Transferable adversarial patches : a potential threat for real-world computer vision algorithms

Pol Labarbarie

université PARIS-SACLAY

Transferable Adversarial Patches : A Potential Threat For Real-World Computer Vision Algorithms

Attaques Par Patch Transférables : Une Menace Potentielle Pour Les Algorithmes De Vision Par Ordinateur Opérant Dans Le Monde Réel

Thèse de doctorat de l'université Paris-Saclay

École doctorale n°580 : sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat : Informatique
Graduate School : Informatique et Sciences du Numérique. Référent : Faculté des sciences d'Orsay

Thèse préparée dans l'unité de recherche **Traitement de l'information et systèmes (Université Paris-Saclay, ONERA)** et à l'**IRT SystemX**, sous la direction de **Stéphane HERBIN**, chercheur à l'ONERA, et le co-encadrement de **Adrien CHAN-HON-TONG**, chercheur à l'ONERA et de **Milad LEYLI-ABADI** ingénieur chercheur à l'IRT SystemX.

Thèse soutenue à Paris-Saclay, le 18 Décembre 2024, par

Pol LABARBARIE

Composition du jury

Membres du jury avec voix délibérative

Anissa MOKRAOUI Professeure des Universités, Université Sorbonne Paris Nord	Présidente
Julien RABIN Maître de conférences (HDR), ENSICaen	Rapporteur
William PUECH Professeur des Universités, Université de Montpellier, CNRS	Rapporteur et examinateur
Stéphane CANU Professeur des Universités, INSA de Rouen, Normandie Université	Examinateur

THESE DE DOCTORAT
NNT : 2024UPASG084

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre : Attaques Par Patch Transférables : Une Menace Potentielle Pour Les Algorithmes De Vision Par Ordinateur Opérant Dans Le Monde Réel

Mots clés : Intelligence Artificielle, Fiabilité de l'IA, Apprentissage Profond

Résumé : Les réseaux de neurones profonds offrent aujourd'hui des performances inégalées notamment pour les fonctions de vision par ordinateur comme par exemple la classification d'images, la détection d'objets et la segmentation sémantique. Malgré ces avancées, les modèles d'apprentissage profond présentent des vulnérabilités qui peuvent être exploitées par des agents malveillants pour induire des comportements dangereux de la part des modèles d'IA. Une des menaces, appelée attaque par *patch*, consiste à introduire dans la scène un objet texturé pour duper le modèle. Par exemple, un patch placé sur un panneau stop peut amener le réseau à le classer à tort comme étant un panneau de limitation de vitesse. Ce type d'attaque soulève d'importants problèmes de sécurité pour les systèmes de vision par ordinateur opérant dans le monde physique. Dans cette thèse, nous étudions si un tel patch peut perturber un système physique dans des conditions d'attaque réalistes, i.e., sans connaissance préalable sur le système ciblé.

Bien que de nombreuses attaques par patch aient été proposées dans la littérature, il n'existe pas, à notre connaissance, de travail qui décrit les caractéristiques essentielles qualifiant une attaque par patch de critique. L'une de nos contributions est la définition de ce que serait une attaque par patch critique. Pour être qualifié de critique, une attaque par patch doit vérifier deux critères essentiels. Tout d'abord, le patch doit être robustes à des transformations physiques, ce qui est résumé par la notion de physicalité du patch. Ensuite, le patch doit être transférable, c'est-à-dire que le patch a la capacité de duper avec succès un réseau sans posséder aucune connaissance préalable sur celui-ci. La transférabilité de l'attaque est un facteur clé, car les systèmes physiques déployés par les entreprises sont souvent opaques ou inconnus. Bien que la physicalité des patchs ait été développée et améliorée par de nombreux travaux, la transférabilité des patchs reste faible et peu de méthode propose de l'améliorer.

Afin de créer une attaque par patch transférable pour une grande variété de classifieurs d'images, nous proposons une nouvelle méthode de conception des patchs. Cette méthode repose sur l'utilisation de la distance de Wasserstein, distance définie entre deux mesures de probabilité. Notre patch est appris en minimisant la distance de Wasserstein entre la distribution des caractéristiques des images corrompues par notre patch et la distribution des caractéristiques d'images d'une classe cible préalablement choisie. Une fois appris et placé dans la scène, notre patch induit plusieurs réseaux à prédire la classe de la distribution ciblée. Nous montrons qu'un tel patch est transférable et peut être implémenté dans le monde physique afin de perturber des classifieurs d'images sans aucune connaissance sur ceux-ci.

Afin d'avantage caractériser la potentielle menace des attaques par patch, nous proposons d'étudier leur transférabilité quand ceux-ci sont développer pour duper des détecteurs d'objets. Les détecteurs d'objets sont des modèles plus complexes que les classifieurs d'objets et sont souvent plus utilisés dans les systèmes opérant dans le monde physique. Nous étudions plus particulièrement les attaques par patch dites *cape d'invisibilité*, un type particulier de patchs conçus pour inhiber la détection d'objets lorsqu'ils leur sont appliqués dessus. Nos résultats révèlent que le protocole dévaluation utilisé dans la littérature comporte plusieurs problèmes rendant l'évaluation de ces patchs incorrecte. Pour y remédier, nous introduisons un problème de substitution qui garantit que le patch produit supprime bien la bonne détection de l'objet que nous souhaitons attaquer. En utilisant ce nouveau processus d'évaluation, nous montrons que les attaques par patch de la littérature ne parviennent pas à inhiber la détection d'objets limitant ainsi leur criticité.

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Title : Transferable Adversarial Patches :
A Potential Threat For Real-World Computer Vision Algorithms
Keywords : Trustworthiness of AI, Deep Learning, Artificial Intelligence

Abstract : The use of Deep Neural Networks has revolutionized the field of computer vision, leading to significant performance improvement in tasks such as image classification, object detection, and semantic segmentation. Despite these breakthroughs, Deep Learning systems have exhibited vulnerabilities that malicious entities can exploit to induce harmful behavior in AI models. One of the threats is adversarial patch attacks, disruptive objects that often resemble stickers and are designed to deceive models when placed in a real-world scene. For example, a patch on a stop sign may sway the network to misclassify it as a speed limit sign. This type of attack raises significant safety issues for computer vision systems operating in the real world. In this thesis, we study if such a patch can disrupt a real-world system without prior knowledge concerning the targeted system.

Even though numerous patch attacks have been proposed in the literature, no work in literature describes the prerequisites of a critical patch. One of our contributions is to propose a definition of what may be a critical adversarial patch. To be characterized as critical, adversarial patch attacks must meet two essential criteria. They must be robust to physical transformations summarized by the notion of patch physicality, and they must exhibit transferability among networks, meaning the patch can successfully fool networks without possessing any knowledge about the targeted system. Transferability is an essential prerequisite for a critical patch, as the targeted real-world system is usually protected and inaccessible from the outside. Although patch physicality has been developed and improved through multiple works, the transferability of patches remains a challenge.

To address the challenge of attack transferability among image classifiers, we introduce a new adversarial patch attack based on the Wasserstein distance, which computes the distance between two probability distributions. We exploit the Wasserstein distance to alter the feature distribution of a set of corrupted images to match another feature distribution from images of a target class. When placed in the scene, our patch causes various state-of-the-art networks to output the class chosen as the target distribution. We show that our patch is more transferable than previous patches and can be implemented in the real world to deceive real-world image classifiers.

In addition to our work on classification networks, we conduct a study on patch transferability against object detectors, as these systems may be more often involved in real-world systems. We focus on invisible cloak patches, a particular type of patches that are designed to hide objects when applied to them. Our findings reveal several significant flaws in the current evaluation protocol, which is used to assess the effectiveness of these patches. To address these flaws, we introduce a surrogate problem that ensures that the produced patch is suppressing the object we want to attack. We show that state-of-the-art patches against object detectors fail to hide objects from being detected, limiting the current patch criticality against real-world systems.

Abstract

The use of Deep Neural Networks has revolutionized the field of computer vision, leading to significant performance improvement in tasks such as image classification, object detection, and semantic segmentation. Despite these breakthroughs, Deep Learning systems have exhibited vulnerabilities that malicious entities can exploit to induce harmful behavior in AI models. One of the threats is adversarial patch attacks, disruptive objects that often resemble stickers and are designed to deceive models when placed in a real-world scene. For example, a patch on a stop sign may sway the network to misclassify it as a speed limit sign. This type of attack raises significant safety issues for computer vision systems operating in the real world. In this thesis, we study if such a patch can disrupt a real-world system without prior knowledge concerning the targeted system.

Even though numerous patch attacks have been proposed in the literature, no work in literature describes the prerequisites of a critical patch. One of our contributions is to propose a definition of what may be a critical adversarial patch. To be characterized as critical, adversarial patch attacks must meet two essential criteria. They must be robust to physical transformations summarized by the notion of patch physicality, and they must exhibit transferability among networks, meaning the patch can successfully fool networks without possessing any knowledge about the targeted system. Transferability is an essential prerequisite for a critical patch, as the targeted real-world system is usually protected and inaccessible from the outside. Although patch physicality has been developed and improved through multiple works, the transferability of patches remains a challenge.

