0, \tau)$
    $\mathcal{G}^r = \texttt{avg\_color}(\mathcal{G}^r)$
    $r = r - \Delta_r$
end while
$\mathbf{x}^{r^*} = \texttt{graph2img}(\mathcal{G}^r)$

The initialization phase is aimed at extracting the IG $\mathcal{G}^{r_0}$ associated with the original image $\mathbf{x} = \mathbf{x}^{r_0}$. We represent this operation with the generic function `img2graph`.

The filtering routine can be seen as an anti-causal discrete dynamical system $\mathcal{G}^{r-\Delta_r} = m(\mathcal{G}^r)$, where m is the map obtaining $\mathcal{G}^{r-\Delta_r}$ from $\mathcal{G}^r$. Specifically, m is composed of the following elementary operations:

1. Merge nodes according to Equation (6.2) (we denote this operation with the function `merge`).

2. Update the color value of each node by averaging the values of all pixels in the node (we denote this operation with the function `avg_color`).

The map m is applied iteratively N_r times until the target resolution r^* is reached. The final step consists in reconstructing the filtered image at the target resolution from the final IG $\mathcal{G}^{r^*}$. We represent this operation with the generic function `graph2img`.

A potential issue that might arise is the co-existence, for a specific node, of multiple candidate nodes for the merging operation. Although not optimal from a computational point of view, one trivial solution would be to choose the candidate node with the smallest adjusted distance. In this regard, the filtering step size Δ_r controls the goodness of the filtering process. Small values of Δ_r help in mitigating the problem of multiple candidate nodes for merging. Since $N_r \propto \Delta_r^{-1}$, this strategy should call for thoughtful considerations about computational cost. In light of this, we set Δ_r, by visual inspection of a few images, to the biggest value that guarantees the preservation of the image's semantic content.

Notice that the IG associated with the filtered image at the target resolution r^* can be extracted with no extra computations as it comes as an intermediate by-product of the filtering procedure. The IG can be enriched by considering additional node attributes besides node color (e.g., shape information by means

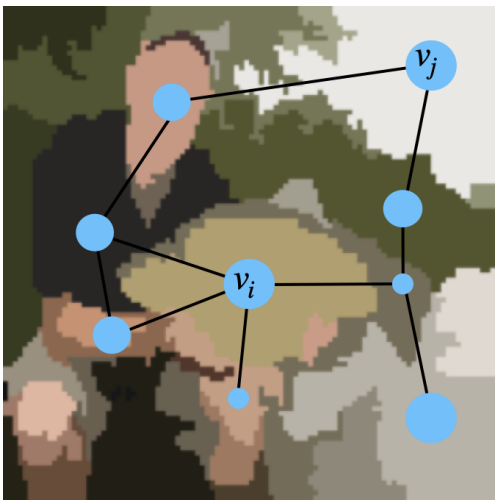

Figure 6.3: A simplified example of an IG associated with an image from the ImageNet dataset. For graphical purposes, only the main nodes are displayed.

of shape descriptors) and encoding additional information about connectivity as edge attributes. A simplified example of IG for an image from the ImageNet dataset is depicted in Figure 6.3. For ease of visualization, only the main nodes of the IG are shown.

According to [Stutz et al., 2018], the IGE framework formally is not a superpixels algorithm as the number of patches is not controllable. In the settings where the IGE is applied in this work (i.e., as a tool to enforce robustness), this represents a clear advantage. Indeed, it allows for an adaptive number of nodes depending on the content of the image, leading to a more natural and semantically meaningful representation of filtered images. We discuss more in depth this point in Section 6.5.2.1.

6.5.2 Filtering As a Defense

In this Section, we describe FAD, our proposed defense against adversarial attacks leveraging the IGE iterative filtering procedure. We first expand the discussion on the motivations behind the IGE framework as a tool to improve the robustness of DNNs in Section 6.5.2.1. Finally, we introduce the full defense protocol used for our experiments in Section 6.5.2.2.

6.5.2.1 Motivation

We want to provide here an intuitive argument to justify the effectiveness of FAD for improving robustness against adversarial examples. Let $\mathbf{x}$ be an input image of n pixels and let $\mathbf{x}^{r^*}$ be its filtered counterpart at a given target resolution r^*. We define n^* as the *effective number of pixels* in the filtered image $\mathbf{x}^{r^*}$. Indeed, we can think of the filtered image as a $\sqrt{n^*} \times \sqrt{n^*}$ image, with

$n^* \leq n$ (and potentially $n^* \ll n$). Notice that n^* corresponds to the number of patches the filtered image is composed of. In the context of the IGE framework, n^* is equivalent to the number of nodes in the IG $\mathcal{G}^{r^*}$ associated with the filtered image $\mathbf{x}^{r^*}$.

A useful observation is that robustness depends on the structure of the input image, which means that there exists an *intrinsic* robustness level associated with every image. If we consider the image as a graph, we can easily understand why this is true. The two key elements that define the image structure are nodes and edges. In the original (i.e., unfiltered) image, the nodes of the associated graph are essentially the individual pixels, and the edges connect each pixel to its 8 neighbors (if we assume 8-connectivity). The structure of the original image is represented by a rigid uniform grid, where a pixel's neighborhood has a fixed size equal to 8. In the filtered image, the nodes of the associated graph are groups of pixels (i.e., patches) that have been merged together, and the edges connect each patch to a variable number of neighboring patches, depending on the content of the image. Therefore, the structure of the filtered image is represented by a non-uniform grid. This difference in the structure between the original image and its filtered counterpart is at the basis of their difference pertaining to the inherent level of robustness. Indeed, the perturbation of a node in the original image directly impacts only the 8 nearby pixels and can thus lead to the emergence of virtual edges and micro-textures. The perturbation of a node in the filtered image, instead, could impact very distant pixels due to the non-uniform topology of the underlying grid and the potentially dense connectivity of the associated graph. As a consequence, when attacking filtered images, the adversary may not be as precise as it would be for original images and this makes it harder to create micro-patterns that could fool the model. The most effective strategy for the adversary to craft a successful attack is to modify the connectivity of the graph, that is, merge adjacent nodes, or split nodes. However, as patches obtained by filtering at low resolutions may be large, this may significantly alter the semantics of the image and make the attack easy to spot.

6.5.2.2 Defense Protocol

The IGE filtering procedure outlined in Section 6.5.1 causes a shift in the statistics of the input image that may result in distribution shifts of batch normalization layers' inputs. Thus, especially in complex datasets like ImageNet, models are less accurate when evaluated on filtered images as population statistics estimated on training data do not match statistics of test data. To overcome this mismatch, the first phase of the protocol consists in fine-tuning the pretrained model to update the batch normalization layers' statistics so as to adapt to the characteristics of filtered images. Since the semantic content of input images is preserved, the fine-tuning process requires only a few epochs to converge.

The second phase is the defense itself, which consists in running the IGE

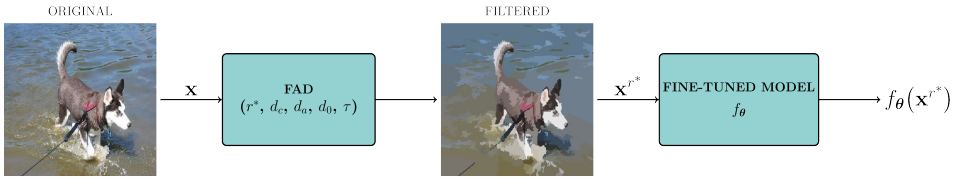

Figure 6.4: Block diagram describing the FAD defense.

iterative filtering routine as a pre-processing step any time an input image is queried. Figure 6.4 reports the described FAD defense. Notice that the FAD defense module inherits the parameters defining the specific instantiation of the IGE filtering method.

6.6 Experimental Results

In this Section, we present three sets of results: (i) we assess the quality of filtered images by visual inspection; (ii) we evaluate the performance of models equipped with the FAD defense in terms of natural accuracy and robustness to adversarial attacks; (iii) we analyze the performance on corrupted data for models trained with filtered images as data augmentation. Regarding point (ii), we compare FAD with AT, as the latter is widely considered the state-of-the-art in the robustness community and any comparison with other methods would add little to the discussion of the results.

6.6.1 Filtering

In Figure 6.5 we show the filtered images as the target resolution varies, representing the progression of the iterative filtering process. When reducing the target resolution, the number of nodes progressively decreases while their size generally increases. Notice that, once fixed the target resolution of the filtered image, nodes with significantly different sizes coexist. This inherent flexibility allows the preservation of important details by maintaining a higher number of small nodes in areas of the image that are not perceptually homogeneous. As a result, the semantic content of the image is not altered even for low values of the target resolution. Additional examples are given in Figures 6.6 and 6.7

In Figure 6.8 are displayed some examples of filtered images from the ImageNet dataset with fixed target resolution $r^* = 0.6$. Notice that the number of nodes is not the same for different images, as it depends on the specific content and, thus, the complexity of the original image. On average, at target resolution $r^* = 0.6$, the number of nodes in filtered ImageNet images in Figure 6.8 is within the range $[150, 1000]$.

Figure 6.5: Example of the effect of the IGE filtering when decreasing the target resolution. Top left panel: $r^* = 1$ (original image). Top right panel: $r^* = 0.7$. Bottom left panel: $r^* = 0.45$. Bottom right panel: $r^* = 0.35$.

Figure 6.6: Examples of the effect of the IGE filtering when decreasing the target resolution. Top left panel: $r^* = 1$ (original image). Top right panel: $r^* = 0.7$. Bottom left panel: $r^* = 0.45$. Bottom right panel: $r^* = 0.35$.

Figure 6.7: Examples of the effect of the IGE filtering when decreasing the target resolution. TTop left panel: $r^* = 1$ (original image). Top right panel: $r^* = 0.7$. Bottom left panel: $r^* = 0.45$. Bottom right panel: $r^* = 0.35$.

Figure 6.8: Examples of filtered ImageNet images with $r^* = 0.6$ and $\Delta_r = 0.1$.

6.6.2 Robustness Evaluation

As discussed in [Tramer et al., 2020], one should develop adaptive attacks to test the robustness of a defense. In this context, the Backward Pass Differentiable Approximation (BPDA) is a gradient approximation technique [Athalye et al., 2018] to deal with non-differentiable components in DL models. Let $\tilde{f}$ be the composition $\tilde{f}(x) = f \circ h(x)$ where h is non-differentiable (or hard to differentiate). In our case, h is the FAD module, and f represents the fine-tuned model. To approximate the total gradient $\nabla_x \tilde{f}(x)$, BPDA needs to find a differentiable function $g(x)$ such that $g(x) \approx h(x)$. Then, when computing the gradient $\nabla_x \tilde{f}(x)$, the forward pass is a standard forward pass through $\tilde{f}$ (including the forward pass through h), while in the backward pass h is replaced by g. In our case, a reasonable choice for g is the identity function $g(x) = x$, in line with experiments performed in [Athalye et al., 2018] to break down defenses based on input transformations. Differently from the mentioned methods, whose apparent improvements in robustness were due to improper testing, our FAD defense achieves promising results under BPDA attacks.

For the sake of even more fair testing, we note that a vanilla application of BPDA may not be effective as one step of gradient might not allow the attacker to craft adversarial patches. Hence, we apply BPDA (as an outer loop) with the addition of an inner attack (addressed to f) where multiple PGD steps are performed. Notice that the additional internal cycles slow down significantly the adversarial attack construction. This technical aspect is usually overlooked in academic research. However, in real-world settings where time efficiency might be required for the attack to be successful, it could play a decisive role.

6.6.3 Experimental Settings

We validate FAD on CIFAR-10 [Krizhevsky et al., 2009c], CIFAR-100 [Krizhevsky et al., 2009c], and ImageNet [Deng et al., 2009] datasets. As discussed in Section 6.5.2.2, filtered images exhibit different statistics if compared to their original counterparts. For this reason, the first phase of the FAD protocol is a fine-tuning of the model. While the optimal solution would be to fine-tune the model with the same resolution used in the FAD module, we decided to fine-tune the model for only one resolution $r^* = 0.6$. Then, the same fine-tuned model is re-used for all the experiments. While not optimal, we do so for practical reasons and yet get very good results. Performance could be further boosted by matching the resolution of the fine-tuning process with that of the FAD module. For all the datasets, we employ ResNet18 [He et al., 2016b] architectures trained for 150 epochs, initial learning rate $\eta = 0.1$ and a drop of η by a factor 10 every 50 epochs. As for BPDA, we evaluate FAD with 50 outer steps and 20 inner steps. Attacks are bounded in ℓ_2-norm. Due to computational constraints, we tested on 30% of the test sets with shuffling.

For CIFAR-10 and CIFAR-100, we trained robust models with 7 PGD steps,

$\alpha = 1$ and $\varepsilon_{tr} \in \{0.5, 1, 2, 3\}$ (attacks bounded in ℓ_2-norm). For ImageNet, we leveraged pretrained robust models from [Salman et al., 2020a] with $\varepsilon_{tr} \in \{0.5, 1, 3, 5\}$.

Models are fine-tuned for 10 epochs with learning rate $\eta = 0.001$. For computational reasons, for ImageNet we only use 300 filtered images per class from the training set for the fine-tuning.

6.6.4 FAD

Tables 6.1 and 6.3 report the results for CIFAR-10/CIFAR-100 and ImageNet, respectively, when attacking models equipped with the FAD defense with BPDA ℓ_2-bounded attacks. As the resolution increases, the attacker prevails over the defense (i.e., the robust accuracy decreases). Robustness is lower for larger values of the perturbation budget ε_{te}, and for values of ε_{te} larger than the reported ones, robustness quickly approaches zero. These behaviors are a clear sign that the FAD defense does not rely on obfuscated gradients. By reducing r^*, the filtered images have fewer textures, and the adversary is less effective, in line with arguments given in Section 6.5.2.1. In Tables 6.2 and 6.4 are reported the performance of AT (against PGD attacks, 50 steps). Notice that at comparable levels of natural accuracy, our FAD defense exhibits better robustness. For example, for ImageNet our defense with $r^* = 0.8$ achieves 56.61% natural accuracy, which is comparable to a model adversarially-trained with $\varepsilon_{tr} = 3$, whose natural accuracy is 53.12%. As for robust accuracy, our defense achieves 41.86%, 28.84% and 5.92% with $\varepsilon_{te} = 3, 5, 10$, respectively, while the adversarially-trained model achieves 31.00%, 17.73% and 2.51%. A natural question arising from the analysis of these results is to explain why FAD achieves a better accuracy-robustness trade-off compared to AT. We hypothesize that this is mainly due to the convolutional architecture. Indeed, IGE applies highly non-linear transformations to the input that cannot be encoded by a CNN with a limited number of layers, and it is well-known in the literature that AT can achieve better performance when the number of layers increases.

Notice that, in order to enforce robustness, filtered images should exhibit the minimum number of nodes (see the discussion in Section 6.5.2.1) while keeping unaltered the semantics of the original image. Traditional superpixel algorithms, like SLIC [Achanta et al., 2012], fail to achieve a satisfactory trade-off, while IGE can successfully preserve important details of the original image at low resolutions (as mentioned in Section 6.6.1). To confirm this fact, we applied FAD with SLIC and obtained results that are not comparable with IGE. In order to be fair, we set the number of nodes for SLIC similar to that provided by IGE at a given resolution. To give an example, we found that for CIFAR-10 the optimal number of nodes for FAD with SLIC is around 20, which is comparable with IGE at resolution 0.5. The best robust accuracy we obtained against attacks with $\varepsilon_{te} = 1$ is less than 25%, 3× less than the one obtained with IGE.

In Section 6.5.2.1, we conjectured that a successful attack against FAD

Table 6.1: Natural and robust accuracy (in %) for CIFAR-10 and CIFAR-100 against BPDA ℓ_2-bounded attacks at different resolutions. FAD models are fine-tuned with $r^* = 0.6$

Defense	Clean	$\varepsilon_{te} = 1$	$\varepsilon_{te} = 2$	$\varepsilon_{te} = 3$	$\varepsilon_{te} = 4$
CIFAR-10					
FAD $r^*=0.8$	89.55	74.48	33.40	1.82	0.00
FAD $r^*=0.7$	86.07	73.63	44.50	8.66	0.68
FAD $r^*=0.6$	85.06	73.18	49.80	18.26	4.39
FAD $r^*=0.5$	82.32	74.38	52.67	27.31	11.36
FAD $r^*=0.4$	78.45	70.15	55.37	35.09	21.16
CIFAR-100					
FAD $r^*=0.8$	63.74	45.31	19.17	3.32	0.23
FAD $r^*=0.7$	59.54	43.23	23.40	8.72	2.99
FAD $r^*=0.6$	52.60	41.24	25.52	13.93	7.49
FAD $r^*=0.5$	47.88	38.74	27.25	16.50	12.79
FAD $r^*=0.4$	41.08	34.99	25.91	19.04	15.01

Table 6.2: Natural and robust accuracy (in %) for CIFAR-10 and CIFAR-100 against PGD ℓ_2-bounded attacks (50 steps) for different values of ε_{tr}.

Defense	Clean	$\varepsilon_{te} = 1$	$\varepsilon_{te} = 2$	$\varepsilon_{te} = 3$	$\varepsilon_{te} = 4$
CIFAR-10					
None	95.01	0.03	0.00	0.00	0.00
AT $\varepsilon_{tr}=0.5$	88.34	37.88	3.39	0.03	0.00
AT $\varepsilon_{tr}=1$	80.14	51.40	15.69	1.85	0.12
AT $\varepsilon_{tr}=2$	64.43	50.83	33.11	14.32	3.33
AT $\varepsilon_{tr}=3$	59.25	48.78	36.52	21.88	9.26
CIFAR-100					
None	76.87	0.01	0.01	0.01	0.01
AT $\varepsilon_{tr}=0.5$	63.68	15.58	1.42	0.12	0.02
AT $\varepsilon_{tr}=1$	57.97	22.33	4.94	0.82	0.08
AT $\varepsilon_{tr}=2$	49.73	29.04	12.76	4.40	1.11
AT $\varepsilon_{tr}=3$	37.41	27.11	17.63	8.97	3.63

Table 6.3: Natural and robust accuracy (in %) for ImageNet against BPDA ℓ_2-bounded attacks at different resolution scales. FAD models are fine-tuned with $r^* = 0.6$

Defense	Clean	$\varepsilon_{te} = 3$	$\varepsilon_{te} = 5$	$\varepsilon_{te} = 10$
ImageNet				
FAD $r^*=0.8$	56.61	41.86	28.84	5.92
FAD $r^*=0.7$	49.54	36.92	28.62	9.14
FAD $r^*=0.6$	43.58	31.86	26.77	12.16
FAD $r^*=0.5$	37.94	31.90	23.61	12.81
FAD $r^*=0.4$	31.51	26.44	22.79	14.10

Table 6.4: Natural and robust accuracy (in %) for ImageNet against PGD ℓ_2-bounded attacks (50 steps) for different values of ε_{tr}.

Defense	Clean	$\varepsilon_{te} = 3$	$\varepsilon_{te} = 5$	$\varepsilon_{te} = 10$
ImageNet				
None	69.79	0.00	0.00	0.00
AT $\varepsilon_{tr}=0.5$	65.48	8.00	0.82	0.01
AT $\varepsilon_{tr}=1$	62.32	18.76	4.60	0.07
AT $\varepsilon_{tr}=3$	53.12	31.00	17.73	2.51
AT $\varepsilon_{tr}=5$	45.59	31.01	21.69	5.88

should merge adjacent nodes or split nodes so as to modify the structure of the Image Graph. As can be seen in Figure 6.9, the adversary tends to create new relatively small yet strong (i.e., with notable color contrast) patches in crafted adversarial examples. This demonstrates that BPDA with additional inner cycles to accumulate gradients represents an appropriate testing framework for our settings. However, it is evident by comparing filtered images (top panel in Figure 6.9) with their adversarial counterparts (bottom panel in Figure 6.9) that the semantics of the image is not preserved in adversarial examples.

Figure 6.9: Filtered images (top) and adversarial examples resulting from BPDA ℓ_2-bounded attacks (bottom).

6.6.5 Data Augmentation with Filtered Images

As IGE removes textures, we would expect to improve robustness to common corruptions if filtered images were incorporated into the training set. To test this hypothesis, we trained a ResNet18 model f_{aug} on CIFAR-10 by augmenting the training set of natural images $\mathbf{x}$ with their corresponding filtered versions $\mathbf{x}^{r^*}$ (we set $r^* = 0.6$). The test set used for the evaluation is the original one (i.e., formed only by natural images). We denote with f_{nat} the natural model trained on the original training set. The results for f_{nat} and f_{aug} are reported in Tables 6.5 and 6.6, respectively. As we may expect, the accuracy of f_{aug} drops compared to that of f_{nat} (from roughly 95% to 93.3%). However, if we consider the performance of the two models under data corruption, f_{aug} is almost always more accurate than f_{nat}. In some cases, e.g., glass blur, the gap is remarkable. The code for applying data corruptions has been adapted from [Salman et al., 2020b].

Table 6.5: CIFAR-10. Accuracy (in %) under data corruptions with different levels of severity for f_{nat}.

	f_{nat}				
Corruption	**Level 1**	**Level 2**	**Level 3**	**Level 4**	**Level 5**
gaussian_noise	78.8	59.45	40.54	33.52	28.78
shot_noise	86.81	77.78	55.88	47.29	35.05
impulse_noise	85.39	74.64	64.77	42.31	26.59
glass_blur	53.34	53.26	59.27	41.63	47.08
defocus_blur	94.91	93.91	90.22	83.19	62.77
motion_blur	92.02	88.82	83.09	84.12	77.5
zoom_blur	94.25	92.71	90.53	85.05	77.86
fog	94.77	93.98	93.16	91.07	79.07
frost	93.41	91.46	88.59	87.35	83.55
snow	90.82	82.02	77.04	74.55	67.71
contrast	94.73	93.19	91.73	87.8	56.06
brightness	94.9	94.72	94.33	93.95	92.55
jpeg_compression	87.27	82.08	79.91	77.25	73.96
pixelate	92.24	87.91	82.65	64.62	44.4
elastic_transform	91.07	90.7	87.57	81.37	76.54

6.7 Conclusions

In this Chapter, we introduced an image filtering framework, dubbed Image-Graph Extractor, that explicitly removes textures without altering the semantics of the image. We showed that IGE can be integrated into pre-trained image classifiers as a defense against ℓ_p-bounded adversarial attacks. Moreover, when filtered images are exploited during training as data augmentation, robustness against common data corruptions can be also improved. Although the main focus is on the use of IGE for adversarial robustness, its flexible Image-Graph representation opens several directions for future research. For example, the Image-Graph representation can be leveraged to cast image classification as a graph classification problem. Additionally, novel data augmentation strategies based on perturbations of nodes can be defined so as to remove common shortcuts such as the background color.

Table 6.6: CIFAR-10. Accuracy (in %) under data corruptions with different levels of severity for f_{aug}.

	f_{aug}				
Corruption	**Level 1**	**Level 2**	**Level 3**	**Level 4**	**Level 5**
gaussian_noise	86.05	76.05	61.71	54.64	47.79
shot_noise	89.17	84.03	69.15	63.04	52.39
impulse_noise	84.34	73.92	63.44	43.09	28.72
glass_blur	85.27	85.56	86.34	76.37	78.08
defocus_blur	92.95	92.92	92.59	91.29	88.35
motion_blur	91.14	90.37	88.51	88.71	85.29
zoom_blur	92.87	93.09	92.87	92.23	91.28
fog	92.68	91.51	89.19	84.52	69.08
frost	92.0	91.0	89.03	88.44	85.49
snow	91.7	88.82	83.03	78.65	77.42
contrast	92.85	90.99	88.68	83.82	53.87
brightness	92.99	92.81	92.49	91.91	90.4
jpeg_compression	88.89	86.19	85.18	83.83	82.0
pixelate	91.86	91.49	90.82	89.94	86.69
elastic_transform	90.24	90.84	90.9	88.64	85.07

Chapter 7

Conclusions

Artificial Intelligence (AI) is nowadays ubiquitous in our everyday life and has revolutionalized how we interact with technology. For example, we can talk to virtual assistants embedded in our smartphones and vacuum cleaners, enjoy a ride on a self-driving taxi, and get better healthcare services thanks to AI-enabled personalized medicine. AI is also being increasingly adopted in sensitive domains to support high-stakes decisions, including justice, finance, and healthcare. Therefore, ensuring that these systems perform reliably and consistently with their expected behavior is a matter of the utmost importance. Unfortunately, this is still far from being a reality. Indeed, there have been countless reported cases in which AI has unintentionally caused harm to people in the form of discrimination against underrepresented groups, privacy breaches, or, in the worst scenario, even death. In light of these unfortunate events, the need to improve the reliability of AI technologies has become compelling. This gave rise to a new subfield in the AI ecosystem, usually referred to as *Trustworthy AI*, that has attracted a great deal of research interest in the past few years.