To address the challenge of attack transferability among image classifiers, we introduce a new adversarial patch attack based on the Wasserstein distance, which computes the distance between two probability distributions. We exploit the Wasserstein distance to alter the feature distribution of a set of corrupted images to match another feature distribution from images of a target class. When placed in the scene, our patch causes various state-of-the-art networks to output the class chosen as the target distribution. We show that our patch is more transferable than previous patches and can be implemented in the real world to deceive real-world image classifiers.

In addition to our work on classification networks, we conduct a study on patch transferability against object detectors, as these systems may be more often involved in real-world systems. We focus on invisible cloak patches, a particular type of patches that are designed to hide objects when applied to them. Our findings reveal several significant flaws in the current evaluation protocol, which is used to assess the effectiveness of these patches. To address these flaws, we introduce a surrogate problem that ensures that the produced patch is suppressing the object we want to attack. We show that state-of-the-art patches against object detectors fail to hide objects from being detected, limiting the current patch criticality against real-world systems.

Résumé

Les réseaux de neurones profonds offrent aujourd'hui des performances inégalées notamment pour les fonctions de vision par ordinateur comme par exemple la classification d'images, la détection d'objets et la segmentation sémantique. Malgré ces avancées, les modèles d'apprentissage profond présentent des vulnérabilités qui peuvent être exploitées par des agents malveillants pour induire des comportements dangereux de la part des modèles d'IA. Une des menaces, appelée attaque par *patch*, consiste à introduire dans la scène un objet texturé pour duper le modèle. Par exemple, un patch placé sur un panneau stop peut amener le réseau à le classer à tort comme étant un panneau de limitation de vitesse. Ce type d'attaque soulève d'importants problèmes de sécurité pour les systèmes de vision par ordinateur opérant dans le monde physique. Dans cette thèse, nous étudions si un tel patch peut perturber un système physique dans des conditions d'attaque réalistes, i.e., sans connaissance préalable sur le système ciblé.

Bien que de nombreuses attaques par patch aient été proposées dans la littérature, il n'existe pas, à notre connaissance, de travail qui décrit les caractéristiques essentielles qualifiant une attaque par patch de critique. L'une de nos contributions est la définition de ce que serait une attaque par patch critique. Pour être qualifié de critique, une attaque par patch doit vérifier deux critères essentiels. Tout d'abord, le patch doit être robustes à des transformations physiques, ce qui est résumé par la notion de physicalité du patch. Ensuite, le patch doit être transférable, c'est-à-dire que le patch a la capacité de duper avec succès un réseau sans posséder aucune connaissance préalable sur celui-ci. La transférabilité de l'attaque est un facteur clé, car les systèmes physiques déployés par les entreprises sont souvent opaques ou inconnus. Bien que la physicalité des patchs ait été développée et améliorée par de nombreux travaux, la transférabilité des patchs reste faible et peu de méthode propose de l'améliorer.

Afin de créer une attaque par patch transférable pour une grande variété de classifieurs d'images, nous proposons une nouvelle méthode de conception des patchs. Cette méthode repose sur l'utilisation de la distance de Wasserstein, distance définie entre deux mesures de probabilité. Notre patch est appris en minimisant la distance de Wasserstein entre la distribution des caractéristiques des images corrompues par notre patch et la distribution des caractéristiques d'images d'une classe cible préalablement choisie. Une fois appris et placé dans la scène, notre patch induit plusieurs réseaux à prédire la classe de la distribution ciblée. Nous montrons qu'un tel patch est transférable et peut être implémenté dans le monde physique afin de perturber des classifieurs d'images sans aucune connaissance sur ceux-ci.

Afin d'avantage caractériser la potentielle menace des attaques par patch, nous proposons d'étudier leur transférabilité quand ceux-ci sont développer pour duper des détecteurs d'objets. Les détecteurs d'objets sont des modèles plus complexes que les classifieurs d'objets et sont souvent plus utilisés dans les systèmes opérant dans le monde physique. Nous étudions plus particulièrement les attaques par patch dites *cape d'invisibilité*, un

type particulier de patchs conçus pour inhiber la détection d'objets lorsqu'ils leur sont appliqués dessus. Nos résultats révèlent que le protocole dévaluation utilisé dans la littérature comporte plusieurs problèmes rendant l'évaluation de ces patchs incorrecte. Pour y remédier, nous introduisons un problème de substitution qui garantit que le patch produit supprime bien la bonne détection de l'objet que nous souhaitons attaquer. En utilisant ce nouveau processus d'évaluation, nous montrons que les attaques par patch de la littérature ne parviennent pas à inhiber la détection d'objets limitant ainsi leur criticité.

Acknowledgments

Cette thèse et ce manuscrit n'auraient jamais vu le jour sans le soutien, l'aide et la relecture attentive de nombreuses personnes que je tiens à remercier.

Tout d'abord, je remercie William Puech et Julien Rabin d'avoir accepté de rapporter ma thèse. Merci pour votre regard et expertise sur mon travail et merci pour vos retours sur mon manuscrit qui me permettent d'améliorer sa qualité. Je tiens également à remercier Anissa Mokraoui et Stéphane Canu pour leur présence dans mon jury de thèse.

Ensuite, je tiens à remercier mes encadrants. Ce génial trio d'encadrants a beaucoup apporté à cette thèse et à moi-même. Stéphane, le doyen du groupe, merci beaucoup pour ton immense expertise en vision et deep learning, ton scepticisme scientifique qui permet toujours de se poser les bonnes questions pour aller plus loin dans son travail, ta sagesse qui permet toujours de voir plus loin que le bout de son nez et d'avoir une vision scientifique à long terme plus cohérente et intéressante. Merci également de m'avoir transmis les us et coutumes des conférences prestigieuses. Milad, merci beaucoup pour ton expertise à la fois académique et industrielle. Tes conseils et tes relectures attentives de mes articles et de mon manuscrit ont contribué significativement à l'obtention de meilleures versions. Les importants petits détails ont toujours été peaufinés grâce à toi. Adrien, tu m'as tellement apporté. Merci beaucoup pour ta joie de vivre inébranlable et ton enthousiasme qui permettent d'être reboosté dans les moments de down. Ton enthousiasme m'a également permis d'avoir une grande liberté dans les sujets que je voulais explorer ou les directions que je voulais donner à ma thèse. Tu as toujours été à fond pour explorer des nouveaux sujets même si en pratique, je suis quand même resté proche du sujet de départ. Merci énormément pour ta réactivité et ta disponibilité, toujours présent pour me donner ton avis sur un code, une idée algorithmique ou un papier. Je garderai tellement de bons souvenirs comme par exemple : les lancements de papier de conférence 10 jours avant la deadline pour finalement revenir sur notre décision car bien sûr c'était un peu ambitieux, les appels téléphoniques le week-end ou tard le soir ou, le mieux, à la réception des reviews, les conversations slack sur n'importe quel sujet, le champagne à l'acceptation de notre papier à ICLR, le prêt de ton ancien vélo (un peu cassé, mais qui roule) pour aller au sport, ta capacité à rédiger des V0 de papier ou de document à une vitesse qui m'est encore inconcevable à une époque où chatgpt n'existait pas encore et bien d'autres bons souvenirs. Merci, à vous trois, vous aurez sans aucun doute une influence sur le chercheur que je suis en train de devenir.

Merci également à Arthur Leclaire, tout d'abord en tant que professeur en master et ensuite en tant qu'encadrant de stage. Mes amis de master se moquent encore de la fois où j'ai reçu ma copie de DS et que je « boudais », car j'avais été amputé de plusieurs points sur ma copie, car je n'avais pas assez justifié mes réponses pour autant correctes. Arthur venait de m'apprendre ce qu'est la rigueur et pour ça merci. Un merci également de m'avoir donné l'opportunité de faire un stage sur le transport optimal. Même si ce stage n'a pas été glorieux de ma part, cela m'a servi par la suite.

Je voudrais maintenant remercier les doctorants de l'ONERA. Merci aux « vieux » doctorants, Maxime, Marius, Quentin, Guillaume, Alexis et Philippe pour les parties de tarot et ensuite les parties de coinche. Merci à Kevin pour les pâtisseries. Je souhaite à Maxime de tout mon cœur que Clermont arrive dans les années qui suivent à se maintenir en top 14. Courage à lui. Pour ce qui est du tarot et de la coinche, malheureusement, les petits nouveaux auront eu raison de nous avec leurs jeux de société trop évolués pour nous, enfin pour moi. Un merci à eux, Clément, Dao, Louis, Liam, et David. Un merci particulier à Marie-Ange de m'avoir tenu au courant des différentes histoires de l'ONERA malgré ma présence variable, et merci pour le dynamisme qu'elle apporte au groupe de doctorants. Un merci également aux permanents de l'ONERA, Philippe, Baptiste, Pauline, Aurélien, Anthelme, Pierre et finalement Martial d'être un chef d'équipe autant attentionné et à l'écoute.