In this thesis, we focused on two important dimensions in the context of Trustworthy AI, i.e., interpretability and robustness, and on two application domains, i.e., Anomaly Detection and Computer Vision.

The first part of the thesis tackled the problem of interpretability in Anomaly Detection. In Chapter 3, we introduced interpretability methods for the interpretation of the Isolation Forest based on its peculiar inner structure. The global Depth-based Isolation Forest Feature Importance (DIFFI) method was designed to provide an explanation of the model as a whole, while its local counterpart Local-DIFFI was designed to provide explanations associated with individual predictions. Based on the global DIFFI method, a novel procedure for unsupervised feature selection was defined. Due to the lack of labeled datasets in many applications of Anomaly Detection, the unsupervised nature of the proposed feature selection technique is particularly useful from a practical point of view.

In the second part of the thesis, we addressed the task of robust image

classification with Convolutional Neural Networks (CNNs). In Chapter 5, we demonstrated that Adversarial Training does not efficiently exploit the model's capacity and induces a trivial dependence on color that may be harmful if simple color-based perturbations are introduced. We unveiled previously unknown properties of the feature maps of adversarially-trained CNNs, i.e., the presence of feature maps that are densely active for all samples in the dataset and the presence of redundancy. Moreover, we proved that, contrary to common belief, the latent representations of adversarially-trained CNNs are not inherently robust. The goal of Chapter 6 was to improve the robustness of pre-trained CNNs by acting on the input space. This was done by designing a novel defense based on an iterative image filtering procedure that successfully removes textures from images without sacrificing their semantic content. Besides improving robustness against adversarial attacks, the filtering procedure can also be employed to increase the accuracy under data corruptions.

The research work described in this thesis was conceived to enhance the reliability of Machine Learning models. By focusing on two application domains of great practical interest, we narrowed the gap that usually separates academia from the real world and provided tools that could benefit both researchers and practitioners. The road toward Trustworthy AI is complicated and will lead to success only with joint efforts from both worlds.

Appendix A

Additional Results on Adversarial Training

In this Appendix, we provide results on robust models trained with ℓ_∞-norm and results on always active feature maps and feature maps redundancy with different values of the hyperparameters τ_{dens} and τ_{sim}.

A.1 Densely Active Feature Maps

We assess the validity of the experiments described in Section 5.6.1 for different values of τ_{dens}. Specifically, we selected large values of τ_{dens} (i.e., 0.85, 0.9, 0.95, and 0.99) since we are interested in the analysis of spatially dense feature maps. Results are reported in Figures A.1 to A.6. Results on robust models trained with ℓ_∞-norm are shown in Figures A.7 to A.10. Similar observations to those discussed in Section 5.6.1 - for ℓ_2 models and $\tau_{dens} = 0.95$ - hold for the experiments provided here.

A.2 Feature Maps Redundancy

We assess the validity of the experiments described in Section 5.6.2 for different values of τ_{sim}. Specifically, we selected large values of τ_{sim} (i.e., 0.85, 0.9, 0.95, and 0.99) since we are interested in the analysis of highly correlated feature maps. Results are reported in Figures A.11 to A.16. Results on robust models trained with ℓ_∞-norm are shown in Figures A.17 to A.20. Similar observations to those discussed in Section 5.6.2 - for ℓ_2 models and $\tau_{sim} = 0.95$ - hold for the experiments provided here.

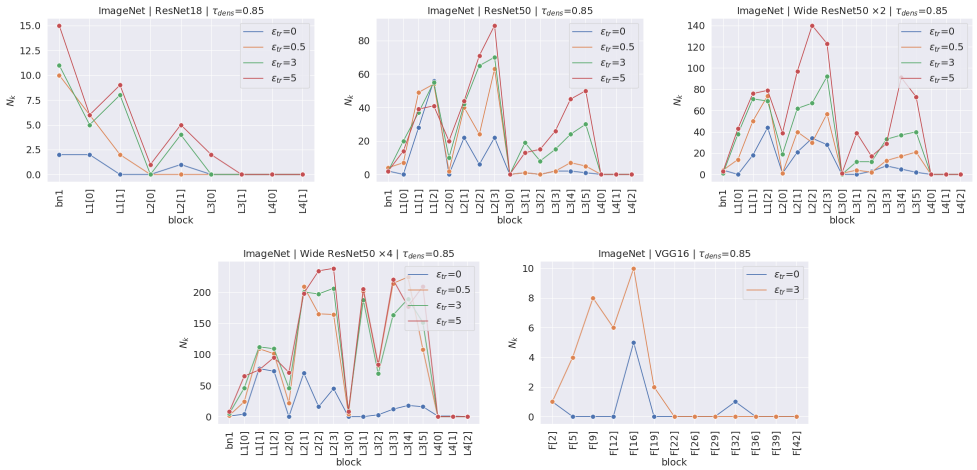

Figure A.1: Number of always densely active feature maps (at level $\tau_{dens} = 0.85$) for ImageNet models. Robust models are trained with ℓ_2-norm.

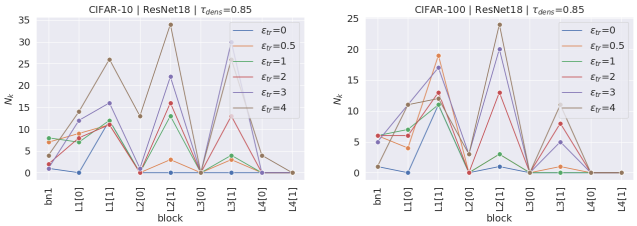

Figure A.2: Number of always densely active feature maps (at level $\tau_{dens} = 0.85$) for CIFAR-10 and CIFAR-100 models. Robust models are trained with ℓ_2-norm.

Figure A.3: Number of always densely active feature maps (at level $\tau_{dens} = 0.9$) for ImageNet models. Robust models are trained with ℓ_2-norm.

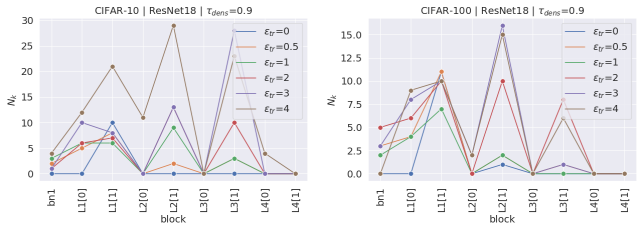

Figure A.4: Number of always densely active feature maps (at level $\tau_{dens} = 0.9$) for CIFAR-10 and CIFAR-100 models. Robust models are trained with ℓ_2-norm.

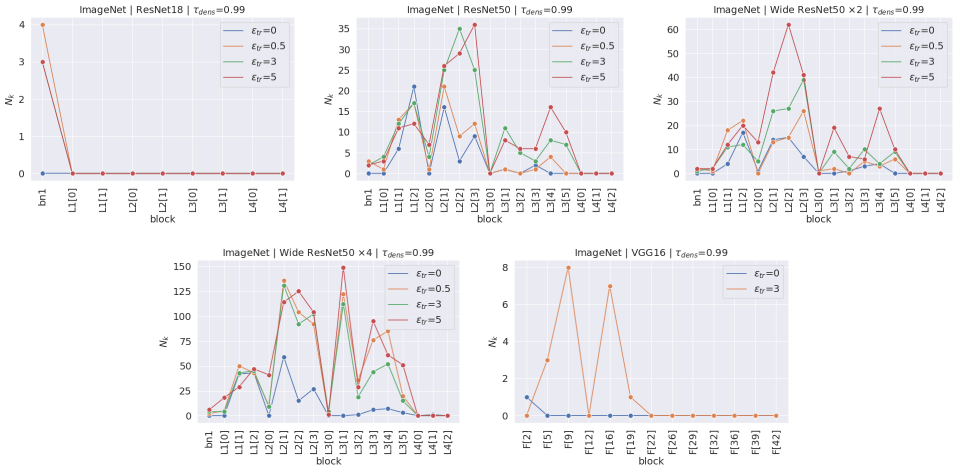

Figure A.5: Number of always densely active feature maps (at level $\tau_{dens} = 0.99$) for ImageNet models. Robust models are trained with ℓ_2-norm.

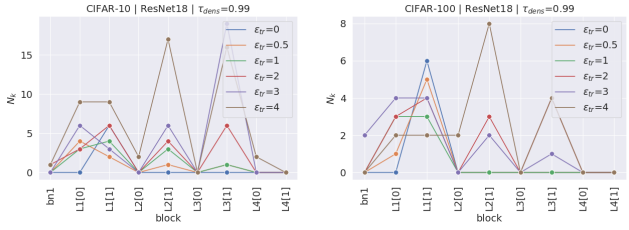

Figure A.6: Number of always densely active feature maps (at level $\tau_{dens} = 0.99$) for CIFAR-10 and CIFAR-100 models. Robust models are trained with ℓ_2-norm.

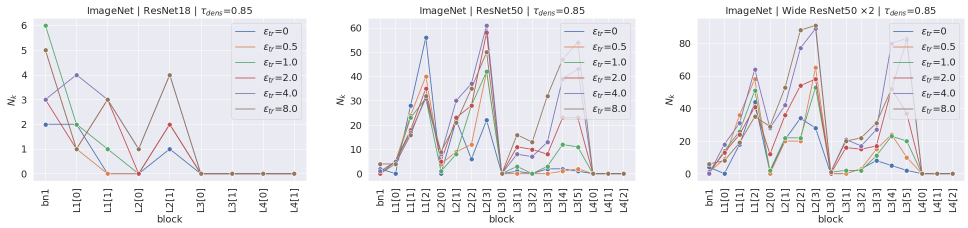

Figure A.7: Number of always densely active feature maps (at level $\tau_{dens} = 0.85$) for ImageNet models. Robust models are trained with ℓ_∞-norm.

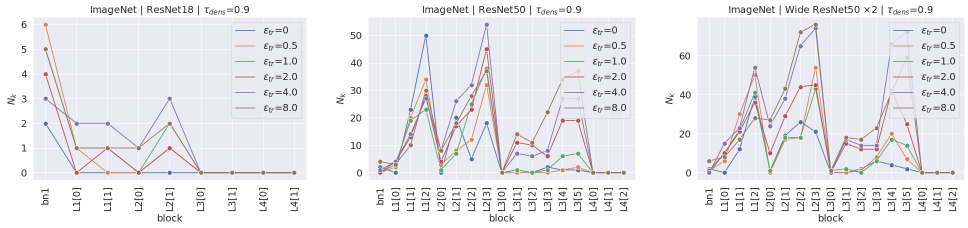

Figure A.8: Number of always densely active feature maps (at level $\tau_{dens} = 0.9$) for ImageNet models. Robust models are trained with ℓ_∞-norm.

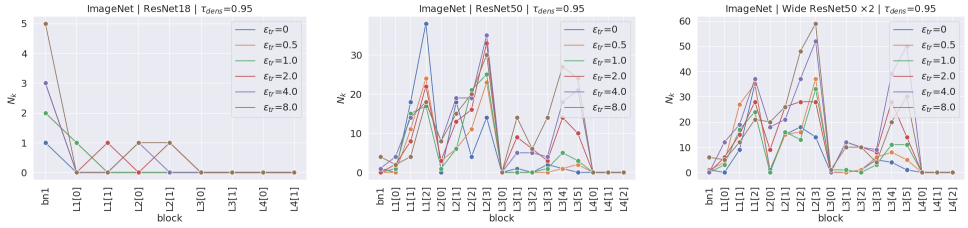

Figure A.9: Number of always densely active feature maps (at level $\tau_{dens} = 0.95$) for ImageNet models. Robust models are trained with ℓ_∞-norm.