Je me souviens au début du programme confiance un peu avant le début de ma thèse, je regardais une présentation qui avait lieu durant un des tous premiers événements de ce programme. L'orateur était en train de présenter les différentes thèses financées par ce programme. Que vois-je ? Un individu, nommé Adrien Le Coz, partagera le même directeur de thèse que moi. J'étais loin d'imaginer que cette personne deviendrait plus qu'un collègue de bureau et qu'on allait réaliser les 400 coups de la thèse ensemble. Merci à toi pour tous ces moments passés ensemble : notre première conférence à Vienne, les nombreuses discussions au tableau sur nos sujets de thèse respectifs, ton partage de la connaissance fine de l'actualité du monde de l'IA, nos fous rires en lisant les papiers absurdes attaque LLM. Étant pas très à l'aise avec l'administratif, j'ai eu la chance de bénéficier de ton sens de l'organisation. Je suis très heureux d'avoir partagé pendant trois années ces bureaux avec toi. Et bien sûr, le meilleur, le plus important pour la fin, merci pour ces nombreuses séances tractions, dips, pompes, squats, ... et surtout surtout récemment MUSCLE UP !!! Validation d'un muscle up obligatoire pour obtenir sa thèse. Goggins serait fier de nous. They don't know us son.

Un merci aux équipes de l'IRT SystemX. Merci à l'IRT et au programme Confiance.AI de nous avoir offert un cadre de thèse idéal entre académique et industriel. Merci au service comptabilité pour votre réactivité dans mon retard à faire mes réservations pour les conférences. Merci à Patrice Aknin de m'avoir donné l'opportunité de présenter mes travaux au comité scientifique de l'IRT. Un merci particulier à Fabien qui m'a fait découvrir le JJB. J'espère avoir le temps pour pratiquer davantage à l'avenir.

Votre train en provenance de Massy TGV va entrer en gare de Bordeaux Saint Jean. (Petit merci à la SCNF au passage). Place maintenant aux remerciements concernant mes proches de Nouvelle-Aquitaine.

Tout d'abord merci au Lucky-Luke de l'enfilage de chaussettes, l'homme qui mettait ses chaussettes plus vite que son ombre, j'ai nommé Dorian. Merci pour nos discussions MMA et nos partages de vidéo de concours de baffes absurdes. Merci à Marie-Mathilde Sancho-Lopez-Diaz-De-Oliveira-Garcia pour ce bon goût de la musique et du partage. Merci à vous deux depuis le master, on a passé des superbes moments ensemble et ça durera.

Place aux écolos bobo bordelais toujours en retard. Merci à Camille pour les nombreuses discussions intéressantes sur les sciences sociales et la politique. Nos sorties à 4 sont "fun" même si vous nous empêchez toujours de parler de maths. Merci au demi-pensionnaire du 7 Rue Achille Allard, Gauthier, de venir si régulièrement pour goûter mes plats et boire des cafés. Ça été un plaisir de discuter de nos sujets de thèse et plus généralement de discuter de maths avec toi. On n'a pas fini de gribouiller des feuilles en

croyant résoudre un super problème et pour ça merci mon ami. Merci également pour nos concours de blague la plus nulle où chacun de nous ne sait pas dire stop et finit par déraper. Pour la suite, je te transmets mon flambeau TGV MAX. Bon courage pour les allers-retours.

Merci aux deux autres mousquetaires du master, Benjamin et Mehdy, pour les fous rires sur le groupe. Vous me manquez.

Petit détour par le bassin d'Arcachon, pays de l'huître. J'ai la chance d'avoir une belle-famille d'ostréiculteurs. Une chance pour changer d'air et profiter de cet environnement magnifique. Merci à Catherine et Olivier de m'avoir si souvent accueilli, nourri (je digère encore), gâté. C'est toujours un plaisir de partager un moment avec vous. Venant des terres, j'ai toujours été méfiant à l'idée de manger une huître, merci de m'avoir prouvé le contraire. Un bisou à Mimi le chat de Sibérie qui n'a jamais vu la neige. Un merci également à Angèle, dit le rat, pour son zèle extrême qui m'a souvent fait rire. Un merci également aux mamies et Bernard pour les repas et les balades en bateau.

Retour aux sources, le Périgord noir. Merci beaucoup mamza de m'avoir concocté des délicieux plats qui me rappellent mon enfance, notamment l'anguile, un délice, les gâteaux aux noix, une tuerie. Merci pour toutes les attentions que tu as pu avoir en général, et merci de m'aider dans la réalisation de ces plats. Merci également pour ton soutien. Bisou à Missou le pirate. Merci beaucoup papa et Sylvie pour les week-ends ressourcements en Périgord. Merci de m'avoir laissé fendre du bois pour me changer les idées. Merci également pour les délicieux plats, toujours ce que je préfère pour me faire plaisir. Merci de votre soutien. Je tiens également à remercier mes grands frères, Pascal, Jean et Louis pour leurs mots motivants.

Et pour conclure ces remerciements je tiens à remercier la personne qui a chamboulé ma vie juste avant la thèse et qui a fait que ma personne et ma vie ne seront plus les mêmes. Mariette, mon amour, merci d'être toi-même, merci de m'avoir tant appris durant ces trois années (j'apprends encore sur l'organisation, du moins j'essaye). Merci de m'avoir autant soutenu pendant les moments intenses d'envoi de papier et surtout pendant la rédaction du manuscrit, comme par exemple, les deux dernières semaines durant lesquelles j'ai été chouchouté. On blague souvent en disant que je suis le pilier sur lequel tu peux t'appuyer, la vérité est que j'ai de la chance, moi, de t'avoir dans ma vie. Merci de m'avoir fait rire, merci d'avoir été rougnousse, merci d'être autant attentionnée avec moi, merci d'avoir été là pour me remonter le morale, merci de m'avoir rassuré, merci d'avoir été la pour me conseiller, merci de m'avoir botté les fesses quand je faisais n'importe quoi (n'en abuse pas non plus), merci pour ton franc parlé, merci d'avoir supporté que je travaille jusqu'à pas d'heures ou que j'emmène mon pc partout pour potentiellement travailler 5 minutes, et finalement merci pour ta sensibilité. Grâce à toi, ma vie en dehors de la thèse a pu être des plus épanoui et cela a eu un impact que l'on sous estimera toujours sur la réalisation de mes travaux. Pour tout ça et pour toutes les choses que j'ai dû sûrement oublier, merci Mariette.

Contents

List of Figures

List of Tables

Chapter 1

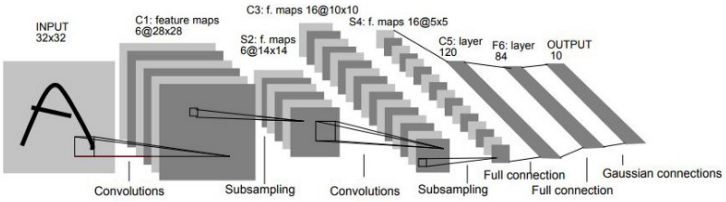
Introduction

The recent years have seen the emergence of *Artificial Intelligence* (AI) systems in several domains. In healthcare, AI enhances diagnostics, personalized treatments (Topol, 2019), and drug discovery (Jumper et al., 2021). AI helps in preventing malicious behaviors by detecting frauds in finance or malware. Natural Language Processing (NLP) recently powers the use of Large Language Models (LLM) (Vaswani et al., 2017), allowing virtual assistants, language translation, and text classification. AI succeeds in playing different games (StarCraft (Vinyals et al., 2019), Stratego (Perolat et al., 2022)) requiring different skills (Vinyals et al., 2019; Perolat et al., 2022). The most famous example is AlphaGO (Silver et al., 2017), which defeated Lee Sedol, a Go world champion, through multiple matches. AI systems have also improved computer vision systems, such as object detection in images (Ren et al., 2015) and image synthesis (Goodfellow et al., 2014a).

As the integration of AI systems expands across various sectors, there is a growing awareness of the necessity for these systems to be trustworthy. A breach of trust among stakeholders can have significant social consequences. For example, bias in AI algorithms can lead to discriminatory outcomes, particularly in sensitive areas such as hiring and law enforcement, raising ethical concerns about equity and justice. Furthermore, the opacity of AI decision-making processes complicates the ability to audit and interpret outcomes, which can undermine user confidence. AI systems exhibit vulnerabilities that malicious entities can exploit to cause harmful AI behavior. These challenges obliged academics and industries to develop new solutions to ensure that AI systems operate reliably. This dissertation focuses on one aspect of trustworthy AI: safety for computer vision systems against malicious physically feasible attacks.

1.1 The advent of the Deep Learning era

The several AI advances describe before have been made possible by the development of *Machine Learning* (ML), a branch of AI consisting of developing algorithms that learn and make decisions from data. Unlike traditional algorithms, which rely on explicit user instructions, ML models process a collection of data to identify patterns, relationships, and correlations within data to make predictions or decisions. The first generation of ML models relies on manual feature engineering. It consists of the selection and the creation of new input variables from raw data. Usually done by an expert, the goal of this feature engineering is to enhance the learning performance of the ML system. This handcrafted engineering has been revolutionized by the advent of a new generation of ML models. *Deep Learning* (DL) leverages neural networks with multiple layers to automatically learn



(a) LeNet5 architecture. Source: LeCun et al. (1998).

(b) Screenshot of the demonstration of LeNet1. Source: Demo.

Figure 1.1: Overview of the application of LeNet to recognize hand-written digits

intricate hierarchies of features directly from raw data, surpassing the limitations and the burden of manual feature engineering.