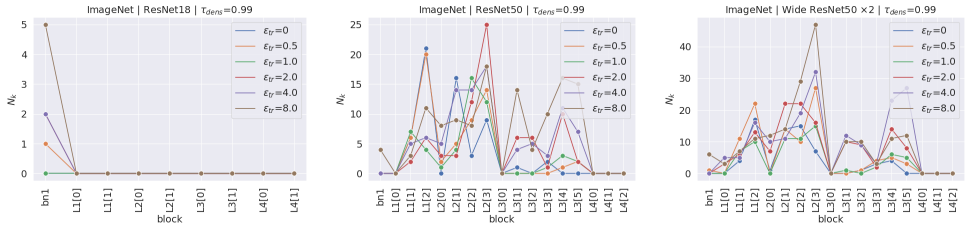

Figure A.10: Number of always densely active feature maps (at level $\tau_{dens} = 0.99$) for ImageNet models. Robust models are trained with ℓ_∞-norm.

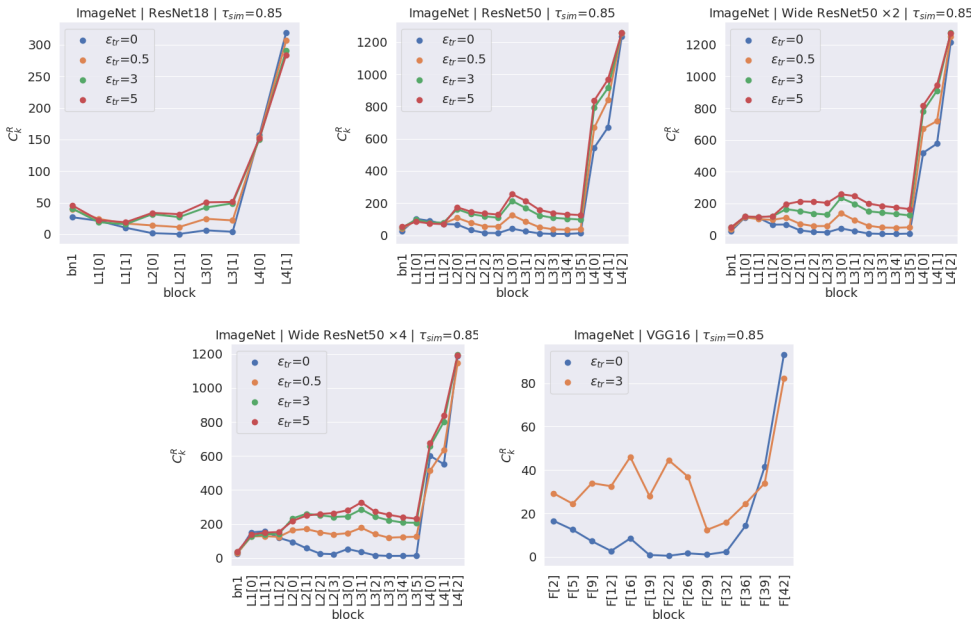

Figure A.11: Number of redundant feature maps (with $\tau_{sim} = 0.85$) for ImageNet models. Robust models are trained with ℓ_2-norm. Values are averaged over 5000 images randomly sampled from the test set.

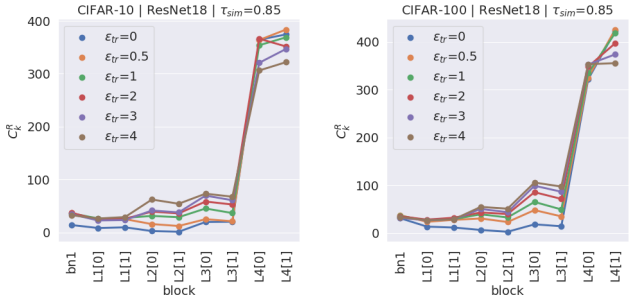

Figure A.12: Number of redundant feature maps (with $\tau_{sim} = 0.85$) for CIFAR-10 and CIFAR-100 models. Robust models are trained with ℓ_2-norm.

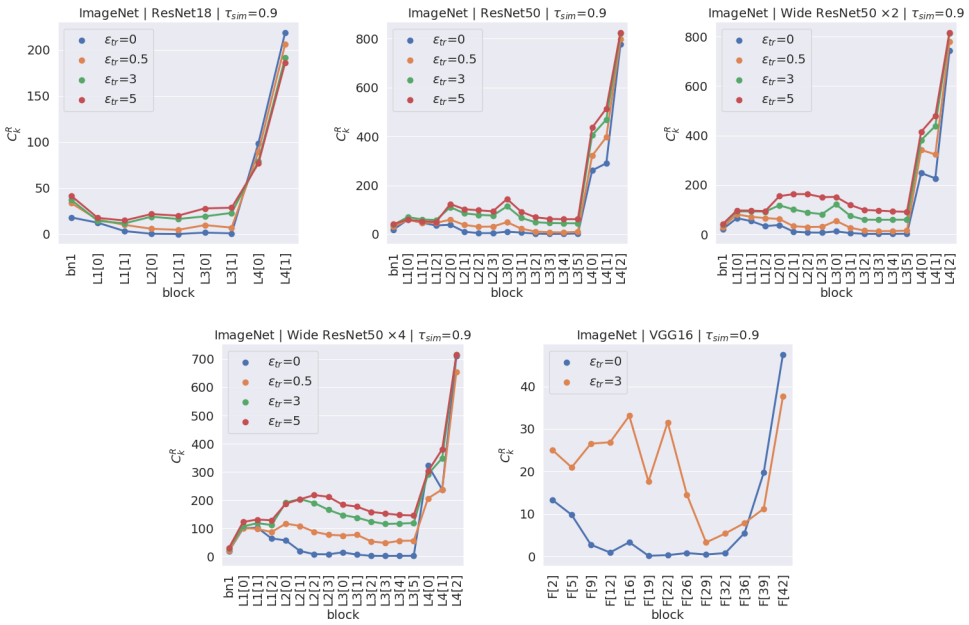

Figure A.13: Number of redundant feature maps (with $\tau_{sim} = 0.9$) for ImageNet models. Robust models are trained with ℓ_2-norm. Values are averaged over 5000 images randomly sampled from the test set.

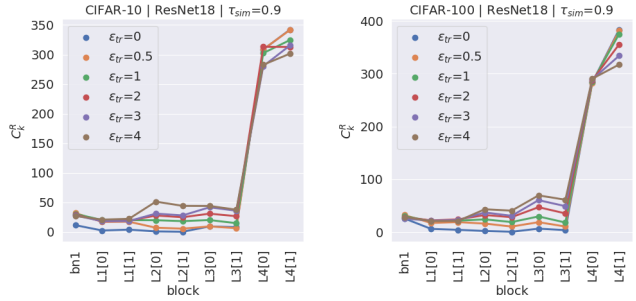

Figure A.14: Number of redundant feature maps (with $\tau_{sim} = 0.9$) for CIFAR-10 and CIFAR-100 models. Robust models are trained with ℓ_2-norm.

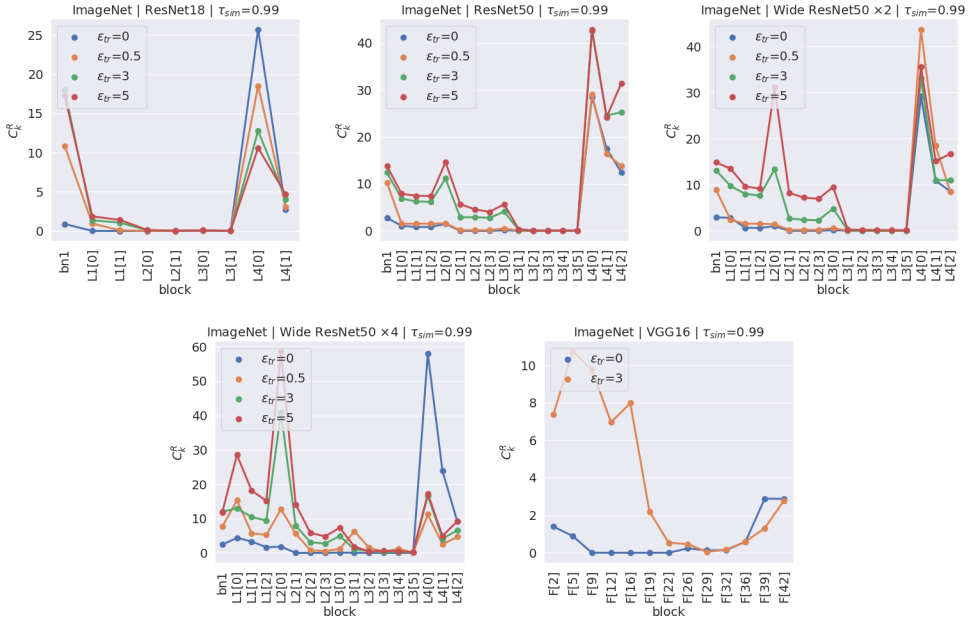

Figure A.15: Number of redundant feature maps (with $\tau_{sim} = 0.99$) for ImageNet models. Robust models are trained with ℓ_2-norm. Values are averaged over 5000 images randomly sampled from the test set.

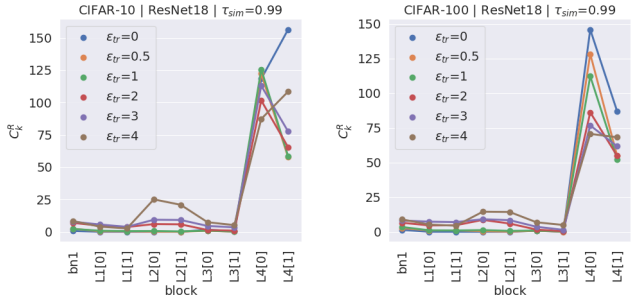

Figure A.16: Number of redundant feature maps (with $\tau_{sim} = 0.99$) for CIFAR-10 and CIFAR-100 models. Robust models are trained with ℓ_2-norm.

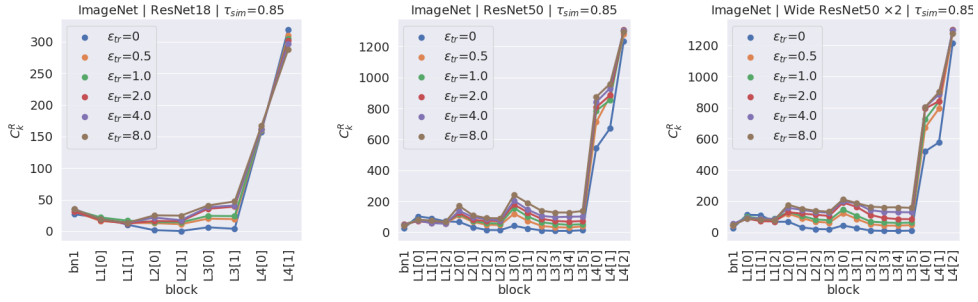

Figure A.17: Number of redundant feature maps (with $\tau_{sim} = 0.85$) for ImageNet models. Robust models are trained with ℓ_∞-norm. Values are averaged over 5000 images randomly sampled from the test set.

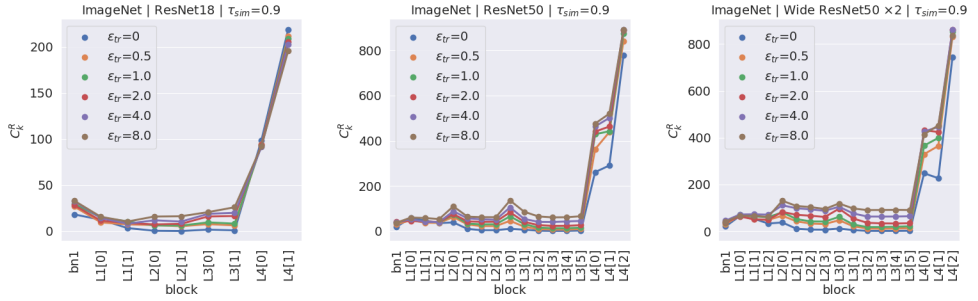

Figure A.18: Number of redundant feature maps (with $\tau_{sim} = 0.9$) for ImageNet models. Robust models are trained with ℓ_∞-norm. Values are averaged over 5000 images randomly sampled from the test set.