1.1.1 A slow start

The rise and success of artificial neural networks has been a long and sinuous journey. The term neural network is inspired by the biological structure of the human brain (McCulloch and Pitts, 1943). The reader can refer to Appendix 1.1b or the introduction provided in a book authored by LeCun et al. (2015) for a short or a more complete history about neural networks, respectively. Neural networks were first introduced in the early 1940s, but interest in the field did not gain momentum until the 1980s. Renewed interest emerged with the development of backpropagation (Rumelhart et al., 1986), a method for efficiently training multi-layer neural networks. This method uses the chain rule to compute the gradient of the loss function (function that quantifies the error margin between a model's prediction and the actual target value) with respect to the weights by expressing the derivative of the composition of two functions in terms of each of their derivatives. The gradient is then used in the stochastic gradient descent algorithm (Robbins and Monro, 1951) to iteratively update the network's weights. This period also saw the introduction of important architectures like the recurrent neural network (RNN) (Hochreiter and Schmidhuber, 1997) and the convolutional neural network (CNN) conceptualized by Fukushima (1980). These architectures became crucial for tasks involving sequential data and images, respectively. A famous example of a CNN used to recognize hand-written digits on paychecks was the work carried out by LeCun et al. (1998) (see Fig. 1.1).

1.1.2 A tremendous success

The success story of Deep Learning began in 2012 with the CNN AlexNet's triumph in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) (Krizhevsky, 2012). The ImageNet competition is a 1000-class image classification task utilizing a dataset comprising over one million annotated images. AlexNet was the first end-to-end trained artificial neural network to beat by a large margin (reducing $\approx$ 10% of the top-5 error) previously introduced image classifiers based on bag-of-words (Perronnin et al., 2010) (see Fig 1.2). The AlexNet's success was essentially due to the access to the biggest computational power, allowing it to scale up and train the network. The success was also due to the use of already developed techniques such as the use of rectified unit as non-linear activation (Nair and Hinton, 2010), the use of dropout (Srivastava et al., 2014) and the pooling operation (Jarrett et al., 2009).

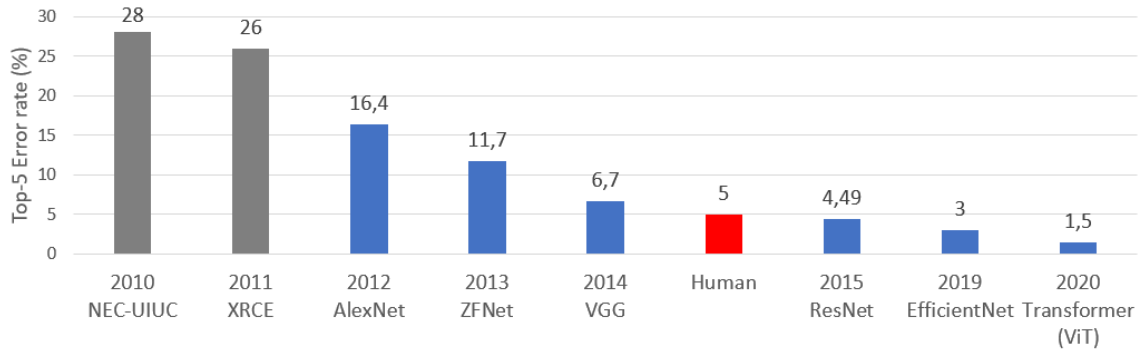


Figure 1.2: Evolution of the top-5 error on ImageNet competition through the years. The top-5 accuracy measures if the true label has been predicted within the five more likely labels.

Since AlexNet, the field of Deep Learning has exploded, and all subsequent winners of the ImageNet competition have been deep neural networks. Innovations lead to deeper and more refined models like VGG (Simonyan and Zisserman, 2014), which emphasized depth and simplicity with its uniform convolutional layers, or e.g., ResNet (He et al., 2016), which introduced residual connections to train extremely deep networks without training degradation. As the pursuit of more effective models continued, the search for efficient networks has become an optimization problem itself, leading to balanced speed/accuracy architecture like EfficientNet (Tan and Le, 2019). Over the years, Deep Learning models have improved and have exceeded human performances in ILSVRC. Today, the success of Deep Learning goes beyond the image classification task, and Deep Learning models have become state-of-the-art for various applications. These achievements have been made possible by scientific accomplishments but also by combining two technological advances; large-scale data acquisitions through the installation of sensors, often called *big data* (initially annotated data, but now not necessarily), and an ever-increasing computation power. Most papers have then been dedicated to improving the performance of Deep Learning models on a particular task. We can observe the progress of the different benchmarks on the Papers With Code platform[1]. However, the widespread adoption of DL in industry has introduced new challenges that must be addressed to make it acceptable as a reliable solution. At the same time, as deep networks grew in size, their learning and decision-making processes became less and less transparent. These deep networks are complex and opaque objects – they are generally referred to as *black-box* systems – and have shown instability. A famous example of network instability is the adversarial example perturbation, where a small bounded perturbation of the input can sway the network to output a chosen target (see Fig. 1.3). The industrial challenges, the poor understanding of DL mechanisms, and its security flaws have raised several critical questions that must be addressed to build an AI system that aims to be deployed. These questions can be gathered around the notion of trustworthy AI.

1.2 Trustworthy AI

AI systems must ensure some properties to be correctly designed, developed, and deployed. Companies or regulators define these properties and form the concept of *Trust-*

[1] https://paperswithcode.com

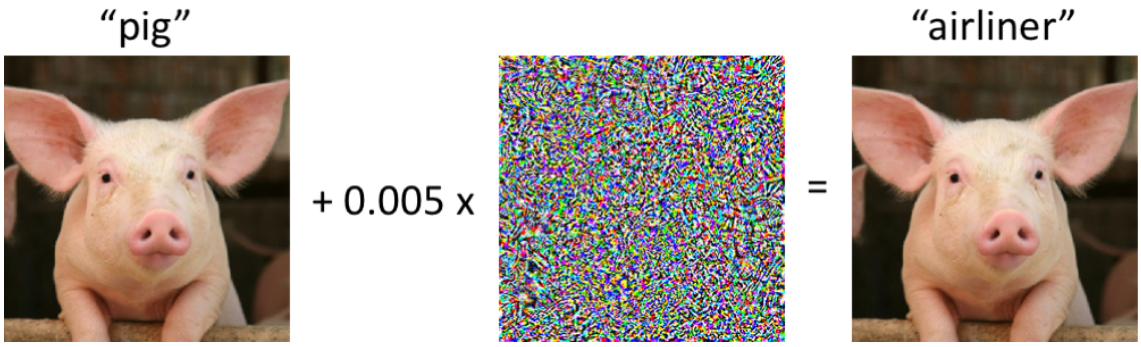


Figure 1.3: Adversarial example targeting an image classification neural network. The original image on the left is correctly classified as a pig. However, after introducing an adversarial perturbation, the resulting image, indistinguishable from the original, is misclassified as an airliner. The adversarial perturbation, illustrated in the center, results from an optimization problem that influences the network's behavior. Source: Szegedy et al. (2013).

worthy AI. Depending on the application, a trustworthy AI must verify properties such as robustness, alignment with ethical considerations, explainability, transparency, or safety.

Robustness. An AI component is considered robust if it demonstrates a consistent level of performance in the target environment (environment in which the system is deployed). This environment can be complex and diverse, and if an AI model is not trained to account for the variation of data distributions across scenarios, its performance may be significantly dropped. Let us consider the example of an object detector deployed in an autonomous vehicle that detects pedestrians. This detector should maintain the same level of performance during the night or the day, if it is raining, snowing, or sunny or if the car is driven in the city or the countryside. This phenomenon is often referenced in the literature as robustness against distribution shift (Taori et al., 2020) and is related to the notion of generalization. Generalization represents the ability of a model to make accurate predictions on unseen data drawn from the same distribution as the training data. In our example, an object detector trained on the data from a particular city must perform well on frames from different parts of this city with the same weather conditions and daytime. A central objective is to design methods to augment generalization and robustness against distribution shifts without re-collection and re-annotation of a large volume of data.

To correctly qualify if a model is “well” operating on a given data distribution, the network's output must align with our expectations and interpretations of what it should represent. For example, for an image classification neural network, the output vector, after the final softmax layer, is generally interpreted as the probability distribution over classes for a given input. However, this interpretation has been shown to be incorrect (Guo et al., 2017). Neural networks trained on ImageNet are often overconfident, meaning that images classified with a probability of 90% will not be correctly classified 90% of the time. To address this problem, a line of works developed the notion of uncertainty quantification for neural networks (Gawlikowski et al., 2023).

Ethics. Ethical considerations are principles explicitly defined by regulators or norms implicitly induced in society. These principles evolve with new regulations or changes in people's thinking. In Western countries, an algorithm is aligned with ethical considera-

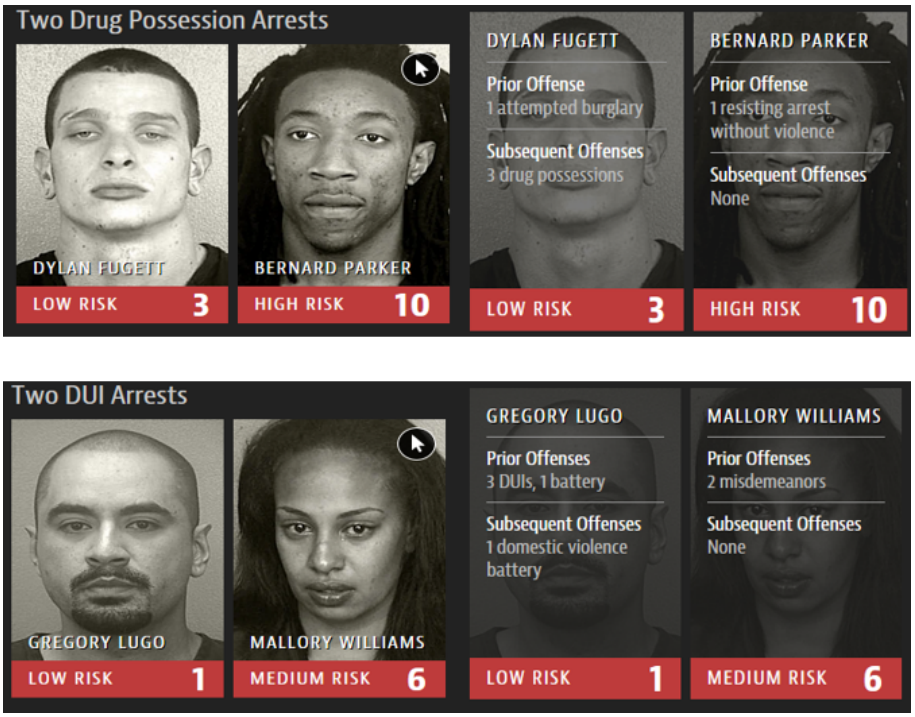


Figure 1.4: Two examples of COMPAS ethnic bias to predict recidivism. For the same criminal offense, COMPAS incorrectly predicts black people as re-offenders, while white people are incorrectly predicted as low risk to re-offend. Pictures from Probublica.

tions if it is diverse, non-discriminative, and fair, meaning that the AI system is developed and used in a way that includes diverse actors and promotes equal access, gender equality, and cultural diversity while avoiding discriminatory impacts and unfair biases. Without specific training considerations, an AI algorithm can be biased and can do so without access to sensitive information. For example, in the U.S., a criminal risk assessment tool named Correctional Offender Management Profiling for Alternative Sanctions (COMPAS)[2] has been used to assess more than 1 million offenders since 2000. This AI software predicts up to ten scores of the defendant's risk of committing a misdemeanor or felony within two years of assessment from 137 features about an individual and her/his past criminal record. The features do not include the defendant's race. In 2016, an independent, non-profit newsroom, ProPublica, published an analysis of COMPAS's efficacy on more than 7000 people (Angwin et al., 2016). This analysis indicated that the predictions were unreliable and racially biased. Even if the overall accuracy for white defendants was slightly higher than its accuracy for black defendants (67.0% vs 63.8%), the errors made by COMPAS were very different for white and black people. Black defendants who did not recidivate were incorrectly predicted to re-offend at a rate of 44.9%, nearly twice their white counterparts at 23.5%. White defendants who did recidivate were also incorrectly predicted to not re-offend at a rate of 47.7%, nearly twice their black counterparts at 28.0%. Two examples are depicted in Figure 1.4. COMPAS appeared to favor white defendants over black defendants by underpredicting recidivism for white and overpredicting recidivism for black defendants. This example shows the importance of sound and careful data collection and AI system training strategy to avoid biases in harmful AI applications.

[2]https://doc.wi.gov/Pages/AboutDOC/COMPAS.aspx

Explainability and Transparency. AI systems are either explainable by design (e.g., linear regressions, decision trees, the KNN method, ...) or either black-box systems (Gunning et al., 2019; Hanif et al., 2021). Generally, the least accurate methods are the most explainable, and the best-performing methods are the least explainable. Deep Learning models fall in the last category, exhibiting high performance but less explainability. These models are very hard to understand due to the mathematical challenges raised by the interplay between the model's architecture, the optimization algorithm used to learn the model, and the training data. For these multiple reasons, they are often referenced as a black-box system. This poor understanding leads academia and the industry to need more explainable models. Explainable AI (XAI) aims to make AI models behavior more understandable and intelligible to humans. Often, XAI can be gathered around the question, "Why do these AI models make such a decision rather than another?". The answer is often incomplete and depends on the task, considered XAI methods, and user expectations. From an application-oriented point of view, a better understanding of the system will help to prevent its potential failures. For example, for an object detection system, understanding if the detection comes from the object itself or the context (background) may help to identify spurious correlations (e.g., the presence of a dog implies the presence of a person). From the customer's perspective, explainability may help to know why the model rejects loan applicants and what they can do to be qualified. These examples illustrate that explainability can be a tool to qualify the model's robustness and its potential biases. Although the definition of XAI is still an open question, it is a keystone between AI, social sciences, and human-AI interactions (Miller, 2019).

Transparency means that users know enough about the intrinsic mechanism of an AI system to understand the whole AI pipeline, how it works and uses data. Misunderstanding the AI system behavior may lead to harmful consequences, depending on the user. If the user is an operator, such as a doctor, misunderstanding the results of an AI scan diagnosis can have severe consequences for the patient. Whereas, if the user is a customer using a biometric system for identification, he/she will be concerned about the purpose for which his/her biometric data is collected and how it is used. This transparency allows users and stakeholders to understand how decisions are made, fostering trust and accountability. For governments and regulators, transparency is a safeguard against companies that may not respect ethical considerations or privacy.

Safety. AI safety aims to ensure the integrity and security of AI systems against carefully crafted and deceptive threats. These threats are vulnerabilities in AI systems that can cause them to make mistakes (Szegedy et al., 2013; Biggio et al., 2012; Brown et al., 2017). Such vulnerabilities should be separated from the robustness issues discussed earlier. For example, adversarial examples are small perturbations designed to fool a model. They are not related to a physical phenomenon that can be observed in the intrinsic distribution of natural data – mathematically, the probability of observing such phenomenon in natural data is null. Safety vulnerabilities can be divided into either training-time attacks or decision-time attacks.

Training-time attacks target model learning through data or model manipulation. In a *data poisoning* attack (Biggio et al., 2012; Gu et al., 2019), the hacker controls a subset of the training data by inserting or modifying training samples to cause the general failure of the network once trained. In a backdoor attack (Li et al., 2022), the method is almost the same, but the goal is to manipulate the network at the moment the hacker inserts a trigger into the data. In the context of federated learning, training-time attacks can also target the

learning of a model's client (in a smartphone, for example) to disrupt the learning of the global model (in a server) (Liu et al., 2022b).

Decision-time attacks are attacks that occur at the time of model inference rather than during the training phase. These attacks can cause security issues by inducing network failures or cause privacy issues by stealing the model or the training data. Neural network failures can be produced by adversarial examples (Szegedy et al., 2013), a norm-bounded perturbation of the input data. For computer vision applications, adversarial examples are perturbations applied to an image that are indistinguishable by the human eyes. This perturbation can cause an image classifier to classify an image of a class to another with high confidence (Fig. 1.3). An example of an algorithm-level threat is the model stealing attack, where the attacker attempts to learn a model that mimics an unknown model accessible through an API (Tramèr et al., 2016). The copy of the model can bypass the API's monetization or be used as a proxy to design a stronger adversarial example against the API's model. For the public to accept AI systems, ensuring data privacy is essential. This data privacy can be jeopardized by membership inference (Shokri et al., 2017), an attack querying a trained model to predict whether or not a particular data was contained in the model's training dataset, or by data reconstruction attacks (Loo et al., 2023), an attack aiming to reconstruct the data that was used when training the model.

Defense mechanisms are used to maintain an AI system's safety. These defenses can be either empirical (Madry et al., 2017), aiming to show that available attacks empirically fail to bypass them, or *certified*, meaning that the system is mathematically robust under certain conditions. Empirical defenses often offer higher robustness performance but are not guaranteed. To ensure the system's robustness, certified defenses are usually based on formal methods (Zhang et al., 2018).

Towards a trustworthy AI. Many initiatives have been launched in Europe to move towards trustworthy AI. For example:

- TAILOR is a European project to provide the scientific foundations for Trustworthy AI in Europe. TAILOR develops a network of excellence research centers, leveraging and combining learning, optimization, and reasoning with the key concepts of trustworthy AI. These systems are meant to provide descriptive, predictive, and prescriptive systems integrating data-driven and knowledge-based approaches;

- CERTAIN (Center for European Research in Trusted AI) is a consortium of German institutes legally part of DFKI (German Research Center for Artificial Intelligence). The CERTAIN consortium aims to work across the value chain from basic research to society, focusing on developing, optimizing, and implementing Trusted AI techniques to provide guarantees and certifications for AI systems in specific use cases. In addition to research, the consortium collaborates with industry, standardization bodies, and political and societal stakeholders to set certification requirements, define AI trust labels, and foster AI literacy;

- The High-level Expert Group on AI, a group of experts appointed by the European Commission, presented their final Assessment List for trustworthy AI to the European Commission. This assessment list gives an initial approach for the evaluation of trustworthy AI to leverage the self-evaluation of developers and deployers of AI. The document[3] can be found here.