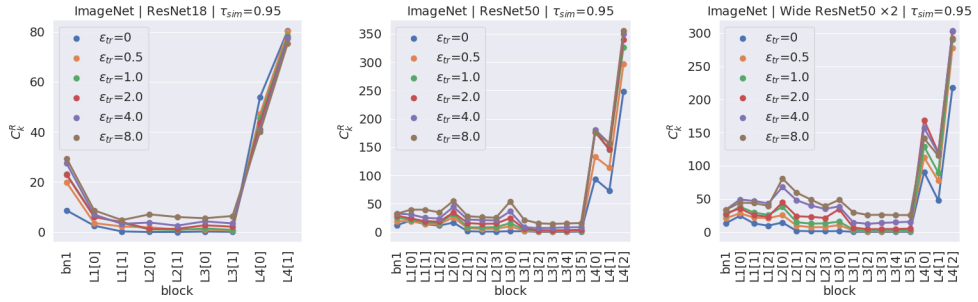

Figure A.19: Number of redundant feature maps (with $\tau_{sim} = 0.95$) for ImageNet models. Robust models are trained with ℓ_∞-norm. Values are averaged over 5000 images randomly sampled from the test set.

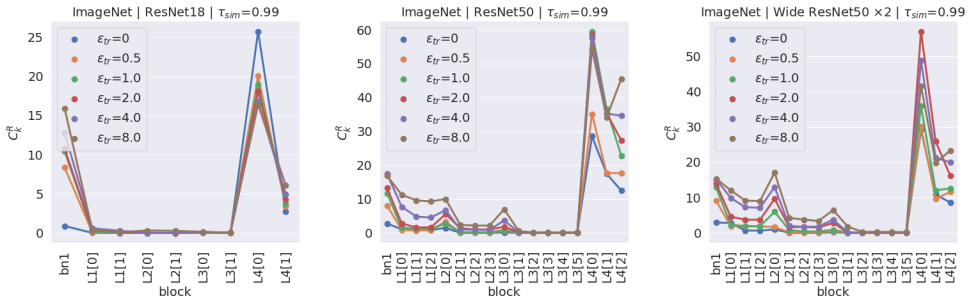

Figure A.20: Number of redundant feature maps (with $\tau_{sim} = 0.99$) for ImageNet models. Robust models are trained with ℓ_∞-norm. Values are averaged over 5000 images randomly sampled from the test set.

Table A.1: Natural accuracy on pixel-averaged images for ImageNet. Robust models are trained with ℓ_∞-norm.

Model	$\varepsilon_{tr} = 0$	$\varepsilon_{tr} = 0.5$	$\varepsilon_{tr} = 1.0$	$\varepsilon_{tr} = 2.0$	$\varepsilon_{tr} = 4.0$	$\varepsilon_{tr} = 8.0$
ResNet18	0.292	0.358	0.362	0.406	0.410	0.436
ResNet50	0.264	0.352	0.322	0.314	0.364	0.340
Wide ResNet50 ×2	0.204	0.342	0.326	0.332	0.390	0.300

A.3 Color Bias

In Table A.1 and Figure A.21, the results on pixel-averaged images and images with contours for robust models trained with ℓ_∞-norm are given. In Table A.2 are listed the accuracies on the ImageNet9 dataset for robust models trained with ℓ_∞-norm. Note that robust models consistently rely on global color information more than natural models. Moreover, they suffer from a larger average accuracy drop - and exhibit larger variance over different colors - when colored contours are added. These experiments confirm findings discussed in Section 5.6.4 and prove that the color bias is an inherent property of adversarially-trained models, independently of the metric used during training to craft adversarial examples.

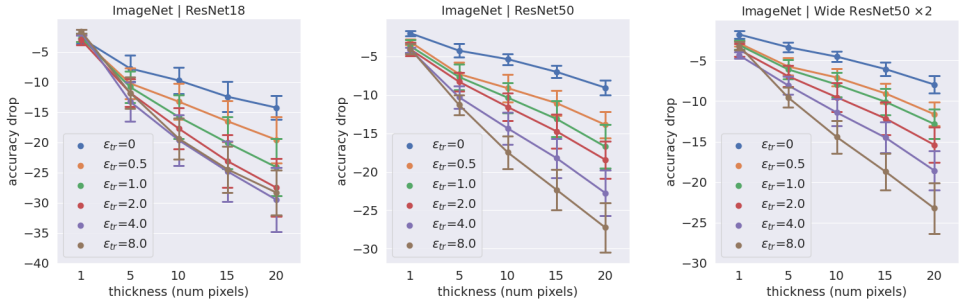

Figure A.21: Colored contours: natural accuracy drop with respect to the original test set on ImageNet. Circles represent the average accuracy drop over different colors (white, red, green, blue) and error bars indicate the corresponding standard deviation. Robust models are trained with ℓ_∞-norm.

Table A.2: Natural accuracy with different pixel-averaging methods for ImageNet-9. Accuracy is computed based on the 9 classes of the ImageNet-9 dataset. Robust models are trained with ℓ_∞-norm. The first line of each block gives the accuracy on the original images.

Model	**Transform**	$\varepsilon_{tr} = 0$	$\varepsilon_{tr} = 0.5$	$\varepsilon_{tr} = 1$	$\varepsilon_{tr} = 2$	$\varepsilon_{tr} = 4$	$\varepsilon_{tr} = 8$
ResNet18	-	93.68	92.62	91.83	89.73	85.36	77.33
	avg	2.42	5.73	6.84	9.38	8.40	8.05
	avg bg	2.57	6.89	8.22	10.12	8.15	8.52
	avg fg	1.11	2.17	2.47	5.38	5.31	3.68
	shape+color	8.69	13.46	15.85	16.00	17.80	17.33
ResNet50	-	95.83	95.80	95.38	94.10	91.90	86.67
	avg	0.77	6.54	8.30	8.47	7.04	6.22
	avg bg	1.09	7.56	9.01	9.48	8.00	7.19
	avg fg	0.05	2.44	4.02	4.02	3.14	2.67
	shape+color	12.30	13.06	15.41	15.88	19.43	17.58
Wide ResNet50 ×2	-	95.83	96.54	96.17	95.58	93.58	90.05
	avg	2.27	8.05	10.10	6.07	8.15	5.73
	avg bg	2.52	8.99	10.86	7.38	9.90	6.99
	avg fg	1.93	3.90	5.53	2.10	3.48	2.44
	shape+color	17.58	12.02	14.30	17.04	19.78	16.35

Appendix B

Details on the IGE Framework

In this Appendix, we provide further details on the IGE framework introduced in Chapter 6.

B.1 Implementation Details

In order to satisfy parallel and efficient image processing, we developed IGE in `CUDA/C++` and NVIDIA `Thrust`. The first stage of filtering is the extraction of connected components. We use the GPU-based CCL (Connected Component Labeling) algorithm adapted from [Playne and Hawick, 2018]. The output of CCL is a label matrix, where each location contains the identifier of the node/patch. After running CCL, we run a CUDA-kernel called `edge` which takes in input the label matrix `edge «< grid, block, block.x*block.y»>` and outputs the graph in a dataframe-like format. After this initialization process, the actual filtering procedure starts. At each step, the merge function is implemented in two phases: the first function has the objective of defining the candidates for merging, while the second function actually merges nodes. After the merge, the filtered image is reconstructed with the function `avg_color` that takes in input the label matrix and the original images and returns the filtered images by averaging color for each node. The graphs can be optionally saved on CSV files and elaborated with the NVIDIA library `libcudf` (https://github.com/rapidsai/cudf/). The python wrapper has been developed using `pybind11`.

B.2 Pre-processing

With high-dimensional images, we have found it beneficial to apply a denoising pre-processing that consists of partial removal of textures by the minimization of the following objective function

$$\arg\min_{\boldsymbol{\vartheta}} \mathcal{L}_{TV}(\boldsymbol{\vartheta}) + \mathcal{L}_{surf}(\boldsymbol{\vartheta}) + \lambda \|\mathbf{x} - \boldsymbol{\vartheta}\|_2^2$$

where $\mathbf{x}$ is the raw image, $\mathcal{L}_{TV}$ is the total variation loss and $\mathcal{L}_{surf}$ is the surface loss. The objective of $\mathcal{L}_{surf}$ loss is to minimize the number of changes in pixels' values. Indeed, total variation can be low even when the number of pixels' changes is very high. The change of pixels' color is defined with the step function $\mathbf{1}(\cdot)$, which is not differentiable. We approximate the step function with the differentiable surrogate:

$$\frac{1}{1+e^{-\alpha x}}$$

where $\alpha \gg 1$.

Bibliography

[Achanta et al., 2012] Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Süsstrunk, S. (2012). Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282.

[Akhtar and Mian, 2018] Akhtar, N. and Mian, A. (2018). Threat of adversarial attacks on deep learning in computer vision: A survey. *Ieee Access*, 6:14410–14430.

[Al-Rubaie and Chang, 2019] Al-Rubaie, M. and Chang, J. M. (2019). Privacy-preserving machine learning: Threats and solutions. *IEEE Security & Privacy*, 17(2):49–58.

[Alvarez-Melis and Jaakkola, 2018] Alvarez-Melis, D. and Jaakkola, T. S. (2018). On the robustness of interpretability methods. *arXiv preprint arXiv:1806.08049*.

[Anwar et al., 2017] Anwar, S., Hwang, K., and Sung, W. (2017). Structured pruning of deep convolutional neural networks. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 13(3):1–18.

[Apley, 2016] Apley, D. W. (2016). Visualizing the effects of predictor variables in black box supervised learning models. *arXiv preprint arXiv:1612.08468*.

[Arrieta et al., 2020] Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., et al. (2020). Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information fusion*, 58:82–115.

[Athalye et al., 2018] Athalye, A., Carlini, N., and Wagner, D. (2018). Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv:1802.00420*.

[Baevski et al., 2020] Baevski, A., Zhou, Y., Mohamed, A., and Auli, M. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460.

[Bai et al., 2021] Bai, Y., Zeng, Y., Jiang, Y., Xia, S.-T., Ma, X., and Wang, Y. (2021). Improving adversarial robustness via channel-wise activation suppressing. *arXiv preprint arXiv:2103.08307*.

[Barbariol et al., 2022] Barbariol, T., Chiara, F. D., Marcato, D., and Susto, G. A. (2022). A Review of Tree-Based Approaches for Anomaly Detection. *Control Charts and Machine Learning for Anomaly Detection in Manufacturing*, pages 149–185.

[Basu et al., 2018] Basu, S., Kumbier, K., Brown, J. B., and Yu, B. (2018). Iterative random forests to discover predictive and stable high-order interactions. *Proceedings of the National Academy of Sciences*, 115(8):1943–1948.

[Bau et al., 2017] Bau, D., Zhou, B., Khosla, A., Oliva, A., and Torralba, A. (2017). Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6541–6549.

[Ben-Gal, 2005] Ben-Gal, I. (2005). Outlier detection. In *Data mining and knowledge discovery handbook*, pages 131–146. Springer.

[Biggio et al., 2012] Biggio, B., Nelson, B., and Laskov, P. (2012). Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*.

[Bommasani et al., 2021] Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. (2021). On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

[Bottou et al., 2018] Bottou, L., Curtis, F. E., and Nocedal, J. (2018). Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311.

[Boulemtafes et al., 2020] Boulemtafes, A., Derhab, A., and Challal, Y. (2020). A review of privacy-preserving techniques for deep learning. *Neurocomputing*, 384:21–45.

[Bovens, 2007] Bovens, M. (2007). Analysing and assessing accountability: A conceptual framework 1. *European law journal*, 13(4):447–468.

[Breiman, 2001] Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

[Breunig et al., 2000] Breunig, M. M., Kriegel, H.-P., Ng, R. T., and Sander, J. (2000). Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104.

[Brown et al., 2020] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

[Brown et al., 2017] Brown, T. B., Mané, D., Roy, A., Abadi, M., and Gilmer, J. (2017). Adversarial patch. *arXiv preprint arXiv:1712.09665*.

[Cai et al., 2017] Cai, Z., He, X., Sun, J., and Vasconcelos, N. (2017). Deep learning with low precision by half-wave gaussian quantization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5918–5926.

[Carletti et al., 2019] Carletti, M., Masiero, C., Beghi, A., and Susto, G. A. (2019). Explainable Machine Learning in Industry 4.0: Evaluating Feature Importance in Anomaly Detection to Enable Root Cause Analysis. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pages 21–26. IEEE.