[3]https://digital-strategy.ec.europa.eu/en/library/assessment-list-trustworthy-artificial-intelligence-altai-

Figure 1.5: An overview of the different actors involved in the Confiance.ai program

The Confiance.ai program In late 2021, the Confiance.ai program was launched in France. It is a joint initiative of industry and academia to develop and promote French capabilities in trustworthy AI. This multidisciplinary environment leverages diverse expertise to address the different challenges (robustness, ethics, explainability and transparency, safety) raised to obtain a Trustworthy AI (see Fig. 1.5). The different challenges were divided into projects (EC for "environment de confiance" or trustworthiness environment), divided into sub-problems to tackle each problem independently. To learn more about the Confiance.ai program, see the Appendix A.2 and the White Book[4].

To develop scientific solutions, academics put forth several proposals, leading to the initiation of doctoral studies and postdoctoral research. PhD candidates were associated with a research organization (typically IRT SystemX) and an academic partner. Each candidate has one advisor from the IRT and one or more advisors in the laboratory. Each PhD topic was associated with an EC. This thesis is associated with EC4 (related to the development of trustworthy AI by design) and was started initially to leverage the safety aspects of already deployed AI systems, especially AI components deployed in cyber-physical systems.

1.3 AI Safety

As previously stated, AI safety aims to guarantee the integrity and security of AI systems against deliberate attacks. This document will concentrate on attacks on computer vision AI systems that have been designed, trained, and deployed safely without any malicious backdoor. To circumvent the decision-making process of this system, a hacker has to

self-assessment

[4]https://www.confiance.ai/contenus-media/

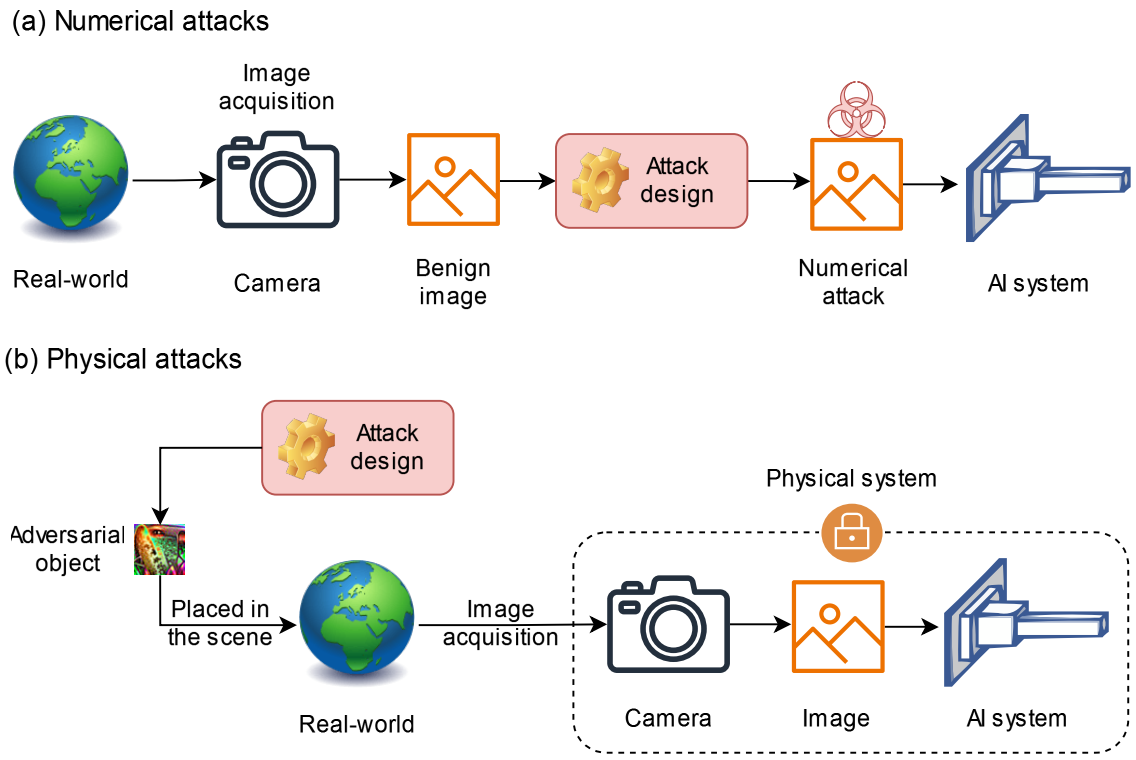


Figure 1.6: Illustration of the two methods to compromise an already deployed computer vision AI system. (a) Numerical Attacks involve modifying the recorded image at the pixel level during the processing pipeline to mislead the AI system. (b) Physical Attacks involve introducing deceptive physical objects into the real-world scene, tricking the AI system into making erroneous predictions during image processing.

corrupt the image being processed by the system. Two methods exist to achieve this: the hacker can numerically alter the processed image by injecting an attack, or he/she can insert an object into the scene that will subsequently deceive the AI system that processes images (see Fig. 1.6). These two attacks have different operational scenarios and are suited to target different computer vision systems successfully. For example, numerical attacks are more effective when targeting numerical applications, such as a social media sensitive-content filter, than a system embedded in an autonomous vehicle. To target the latter system, a hacker using numerical attacks would require access to the system to inject the attack after the image is captured, which is an unrealistic scenario. Conversely, physical attacks involve introducing a disruptive element (e.g., a new object) into the scene, making them a more suitable approach for targeting real-world systems than numerical attacks.

Physical attacks have been introduced by Brown et al. (2017). Rather than finding a small additive adversarial noise, they constrain the optimization procedure to a small part of the image while also allowing the optimization to be unconstrained in magnitude. At the end of the optimization, a small textured patch will be produced that may cause the image classifier to predict the wrong category, e.g., a *toaster* when printed and placed in a scene. This new type of attack, named *adversarial patch attack* (APA), enables a variety of physical-system threats: adversarial stickers that fool an image classifier to predict a selected class (Fig. 1.7 (a)), adversarial t-shirts (Hu et al., 2022) hiding the person that wears it (Fig. 1.7 (b)), adversarial posters suppressing the detection of innocent people

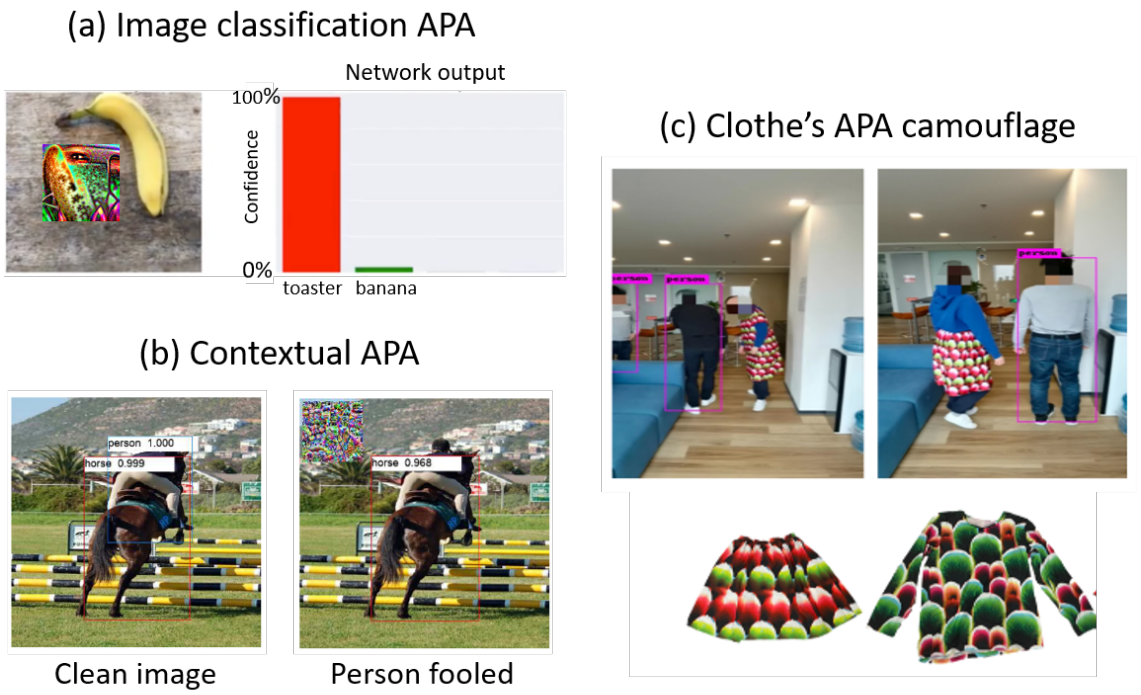


Figure 1.7: Examples of adversarial patch attacks targeting different real-world systems. In the top-left image (a), the patch is designed to deceive the image classifier, causing it to output the desired class, *toaster*, with high probability. In the bottom-left image (b), the patch, when placed at the top-left, effectively suppresses the detection of the person despite not overlapping the person's body. In the image on the right (c), the t-shirt or the skirt successfully conceals the detection of the person wearing it. Sources: Image classification APA inspired by Brown et al. (2017), Contextual APA from Saha et al. (2020) and Clothe's APA camouflage from Hu et al. (2022).

walking around (Fig. 1.7 (c)) (Saha et al., 2020), road marks altering depth prediction in a road scene (Yamanaka et al., 2020). While APAs can be utilized as a numerical attack, their primary value lies in their physicality through printing. Consequently, within the existing literature, numerical attacks are typically associated with adversarial noises (often called adversarial examples in the literature), whereas physical attacks refer to adversarial patch attacks.