[Carmon et al., 2019] Carmon, Y., Raghunathan, A., Schmidt, L., Liang, P., and Duchi, J. C. (2019). Unlabeled data improves adversarial robustness. *arXiv preprint arXiv:1905.13736*.

[Casper et al., 2019] Casper, S., Boix, X., D'Amario, V., Guo, L., Schrimpf, M., Vinken, K., and Kreiman, G. (2019). Frivolous units: Wider networks are not really that wide. *arXiv preprint arXiv:1912.04783*.

[Chakraborty et al., 2018] Chakraborty, A., Alam, M., Dey, V., Chattopadhyay, A., and Mukhopadhyay, D. (2018). Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*.

[Chandola et al., 2009] Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58.

[Che et al., 2016] Che, Z., Purushotham, S., Khemani, R., and Liu, Y. (2016). Interpretable deep models for ICU outcome prediction. In *AMIA Annual Symposium Proceedings*, volume 2016, page 371. American Medical Informatics Association.

[Chen,] Chen, Angela (July 31st, . Ibm's watson gave unsafe recommendations for treating cancer. https://www.theverge.com/2018/7/26/17619382/ibms-watson-cancer-ai-healthcare-science. The Verge; accessed 02 January 2023.

[Chen et al., 2020a] Chen, P., Agarwal, C., and Nguyen, A. (2020a). The shape and simplicity biases of adversarially robust imagenet-trained cnns. *arXiv preprint arXiv:2006.09373*.

[Chen et al., 2020b] Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020b). A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

[Chen et al., 2020c] Chen, Y., Xie, Y., Song, L., Chen, F., and Tang, T. (2020c). A survey of accelerator architectures for deep neural networks. *Engineering*, 6(3):264–274.

[Chen et al., 2021] Chen, Z., Chen, D., Zhang, X., Yuan, Z., and Cheng, X. (2021). Learning Graph Structures with Transformer for Multivariate Time Series Anomaly Detection in IoT. *IEEE Internet of Things Journal*.

[Cisse et al., 2017] Cisse, M., Bojanowski, P., Grave, E., Dauphin, Y., and Usunier, N. (2017). Parseval networks: Improving robustness to adversarial examples. In *International Conference on Machine Learning*, pages 854–863. PMLR.

[Dang et al., 2014] Dang, X. H., Assent, I., Ng, R. T., Zimek, A., and Schubert, E. (2014). Discriminative features for identifying and interpreting outliers. In *2014 IEEE 30th international conference on data engineering*, pages 88–99. IEEE.

[De and Pedersen, 2021] De, K. and Pedersen, M. (2021). Impact of colour on robustness of deep neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21–30.

[DeepMind,] DeepMind. Accelerating the fight against plastic pollution. https://unfolded.deepmind.com/stories/accelerating-the-fight-against-plastic-pollution. Online; accessed 02 January 2023.

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

[Devlin et al., 2018] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[Devlin et al., 2019] Devlin, S., Singh, C., Murdoch, W. J., and Yu, B. (2019). Disentangled Attribution Curves for Interpreting Random Forests and Boosted Trees. *arXiv preprint arXiv:1905.07631*.

[Dhillon et al., 2018] Dhillon, G. S., Azizzadenesheli, K., Lipton, Z. C., Bernstein, J., Kossaifi, J., Khanna, A., and Anandkumar, A. (2018). Stochastic activation pruning for robust adversarial defense. *arXiv preprint arXiv:1803.01442*.

[Ding et al., 2019] Ding, G. W., Lui, K. Y. C., Jin, X., Wang, L., and Huang, R. (2019). On the sensitivity of adversarial robustness to input data distributions. In *ICLR (Poster)*.

[Ding and Fei, 2013] Ding, Z. and Fei, M. (2013). An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window. *IFAC Proceedings Volumes*, 46(20):12–17.

[Ding et al., 2015] Ding, Z., Fei, M., and Du, D. (2015). An online anomaly detection method for stream data using isolation principle and statistic histogram. *International Journal of Modeling, Simulation, and Scientific Computing*, 6(02):1550017.

[Doshi-Velez and Kim, 2017] Doshi-Velez, F. and Kim, B. (2017). Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

[Dosovitskiy et al., 2020] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

[Dziugaite et al., 2016] Dziugaite, G. K., Ghahramani, Z., and Roy, D. M. (2016). A study of the effect of jpg compression on adversarial images. *arXiv preprint arXiv:1608.00853*.

[Engstrom et al., 2019] Engstrom, L., Ilyas, A., Santurkar, S., Tsipras, D., Tran, B., and Madry, A. (2019). Adversarial robustness as a prior for learned representations. *arXiv preprint arXiv:1906.00945*.

[Fawzi et al., 2022] Fawzi, A., Balog, M., Huang, A., Hubert, T., Romera-Paredes, B., Barekatain, M., Novikov, A., R Ruiz, F. J., Schrittwieser, J., Swirszcz, G., et al. (2022). Discovering faster matrix multiplication algorithms with reinforcement learning. *Nature*, 610(7930):47–53.

[Fisher et al., 2018] Fisher, A., Rudin, C., and Dominici, F. (2018). Model Class Reliance: Variable importance measures for any machine learning model class, from the" Rashomon" perspective. *arXiv preprint arXiv:1801.01489*.

[Floridi, 2019] Floridi, L. (2019). Establishing the rules for building trustworthy ai. *Nature Machine Intelligence*, 1(6):261–262.

[Fredrikson et al., 2015] Fredrikson, M., Jha, S., and Ristenpart, T. (2015). Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1322–1333.

[Friedman, 2001] Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

[Gao et al., 2020] Gao, Y., Doan, B. G., Zhang, Z., Ma, S., Zhang, J., Fu, A., Nepal, S., and Kim, H. (2020). Backdoor attacks and countermeasures on deep learning: A comprehensive review. *arXiv preprint arXiv:2007.10760*.

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

[Georgescu et al., 2021] Georgescu, M.-I., Barbalau, A., Ionescu, R. T., Khan, F. S., Popescu, M., and Shah, M. (2021). Anomaly detection in video via self-supervised and multi-task learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12742–12752.

[Gilpin et al., 2018] Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., and Kagal, L. (2018). Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pages 80–89. IEEE.

[Goh et al., 2021] Goh, G., Cammarata, N., Voss, C., Carter, S., Petrov, M., Schubert, L., Radford, A., and Olah, C. (2021). Multimodal neurons in artificial neural networks. *Distill*, 6(3):e30.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

[Goodfellow et al., 2014] Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

[Gou et al., 2021] Gou, J., Yu, B., Maybank, S. J., and Tao, D. (2021). Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819.

[Gu et al., 2018] Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, G., Cai, J., et al. (2018). Recent advances in convolutional neural networks. *Pattern recognition*, 77:354–377.

[Guidotti et al., 2019] Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., and Pedreschi, D. (2019). A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):93.

[Gunning and Aha, 2019] Gunning, D. and Aha, D. (2019). DARPA's Explainable Artificial Intelligence (XAI) Program. *AI Magazine*, 40(2):44–58.

[Guo et al., 2017] Guo, C., Rana, M., Cisse, M., and Van Der Maaten, L. (2017). Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*.

[Guo et al., 2018] Guo, Y., Zhang, C., Zhang, C., and Chen, Y. (2018). Sparse dnns with improved adversarial robustness. *arXiv preprint arXiv:1810.09619*.

[Gupta et al., 2018] Gupta, N., Eswaran, D., Shah, N., Akoglu, L., and Faloutsos, C. (2018). Beyond outlier detection: LOOKOUT for pictorial explanation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 122–138. Springer.

[Guruswami and Rudra, 2008] Guruswami, V. and Rudra, A. (2008). Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on information theory*, 54(1):135–150.

[Han et al., 2015] Han, S., Mao, H., and Dally, W. J. (2015). Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.

[Han et al., 2020] Han, W., Zhang, Z., Zhang, Y., Yu, J., Chiu, C.-C., Qin, J., Gulati, A., Pang, R., and Wu, Y. (2020). Contextnet: Improving convolutional neural networks for automatic speech recognition with global context. *arXiv preprint arXiv:2005.03191*.

[Hardt et al., 2016] Hardt, M., Price, E., and Srebro, N. (2016). Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29.

[Hariri et al., 2019] Hariri, S., Kind, M. C., and Brunner, R. J. (2019). Extended isolation forest. *IEEE Transactions on Knowledge and Data Engineering*, 33(4):1479–1489.

[Hawkins, 1980] Hawkins, D. M. (1980). *Identification of outliers*, volume 11. Springer.

[He et al., 2020] He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738.

[He et al., 2016a] He, K., Zhang, X., Ren, S., and Sun, J. (2016a). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

[He et al., 2016b] He, K., Zhang, X., Ren, S., and Sun, J. (2016b). Identity mappings in deep residual networks. *arXiv:1603.05027*.

[He et al., 2006] He, X., Cai, D., and Niyogi, P. (2006). Laplacian score for feature selection. In *Advances in neural information processing systems*, pages 507–514.

[He et al., 2017] He, Y., Zhang, X., and Sun, J. (2017). Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1389–1397.

[He et al., 2003] He, Z., Xu, X., and Deng, S. (2003). Discovering cluster-based local outliers. *Pattern recognition letters*, 24(9-10):1641–1650.

[Hermann et al., 2020] Hermann, K., Chen, T., and Kornblith, S. (2020). The origins and prevalence of texture bias in convolutional neural networks. *Advances in Neural Information Processing Systems*, 33.

[Hillman, 2019] Hillman, N. L. (2019). The use of artificial intelligence in gauging the risk of recidivism. *Judges J.*, 58:36.

[Hornik et al., 1989] Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366.

[Huang et al., 2021] Huang, Y., Zhang, H., Shi, Y., Kolter, J. Z., and Anandkumar, A. (2021). Training certifiably robust neural networks with efficient local lipschitz bounds. *Advances in Neural Information Processing Systems*, 34.

[Ilyas et al., 2019] Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. (2019). Adversarial examples are not bugs, they are features. *arXiv preprint arXiv:1905.02175*.

[Ioffe and Szegedy, 2015] Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv:1502.03167*.

[Jumper et al., 2021] Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. (2021). Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589.

[Kannan et al., 2018] Kannan, H., Kurakin, A., and Goodfellow, I. (2018). Adversarial logit pairing. *arXiv preprint arXiv:1803.06373*.

[Karczmarek et al., 2020] Karczmarek, P., Kiersztyn, A., Pedrycz, W., and Al, E. (2020). K-Means-based isolation forest. *Knowledge-based systems*, 195:105659.

[Kaur et al., 2022] Kaur, D., Uslu, S., Rittichier, K. J., and Durresi, A. (2022). Trustworthy artificial intelligence: a review. *ACM Computing Surveys (CSUR)*, 55(2):1–38.

[Kiersztyn et al., 2021] Kiersztyn, A., Karczmarek, P., Kiersztyn, K., and Pedrycz, W. (2021). Detection and classification of anomalies in large data sets on the basis of information granules. *IEEE Transactions on Fuzzy Systems*.

[Kim and Doshi-Velez, 2021] Kim, B. and Doshi-Velez, F. (2021). Machine learning techniques for accountability. *AI Magazine*, 42(1):47–52.

[Knorr and Ng, 1998] Knorr, E. M. and Ng, R. T. (1998). Algorithms for mining distancebased outliers in large datasets. In *Proceedings of the international conference on very large data bases*, pages 392–403. Citeseer.

[Ko et al., 2022] Ko, K.-T., Lennartz, F., Mekhaiel, D., Guloglu, B., Marini, A., Deuker, D. J., Long, C. A., Jore, M. M., Miura, K., Biswas, S., et al. (2022). Structure of the malaria vaccine candidate pfs48/45 and its recognition by transmission blocking antibodies. *Nature Communications*, 13(1):1–11.

[Kolesnikov et al., 2019] Kolesnikov, A., Zhai, X., and Beyer, L. (2019). Revisiting self-supervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1920–1929.

[Kovacevic and Chebira, 2008] Kovacevic, J. and Chebira, A. (2008). *An introduction to frames*. Now Publishers Inc.

[Krizhevsky et al., 2009a] Krizhevsky, A., Hinton, G., et al. (2009a). Learning multiple layers of features from tiny images.

[Krizhevsky et al., 2009b] Krizhevsky, A., Nair, V., and Hinton, G. (2009b). Cifar-10 and cifar-100 datasets. *URl: https://www. cs. toronto. edu/kriz/cifar. html*, 6(1):1.

[Krizhevsky et al., 2009c] Krizhevsky, A., Nair, V., and Hinton, G. (2009c). Cifar-10 and cifar-100 datasets. *URl: https://www. cs. toronto. edu/kriz/cifar. html*, 6.

[Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105.

[Krizhevsky et al., 2017] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90.

[Kurakin et al., 2016] Kurakin, A., Goodfellow, I., and Bengio, S. (2016). Adversarial machine learning at scale. *arXiv:1611.01236*.

[Lam et al., 2022] Lam, R., Sanchez-Gonzalez, A., Willson, M., Wirnsberger, P., Fortunato, M., Pritzel, A., Ravuri, S., Ewalds, T., Alet, F., Eaton-Rosen,

Z., et al. (2022). Graphcast: Learning skillful medium-range global weather forecasting. *arXiv preprint arXiv:2212.12794*.

[Lambrecht and Tucker, 2019] Lambrecht, A. and Tucker, C. (2019). Algorithmic bias? an empirical study of apparent gender-based discrimination in the display of stem career ads. *Management science*, 65(7):2966–2981.

[LeCun et al., 2015] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444.

[Li et al., 2018] Li, O., Liu, H., Chen, C., and Rudin, C. (2018). Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[Li et al., 2019] Li, X., Wang, Y., Basu, S., Kumbier, K., and Yu, B. (2019). A Debiased MDI Feature Importance Measure for Random Forests. *arXiv preprint arXiv:1906.10845*.

[Li et al., 2012] Li, Z., Yang, Y., Liu, J., Zhou, X., and Lu, H. (2012). Unsupervised feature selection using nonnegative spectral analysis. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

[Liang and Huang, 2020] Liang, Y. and Huang, D. (2020). Large norms of cnn layers do not hurt adversarial robustness. *arXiv preprint arXiv:2009.08435*.

[Lim et al., 2021] Lim, B., Arık, S. Ö., Loeff, N., and Pfister, T. (2021). Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764.

[Lipton, 2018] Lipton, Z. C. (2018). The Mythos of Model Interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57.

[Liu et al., 2008] Liu, F. T., Ting, K. M., and Zhou, Z.-H. (2008). Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE.

[Liu et al., 2010] Liu, F. T., Ting, K. M., and Zhou, Z.-H. (2010). On detecting clustered anomalies using SCiForest. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 274–290. Springer.

[Liu et al., 2012] Liu, F. T., Ting, K. M., and Zhou, Z.-H. (2012). Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(1):3.

[Liu et al., 2021] Liu, H., Dai, Z., So, D. R., and Le, Q. V. (2021). Pay attention to mlps. *arXiv preprint arXiv:2105.08050*.

[Liu et al., 2022] Liu, H., Wang, Y., Fan, W., Liu, X., Li, Y., Jain, S., Liu, Y., Jain, A., and Tang, J. (2022). Trustworthy ai: A computational perspective. *ACM Transactions on Intelligent Systems and Technology*, 14(1):1–59.

[Liu et al., 2017] Liu, N., Shin, D., and Hu, X. (2017). Contextual outlier interpretation. *arXiv preprint arXiv:1711.10589*.

[Liu et al., 2020] Liu, Y., Ma, X., Bailey, J., and Lu, F. (2020). Reflection backdoor: A natural backdoor attack on deep neural networks. In *European Conference on Computer Vision*, pages 182–199. Springer.

[Lu et al., 2017] Lu, J., Sibai, H., Fabry, E., and Forsyth, D. (2017). No need to worry about adversarial examples in object detection in autonomous vehicles. *arXiv preprint arXiv:1707.03501*.

[Lundberg et al., 2018] Lundberg, S. M., Erion, G. G., and Lee, S.-I. (2018). Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*.

[Lundberg and Lee, 2017] Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pages 4765–4774.

[Ma et al., 2020] Ma, H., Ghojogh, B., Samad, M. N., Zheng, D., and Crowley, M. (2020). Isolation Mondrian forest for batch and online anomaly detection. In *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 3051–3058. IEEE.

[Ma et al., 2021] Ma, X., Wu, J., Xue, S., Yang, J., Zhou, C., Sheng, Q. Z., Xiong, H., and Akoglu, L. (2021). A comprehensive survey on graph anomaly detection with deep learning. *IEEE Transactions on Knowledge and Data Engineering*.

[Macha and Akoglu, 2018] Macha, M. and Akoglu, L. (2018). Explaining anomalies in groups with characterizing subspace rules. *Data Mining and Knowledge Discovery*, 32(5):1444–1480.

[Madry et al., 2017] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

[Mao et al., 2020] Mao, C., Gupta, A., Nitin, V., Ray, B., Song, S., Yang, J., and Vondrick, C. (2020). Multitask learning strengthens adversarial robustness. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 158–174. Springer.

[Mehrabi et al., 2021] Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., and Galstyan, A. (2021). A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 54(6):1–35.

[Melis and Jaakkola, 2018] Melis, D. A. and Jaakkola, T. (2018). Towards robust interpretability with self-explaining neural networks. In *Advances in Neural Information Processing Systems*, pages 7775–7784.

[Meneghetti et al., 2018] Meneghetti, L., Terzi, M., Del Favero, S., Susto, G. A., and Cobelli, C. (2018). Data-driven anomaly recognition for unsupervised model-free fault detection in artificial pancreas. *IEEE Transactions on Control Systems Technology*.

[Miao et al., 2018] Miao, X., Liu, Y., Zhao, H., and Li, C. (2018). Distributed online one-class support vector machine for anomaly detection over networks. *IEEE transactions on cybernetics*, 49(4):1475–1488.

[Molnar, 2022] Molnar, C. (2022). *Interpretable Machine Learning*. 2 edition.

[Moosavi-Dezfooli et al., 2017] Moosavi-Dezfooli, S.-M., Fawzi, A., Fawzi, O., and Frossard, P. (2017). Universal adversarial perturbations. *arXiv:1610.08401*.

[Moosavi-Dezfooli et al., 2016] Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. (2016). Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2574–2582.

[Moosavi-Dezfooli et al., 2018] Moosavi-Dezfooli, S.-M., Fawzi, A., Uesato, J., and Frossard, P. (2018). Robustness via curvature regularization, and vice versa.

[Murdoch et al., 2018] Murdoch, W. J., Liu, P. J., and Yu, B. (2018). Beyond word importance: Contextual decomposition to extract interactions from LSTMs. *arXiv preprint arXiv:1801.05453*.

[Nayak et al., 2021] Nayak, R., Pati, U. C., and Das, S. K. (2021). A comprehensive review on deep learning-based methods for video anomaly detection. *Image and Vision Computing*, 106:104078.

[Oreshkin et al., 2019] Oreshkin, B. N., Carpov, D., Chapados, N., and Bengio, Y. (2019). N-beats: Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv:1905.10437*.

[Özdenizci and Legenstein, 2021] Özdenizci, O. and Legenstein, R. (2021). Training adversarially robust sparse networks via bayesian connectivity sampling. In *International Conference on Machine Learning*, pages 8314–8324. PMLR.

[Pang et al., 2021] Pang, G., Shen, C., Cao, L., and Hengel, A. V. D. (2021). Deep learning for anomaly detection: A review. *ACM Computing Surveys (CSUR)*, 54(2):1–38.

[Pang et al., 2019] Pang, T., Xu, K., Du, C., Chen, N., and Zhu, J. (2019). Improving adversarial robustness via promoting ensemble diversity. In *International Conference on Machine Learning*, pages 4970–4979. PMLR.

[Papernot et al., 2016] Papernot, N., McDaniel, P., Wu, X., Jha, S., and Swami, A. (2016). Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 582–597. IEEE.

[Playne and Hawick, 2018] Playne, D. P. and Hawick, K. (2018). A new algorithm for parallel connected-component labelling on gpus. *IEEE Transactions on Parallel and Distributed Systems*, 29(6):1217–1230.

[Puggini and McLoone, 2018] Puggini, L. and McLoone, S. (2018). An enhanced variable selection and Isolation Forest based methodology for anomaly detection with OES data. *Engineering Applications of Artificial Intelligence*, 67:126–135.

[Purificato et al., 2022] Purificato, E., Lorenzo, F., Fallucchi, F., and De Luca, E. W. (2022). The use of responsible artificial intelligence techniques in the context of loan approval processes. *International Journal of Human–Computer Interaction*, pages 1–20.

[Qin et al., 2019] Qin, C., Martens, J., Gowal, S., Krishnan, D., Dvijotham, K., Fawzi, A., De, S., Stanforth, R., and Kohli, P. (2019). Adversarial robustness through local linearization. *arXiv preprint arXiv:1907.02610*.

[Quach,] Quach, Katyanna (October 28th, . Researchers made an openai gpt-3 medical chatbot as an experiment. it told a mock patient to kill themselves. https://www.theregister.com/2020/10/28/gpt3_medical_chatbot_experiment/. The Register; accessed 02 January 2023.

[Quinonero-Candela et al., 2008] Quinonero-Candela, J., Sugiyama, M., Schwaighofer, A., and Lawrence, N. D. (2008). *Dataset shift in machine learning*. Mit Press.

[Radford et al., 2021] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

[Ramaswamy et al., 2000] Ramaswamy, S., Rastogi, R., and Shim, K. (2000). Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 427–438.

[Ribeiro et al., 2016] Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

[Ribeiro et al., 2018] Ribeiro, M. T., Singh, S., and Guestrin, C. (2018). Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

[Ruder, 2017] Ruder, S. (2017). An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

[Rudin, 2019] Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215.

[Šabić et al., 2021] Šabić, E., Keeley, D., Henderson, B., and Nannemann, S. (2021). Healthcare and anomaly detection: using machine learning to predict anomalies in heart rate data. *AI & SOCIETY*, 36(1):149–158.

[Sabokrou et al., 2017] Sabokrou, M., Fayyaz, M., Fathy, M., and Klette, R. (2017). Deep-cascade: Cascading 3d deep neural networks for fast anomaly detection and localization in crowded scenes. *IEEE Transactions on Image Processing*, 26(4):1992–2004.

[Salman et al., 2020a] Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., and Madry, A. (2020a). Do adversarially robust imagenet models transfer better? *arXiv preprint arXiv:2007.08489*.

[Salman et al., 2020b] Salman, H., Ilyas, A., Engstrom, L., Vemprala, S., Madry, A., and Kapoor, A. (2020b). Unadversarial examples: Designing objects for robust vision. *arXiv preprint arXiv:2012.12235*.

[Scaman and Virmaux, 2018] Scaman, K. and Virmaux, A. (2018). Lipschitz regularity of deep neural networks: analysis and efficient estimation. *arXiv preprint arXiv:1805.10965*.

[Shafahi et al., 2019] Shafahi, A., Saadatpanah, P., Zhu, C., Ghiasi, A., Studer, C., Jacobs, D., and Goldstein, T. (2019). Adversarially robust transfer learning. *arXiv preprint arXiv:1905.08232*.

[Shaham et al., 2015] Shaham, U., Yamada, Y., and Negahban, S. (2015). Understanding adversarial training: Increasing local stability of neural nets through robust optimization. *arXiv preprint arXiv:1511.05432*.

[Shaham et al., 2018] Shaham, U., Yamada, Y., and Negahban, S. (2018). Understanding adversarial training: Increasing local stability of supervised models through robust optimization. *Neurocomputing*, 307:195–204.

[Sharif Razavian et al., 2014] Sharif Razavian, A., Azizpour, H., Sullivan, J., and Carlsson, S. (2014). CNN features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813.

[Sharma et al., 2005] Sharma, G., Wu, W., and Dalal, E. N. (2005). The ciede2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research & Application: Endorsed by Inter-Society Color Council, The Colour Group (Great Britain), Canadian Society for Color, Color Science Association of Japan, Dutch Society for the Study of Color, The Swedish Colour Centre Foundation, Colour Society of Australia, Centre Français de la Couleur*, 30(1):21–30.

[Shead,] Shead, Sam (December 29th, . Amazon's alexa assistant told a child to do a potentially lethal challenge. https://www.cnbc.com/2021/12/29/amazons-alexa-told-a-child-to-do-a-potentially-lethal-challenge.html. CNBC.com; accessed 02 January 2023.