A substantial body of literature has been devoted to the design of defense mechanisms to counter adversarial attacks. To defend against adversarial noises, a vast corpus of empirical and certified defenses has been developed. Currently, adversarial training (Madry et al., 2017), an empirical defense mechanism that includes attacks during the network training to make it resilient, represents a state-of-the-art defense and offers robust performance. Nevertheless, this approach is only partially applicable for making a network robust against APA due to the higher cost of APA training. Alternative forms of defense have been proposed, including masking-based defense (Naseer et al., 2019b; Liu et al., 2022a; Xu et al., 2023; Tarchoun et al., 2023) or network feature space detection defense (Yu et al., 2021). The race between attackers and defenders can be likened to a game of cat and mouse. However, defenders against APAs have yet to reach the same performance level as adversarial training in defending against adversarial noises. The cat-and-mouse

game between attackers and defenders is advantageous for defenders concerning adversarial noises, but it is not yet clear for APAs.

1.4 Thesis goal and contributions

Definition of a critical APA. The lack of a cornerstone adversarial patch defense and the increase in APA's attack performance illustrates that APA may be critical for physical-based systems like autonomous vehicles. However, no work in literature describes the prerequisites of a critical APA. A contribution of this manuscript is to propose a definition of what may be a critical adversarial patch. To effectively target physical systems, APAs must be robust to physical transformations and transferable to multiple neural networks. The physicality of an APA represents its ability to be resilient to a wide range of physical transformations that may appear during a camera recording (change in the object-camera angle, pose, distance, scene lighting, blurring, distance patch target object, multi-scene, ...). A low patch physicality will result in an attack performance drop if a scene setting changes. Patch transferability refers to the ability of the patch to fool different models from different architectures or the same architecture but with a different initialization or training procedure. The transferability measures the attack performance when the attack designed on model A targets a different model B. It is an essential prerequisite for a critical APA, as the malicious person may not know the model deployed in the targeted physical system. Generally, model A, named the source model, performs the same task as the unknown targeted model. For example, the malicious person wanting to fool an image classifier may design its patch on a source model that is an image classifier.

Although APA's physicality has been developed and improved through multiple works ((Hu et al., 2022) design a patch invariant to patch camera angle), the transferability of APA remains a challenge. The adversarial patches proposed in the different papers generally show a good attacking performance in white-box configuration (applied on the same known model that they have been learned) but cannot transfer.

Towards a transferable APA against image classification. To better understand what might be causing the transferability of the patch, we first study the case of transferability between image classifiers. This situation should be easier to understand as all image classifiers can be decomposed in two steps: an encoder extracting the features of images followed by a linear head aggregating these different features to perform the classification. Object detectors are much more complex deep networks as they do not exhibit a common structure between the different architectures and are built as multiple interacting blocks.

Recent works seem to show that the transferability of patches between image classifiers seems to be realizable: Brown et al. (2017) and Doan et al. (2022) show that a patch designed on a specific ResNet (He et al., 2016) was capable of fooling another ResNet or a DenseNet (Huang et al., 2017). However, it is worth noting that these works measure patch transferability on old versions of CNNs trained with a poorly generalizable learning policy compared to recent ones. This thesis contributes to extending the transferability study on recent architectures like Transformers or CNNs trained with recently developed learning policies.

To leverage large-scale transferability between old and new classifier architectures, we propose a new methodology for APA design. This methodology avoids the reliance of APA design on source model decision boundaries and addresses the limitations of

transfer-based attacks by introducing a distribution-oriented approach. Our patch is learned using the Wasserstein distance (Peyré et al., 2019) to globally alter the feature distribution of a set of images to match another distribution taken from a target class. When placed in the scene, our patch causes the network to output the class chosen as the target distribution. We validate our method on ImageNet-1K (Deng et al., 2009) by conducting extensive experiments. We show that our APA is more model transferable and is more physically feasible than previous APAs for a large ensemble of network architectures, including classical CNNs (Simonyan and Zisserman, 2014; Szegedy et al., 2016; He et al., 2016; Huang et al., 2017), recent CNNs (Tan and Le, 2019; Liu et al., 2022c) and Vision Transformers (Touvron et al., 2021; Liu et al., 2021).

A transferable inhibition APA to hide objects. A subsequent goal is to create a transferable patch against object detection, a more complex task than image classification. Object detection is more likely to be used in physical systems, as it allows the detection, localization, and classification of multiple objects in a single image. If we naively apply the patch obtained by our distribution-based method on an object detector, it creates false alarms on the patch. However, creating false alarms on the patch is much less critical for the system than suppressing good detections. It is much more critical for an attacker to either try to hide someone by designing a t-shirt or perturb the detection of innocent pedestrians crossing the road by positioning a poster near a crossroad. The goal of a patch is entirely different. Rather than trying to create a patch that catches the decision and is classified as an object, the goal is now to design a patch that inhibits the detection of a particular class.

To ensure that the produced patch is suppressing the object we want to attack, we propose changing the targeted task on which the patch is learned and evaluated, and designing the patch to target a multi-label classification network instead. A multi-label classification network outputs a binary response for each class, representing whether or not the class is present in the image. The patch performance is then measured by the overall reduction of true positive rates for different class probability thresholds. We show that in the white-box setting, it is possible to create a patch that disturbs the detection of a person. However, we conducted experiments that indicated that creating a transferable patch, even when the architecture is fixed and networks are learned using different learning recipes, is challenging.

To summarize, in this thesis, we try to analyze more precisely the criticality of APAs by:

1. defining various categories of evaluation criteria. We propose an evaluation methodology based on these criteria to evaluate the patch's criticality. We use this methodology to evaluate the different patches from the literature and show that patches do not transfer across network architectures or network initialization/training recipes, limiting patches' criticality (Chapter 3);

2. introducing a new framework based on optimal transport for creating patch attacks that are highly transferable to unknown networks. This framework is based on the idea of attacking feature distributions, which is independent of the classifier decision boundaries and has several optimization benefits over previous feature-based methods. We show that our attack works for a large spectrum of deep networks, including Convolutional Neural Networks, Transformers, and adversarially trained models, and show transferability superiority through extensive experiments on the

ImageNet-1K dataset. We illustrate through physical experiments that our patch is potentially harmful to real-world image classifiers even in the presence of a defense mechanism (Chapter 4);

3. addressing previous APAs against object detector limitations by proposing a scientific objective and evaluation procedure to assess the efficacy of attacks against object detectors. This new scientific objective is to create a patch against a multi-label classification network and measure its impact on inhibiting true positive detections. We develop a new APA that targets multi-label classifiers and demonstrate that this APA can suppress the detection of a person in the white-box setting. We illustrate that designing a transferable APA against the multi-label classification task is challenging (Chapter 5).

1.5 Outline of the manuscript

Chapter 2 introduces the tools to better understand the adversarial attacks research field. This chapter provides general materials, while the following contribution chapters delve into specific materials, offering detailed insights into particular aspects of the topic. In this chapter, we first present the main architectures used in image classification and object detection. We explain how these models are trained and what may influence their learning. Then, we introduce the two attacks that aim to target an already deployed network. We describe how to generate them and which properties are used to characterize them. Finally, we describe the pioneer works that developed adversarial patch attacks, which is the most physically realizable type of attack.

We define in Chapter 3 what critical patches are and introduce different categories of evaluation criteria useful to evaluate APA's criticality. We use these criteria to evaluate the different patches and conclude that actual patches do not transfer across models.

Chapter 4 is dedicated to the design of a transferable patch against image classifiers. We study previous patch attack limitations and introduce a new method that resolves these limitations. We numerically evaluate our new APA and show its superiority over previous methods for a large number of state-of-the-art networks. We illustrate through physical experiments that our patch is potentially harmful to real-world image classifiers, even in the presence of a defense mechanism.

Finally, in Chapter 5, we introduce a new scientific objective and evaluation procedure to evaluate the APA's efficacy against object detectors. We develop a new APA against this new multi-label surrogate problem and demonstrate its attacking performance in the white-box setting. We then illustrate the limitations in designing a transferable APA against multi-label classifiers and, thus, against object detectors.

1.6 List of Publications

- Labarbarie, P., Chan-Hon-Tong, A., Herbin, S., & Leyli-Abadi, M. (2022, July). Benchmarking and deeper analysis of adversarial patch attack on object detectors. In Workshop Artificial Intelligence Safety-AI Safety (IJCAI-ECAI conference). (Oral)

- Labarbarie, P., Chan-Hon-Tong, A., Herbin, S., & Leyli-Abadi, M. (2022). Carpet-bombing patch: attacking a deep network without usual requirements. arXiv preprint

arXiv:2212.05827.

- Labarbarie, P., Chan-Hon-Tong, A., Herbin, S., & Leyli-Abadi, M. (2024, May). Optimal transport based adversarial patch to leverage large scale attack transferability. In The International Conference on Learning Representations (ICLR 2024). (Poster)

Chapter 2

Adversarial attacks toolkit

In this chapter, we introduce the basic notions useful for the rest of the manuscript. The chapter content is global, while in the following chapters, we provide chapter-specific material to present our contributions.