[Shi et al., 2020] Shi, C., Holtz, C., and Mishne, G. (2020). Online adversarial purification based on self-supervised learning. In *International Conference on Learning Representations*.

[Shi et al., 2021] Shi, Y., Wang, Y., Wu, C., Yeh, C.-F., Chan, J., Zhang, F., Le, D., and Seltzer, M. (2021). Emformer: Efficient memory transformer based acoustic model for low latency streaming speech recognition. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6783–6787. IEEE.

[Shokri et al., 2017] Shokri, R., Stronati, M., Song, C., and Shmatikov, V. (2017). Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE.

[Simonyan et al., 2013] Simonyan, K., Vedaldi, A., and Zisserman, A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

[Simonyan and Zisserman, 2014] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[Song et al., 2017] Song, Y., Kim, T., Nowozin, S., Ermon, S., and Kushman, N. (2017). Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*.

[Staerman et al., 2019] Staerman, G., Mozharovskyi, P., Clémençon, S., and d'Alché Buc, F. (2019). Functional isolation forest. In *Asian Conference on Machine Learning*, pages 332–347. PMLR.

[Stephenson et al., 2021] Stephenson, C., Padhy, S., Ganesh, A., Hui, Y., Tang, H., and Chung, S. (2021). On the geometry of generalization and memorization in deep neural networks. *arXiv preprint arXiv:2105.14602*.

[Strobl et al., 2008] Strobl, C., Boulesteix, A.-L., Kneib, T., Augustin, T., and Zeileis, A. (2008). Conditional variable importance for random forests. *BMC bioinformatics*, 9(1):307.

[Stutz et al., 2018] Stutz, D., Hermans, A., and Leibe, B. (2018). Superpixels: An evaluation of the state-of-the-art. *Computer Vision and Image Understanding*, 166:1–27.

[Su et al., 2019] Su, J., Vargas, D. V., and Sakurai, K. (2019). One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841.

[Suri et al., 2019] Suri, N. M. R., Murty, M. N., and Athithan, G. (2019). *Outlier detection: techniques and applications*. Springer.

[Susto et al., 2011] Susto, G. A., Beghi, A., and De Luca, C. (2011). A virtual metrology system for predicting cvd thickness with equipment variables and qualitative clustering. In *ETFA2011*, pages 1–4. IEEE.

[Szegedy et al., 2013] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

[Tan et al., 2011] Tan, S. C., Ting, K. M., and Liu, T. F. (2011). Fast anomaly detection for streaming data. In *Twenty-Second International Joint Conference on Artificial Intelligence*.

[Tang et al., 2015] Tang, G., Pei, J., Bailey, J., and Dong, G. (2015). Mining multidimensional contextual outliers from categorical relational data. *Intelligent Data Analysis*, 19(5):1171–1192.

[Tang et al., 2002] Tang, J., Chen, Z., Fu, A. W.-C., and Cheung, D. W. (2002). Enhancing effectiveness of outlier detections for low density patterns. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 535–548. Springer.

[Tang et al., 2021] Tang, S., Gong, R., Wang, Y., Liu, A., Wang, J., Chen, X., Yu, F., Liu, X., Song, D., Yuille, A., Torr, P. H., and Tao, D. (2021). Robustart: Benchmarking robustness on architecture design and training techniques. *https://arxiv.org/pdf/2109.05211.pdf*.

[Taori et al., 2020] Taori, R., Dave, A., Shankar, V., Carlini, N., Recht, B., and Schmidt, L. (2020). Measuring robustness to natural distribution shifts in image classification. *Advances in Neural Information Processing Systems*, 33:18583–18599.

[Terzi et al., 2020] Terzi, M., Achille, A., Maggipinto, M., and Susto, G. A. (2020). Adversarial training reduces information and improves transferability. *arXiv preprint arXiv:2007.11259*.

[Tolomei et al., 2017] Tolomei, G., Silvestri, F., Haines, A., and Lalmas, M. (2017). Interpretable predictions of tree-based ensembles via actionable feature tweaking. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 465–474.

[Tramer et al., 2020] Tramer, F., Carlini, N., Brendel, W., and Madry, A. (2020). On adaptive attacks to adversarial example defenses. *arXiv preprint arXiv:2002.08347*.

[Tramèr et al., 2016] Tramèr, F., Zhang, F., Juels, A., Reiter, M. K., and Ristenpart, T. (2016). Stealing machine learning models via prediction {APIs}. In *25th USENIX security symposium (USENIX Security 16)*, pages 601–618.

[Tsipras et al., 2018] Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. (2018). Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*.

[Ukil et al., 2016] Ukil, A., Bandyoapdhyay, S., Puri, C., and Pal, A. (2016). Iot healthcare analytics: The importance of anomaly detection. In *2016 IEEE 30th international conference on advanced information networking and applications (AINA)*, pages 994–997. IEEE.

[Utrera et al., 2020] Utrera, F., Kravitz, E., Erichson, N. B., Khanna, R., and Mahoney, M. W. (2020). Adversarially-trained deep nets transfer better: Illustration on image classification. In *International Conference on Learning Representations*.

[Valdes et al., 2016] Valdes, G., Luna, J. M., Eaton, E., Simone, C. B., Ungar, L. H., and Solberg, T. D. (2016). MediBoost: a patient stratification tool for interpretable decision making in the era of precision medicine. *Scientific reports*, 6(1):1–8.

[van Esch et al., 2021] van Esch, P., Black, J. S., and Arli, D. (2021). Job candidates' reactions to ai-enabled job application processes. *AI and Ethics*, 1(2):119–130.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

[Verma and Rubin, 2018] Verma, S. and Rubin, J. (2018). Fairness definitions explained. In *2018 ieee/acm international workshop on software fairness (fairware)*, pages 1–7. IEEE.

[Wakabayashi,] Wakabayashi, Daisuke (March 19, . Self-driving uber car kills pedestrian in arizona, where robots roam. https://www.nytimes.com/2018/03/19/technology/uber-driverless-fatality.html. The New York Times; accessed 02 January 2023.

[Wang et al., 2019] Wang, H., Bah, M. J., and Hammad, M. (2019). Progress in outlier detection techniques: A survey. *Ieee Access*, 7:107964–108000.

[Wang et al., 2018a] Wang, L., Ding, G. W., Huang, R., Cao, Y., and Lui, Y. C. (2018a). Adversarial robustness of pruned neural networks.

[Wang et al., 2018b] Wang, X., He, X., Feng, F., Nie, L., and Chua, T.-S. (2018b). Tem: Tree-enhanced embedding model for explainable recommendation. In *Proceedings of the 2018 World Wide Web Conference*, pages 1543–1552.

[Wieringa, 2020] Wieringa, M. (2020). What to account for when accounting for algorithms: a systematic literature review on algorithmic accountability. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 1–18.

[Wolf et al., 2017] Wolf, M. J., Miller, K. W., and Grodzinsky, F. S. (2017). Why we should have seen that coming: comments on microsoft's tay "experiment," and wider implications. *The ORBIT Journal*, 1(2):1–12.

[Wong and Kolter, 2018] Wong, E. and Kolter, Z. (2018). Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5286–5295. PMLR.

[Wu et al., 2014] Wu, K., Zhang, K., Fan, W., Edwards, A., and Philip, S. Y. (2014). Rs-forest: A rapid density estimator for streaming anomaly detection. In *2014 IEEE International Conference on Data Mining*, pages 600–609. IEEE.

[Xiao et al., 2020] Xiao, K., Engstrom, L., Ilyas, A., and Madry, A. (2020). Noise or signal: The role of image backgrounds in object recognition. *arXiv preprint arXiv:2006.09994*.

[Xiao et al., 2018] Xiao, K. Y., Tjeng, V., Shafiullah, N. M., and Madry, A. (2018). Training for faster adversarial robustness verification via inducing relu stability. *arXiv preprint arXiv:1809.03008*.

[Xie et al., 2020] Xie, C., Tan, M., Gong, B., Wang, J., Yuille, A. L., and Le, Q. V. (2020). Adversarial examples improve image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 819–828.

[Xie et al., 2017] Xie, C., Wang, J., Zhang, Z., Ren, Z., and Yuille, A. (2017). Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991*.

[Xie et al., 2019] Xie, C., Wu, Y., Maaten, L. v. d., Yuille, A. L., and He, K. (2019). Feature denoising for improving adversarial robustness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 501–509.

[Xie and Yuille, 2019] Xie, C. and Yuille, A. (2019). Intriguing properties of adversarial training at scale. *arXiv preprint arXiv:1906.03787*.

[Yang et al., 2017] Yang, J., Zhou, C., Yang, S., Xu, H., and Hu, B. (2017). Anomaly detection based on zone partition for security protection of industrial cyber-physical systems. *IEEE Transactions on Industrial Electronics*, 65(5):4257–4267.

[Yang et al., 2011] Yang, Y., Shen, H. T., Ma, Z., Huang, Z., and Zhou, X. (2011). L2, 1-norm regularized discriminative feature selection for unsupervised. In *Twenty-second international joint conference on artificial intelligence*.

[Yang et al., 2020] Yang, Y.-Y., Rashtchian, C., Zhang, H., Salakhutdinov, R. R., and Chaudhuri, K. (2020). A closer look at accuracy vs. robustness. In *NeurIPS*.

[Ye et al., 2019] Ye, S., Xu, K., Liu, S., Cheng, H., Lambrechts, J.-H., Zhang, H., Zhou, A., Ma, K., Wang, Y., and Lin, X. (2019). Adversarial robustness vs. model compression, or both? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 111–120.

[Yu et al., 2002] Yu, D., Sheikholeslami, G., and Zhang, A. (2002). Findout: Finding outliers in very large datasets. *Knowledge and information Systems*, 4(4):387–412.

[Yu et al., 2022] Yu, J., Wang, Z., Vasudevan, V., Yeung, L., Seyedhosseini, M., and Wu, Y. (2022). Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*.

[Yuan et al., 2015] Yuan, Y., Ma, D., and Wang, Q. (2015). Hyperspectral anomaly detection by graph pixel selection. *IEEE transactions on cybernetics*, 46(12):3123–3134.

[Zagoruyko and Komodakis, 2016] Zagoruyko, S. and Komodakis, N. (2016). Wide residual networks. *arXiv preprint arXiv:1605.07146*.

[Zeiler and Fergus, 2014] Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer.

[Zhang et al., 2020] Zhang, H., Ren, Z., Xin, S., Liu, S., Lan, C., and Sun, X. (2020). A scale-adaptive positive selection algorithm based on B-cell immune mechanisms for anomaly detection. *Engineering Applications of Artificial Intelligence*, 94:103805.

[Zhang et al., 2019a] Zhang, H., Yu, Y., Jiao, J., Xing, E., El Ghaoui, L., and Jordan, M. (2019a). Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, pages 7472–7482. PMLR.

[Zhang et al., 2019b] Zhang, L., Zhao, J., and Li, W. (2019b). Online and Unsupervised Anomaly Detection for Streaming Data Using an Array of Sliding Windows and PDDs. *IEEE Transactions on Cybernetics*.

[Zhang et al., 2017] Zhang, M., Chen, C., Wo, T., Xie, T., Bhuiyan, M. Z. A., and Lin, X. (2017). SafeDrive: online driving anomaly detection from large-scale vehicle data. *IEEE Transactions on Industrial Informatics*, 13(4):2087–2096.

[Zhang and Zhu, 2019] Zhang, T. and Zhu, Z. (2019). Interpreting adversarially trained convolutional neural networks. In *International Conference on Machine Learning*, pages 7502–7511. PMLR.

[Zhao and Hryniewicki, 2018] Zhao, Y. and Hryniewicki, M. K. (2018). Xgbod: improving supervised outlier detection with unsupervised representation learning. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

[Zhao and Liu, 2007] Zhao, Z. and Liu, H. (2007). Spectral feature selection for supervised and unsupervised learning. In *Proceedings of the 24th international conference on Machine learning*, pages 1151–1157.

[Zhou et al., 2022] Zhou, Y., Ren, H., Li, Z., and Pedrycz, W. (2022). Anomaly detection based on a granular Markov model. *Expert Systems with Applications*, 187:115744.

[Zhu et al., 2019] Zhu, L., Liu, Z., and Han, S. (2019). Deep leakage from gradients. *Advances in neural information processing systems*, 32.

[Zügner et al., 2018] Zügner, D., Akbarnejad, A., and Günnemann, S. (2018). Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2847–2856.