As our objective is to design attacks to fool neural networks, the first part of this chapter is dedicated to their description. The triptych architecture, training data, and learning strategy characterize neural networks. Architecture forms the core of their design. We begin by describing how these architectures have evolved over the years, and we introduce several architectures used in image classification and object detection. We then present the different learning strategies that influence the obtained network weights, resulting in different network behavior and performance even for networks sharing the same architecture.

The second part of this chapter gives an overview of two adversarial attacks designed to target a well-trained network. We describe their generation process and characteristics. Finally, we detail the pioneer works that developed adversarial patch attacks, the most physically realizable attack of the two. This attack is the main topic of the thesis and is further studied in the following chapters.

2.1 Computer Vision

Computer vision is a field of computer science that aims to enable computers to solve tasks by replicating human visual abilities. These tasks may involve pattern recognition, motion analysis, and image restoration. This manuscript will focus on pattern recognition tasks, particularly image classification and object detection. To perform such tasks, computer vision seeks to answer the questions of extracting relevant descriptors from images, what relevance means, and how to use these descriptors to perform the task. Suppose we want to create an AI system that classifies whether a dog is present in an image. A relevant descriptor for performing this task might be detecting the presence of a muzzle in the image. However, the unique presence of a muzzle in the image is insufficient. This strategy will lead to misclassifying a cat or a pig as a dog. To correctly determine whether a dog is present in the image, more than one descriptor must be extracted and then aggregated to make a decision. The design of a computer vision system may be divided into two goals: how to extract relevant descriptors and how to use these descriptors to perform a task. The first goal is often called *representation learning*, while the second can be called *task learning*. The quality of task learning, and thus the performance of computer vision, often depends on the quality of the extracted descriptors.

The term descriptor refers to representations computed from raw images by an algorithm. It may have different names in the literature, e.g., *visual representation*, or *features*. Before the advent of deep learning, feature extraction was performed using handcrafted methods such as Scale Invariant Feature Transform (SIFT) (Lowe, 2004), Speeded-Up Robust Features (SURF) (Bay et al., 2006), or Histograms of Oriented Gradients (HOG) (Dalal and Triggs, 2005). These methods were developed to capture specific aspects of an image that the designers anticipated would be relevant to solving the task. Extracted features may have desirable properties such as invariance to scale, rotation, translation, illumination, or blur depending on the extraction method. Finding a relevant feature extraction method to perform a certain task can take many years to achieve success and capture only limited aspects of an image, which can limit their effectiveness in diverse or complex scenarios.

Deep learning has revolutionized handcrafted feature engineering by introducing the idea of "end-to-end" representations and task learning (LeCun et al., 2015). Deep learning models, particularly Convolutional Neural Networks (CNNs), can directly learn rich and hierarchical feature representations from raw data. This end-to-end learning paradigm allows deep learning models to automatically discover and optimize features, making them far more adaptable and robust in capturing the nuances of visual information. The shift to deep learning has enabled remarkable improvements in accuracy and efficiency across a wide range of computer vision tasks, redefining state-of-the-art tasks and opening up new possibilities in the field. The first pattern recognition task that shown major improvement was the image classification task.

2.1.1 Image classifiers

In this section, we review the significant achievements made in recent years in designing image classification neural networks. Image classification models aim to predict a single label that describes the image content. To learn features to perform image classification, the first developed and used architecture was convolution based architectures. A Convolutional Neural Network (CNN) is a specialized neural network designed primarily for processing structured grid data, such as images. Leveraging layers that perform convolutions, CNNs automatically and adaptively learn spatial hierarchies of features from input data. Each convolutional layer applies a series of filters to the input, creating feature maps that capture local patterns like edges, textures, and more complex shapes as we progress deeper into the network. The final layers of a CNN usually consist of fully connected layers that perform high-level reasoning and classification tasks based on the features extracted. CNNs can then be represented into a two-part scheme in which the first part, usually named *backbone* or *encoder*, is built with convolutional operations and aims to extract relevant features while the latest part uses these features to perform classification. CNNs have become popular due to their advantageous properties, including sparse interactions with local connectivity, parameter sharing with reduced numbers, and equivariant representation. Equivariant means that the network encodes in a meaningful way image transformations, e.g., rotation or scaling. Hereafter, we introduce some major image classifiers.

AlexNet. The first major success of deep learning for image classification was AlexNet's first place in the ImageNet challenge (Krizhevsky, 2012). AlexNet's success can be attributed to its access of biggest computational power to scale up and train the network and

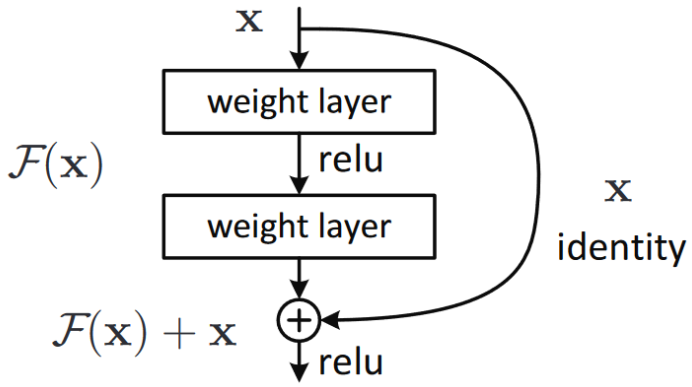


Figure 2.1: Representation of a ResNet residual block. Source: He et al. (2016)

the use of already developed techniques such as the use of rectified linear units (ReLU) for non-linear activation (Nair and Hinton, 2010), the use of dropout to reduce overfitting (Srivastava et al., 2014) and the use of pooling operation to reduce dimensionality and computation, enhancing robustness to spatial variations (Jarrett et al., 2009).

VGG. Building on this momentum, the Visual Geometry Group (VGG) at the University of Oxford introduced the VGG network family, which further explored the impact of network depth on performance (Simonyan and Zisserman, 2014). Although VGG ranked 2nd in the 2014 ILSVRC, it has made a prolonged impact in the deep learning community due to its simple architecture and its "understandable" feature map. The improvement in VGG's accuracy is attributed to the replacement of AlexNet's large kernel-sized filters with 3×3 kernel-sized filters and the increase in the network's depth by adding more layers.

ResNet. The ResNet architecture is one of today's most famous and commonly used architecture (He et al., 2016). This paper introduces residual connection: the input of an earlier convolutional layer is summed to the input of another future convolutional layer several layers later (Fig. 2.1). They show that these residual connections help to stabilize the gradient during training (avoiding gradient vanishing), enabling a deeper network and better accuracy. Notably, they increased the depth of VGG networks by a factor of eight. This breakthrough not only maintained high accuracy but also significantly reduced training time, thus marking a new era in the design and implementation of deep neural networks.

EfficientNet. The growing interest in finding better architectures gave rise to Neural Architecture Search (NAS) (Elsken et al., 2019), a set of techniques aimed at automating the design of neural networks (AutoML). NAS leverages ML or optimization-based methods to explore and identify optimal network configurations, reducing the need for manual experimentation and enabling the discovery of novel architectures that surpass human-designed models. One of the most notable outcomes of NAS is the EfficientNet family (Tan and Le, 2019), which introduced a principled method for scaling networks called compound scaling. EfficientNet simultaneously scales depth, width, and resolution of the network in a balanced manner, achieving superior accuracy with fewer computational resources. This holistic approach demonstrated that well-designed, efficiently scaled mod-

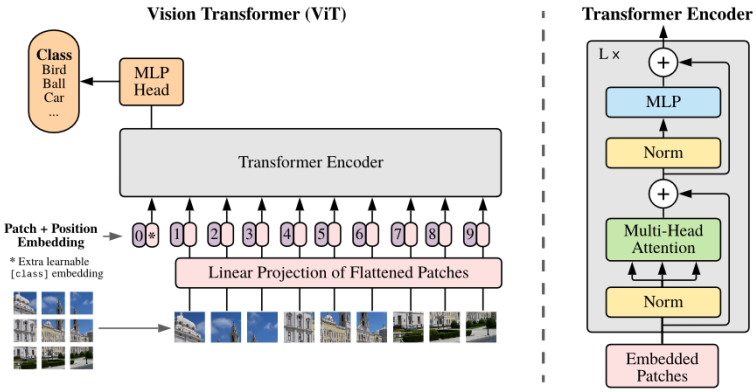


Figure 2.2: Representation of the Vision Transformer's (ViT) architecture and a Transformer encoder block. First, the image is split into fixed-size patches (usually 16×16-pixels sized patches), linearly projected into a subspace, to which a positional embedding is added. The result is then fed to a Transformer encoder in which, at each step, an extra token named class token is updated. At the end, the class token is used to perform classification. Source: Dosovitskiy et al. (2020)

els could achieve state-of-the-art performance, representing a significant advancement in the field of neural network architecture.

Vision Transformer (ViT). Transformers (Vaswani et al., 2017) were initially introduced within the field of NLP as a powerful model for capturing long-range dependencies in textual data. Unlike traditional recurrent neural networks (RNNs) (Chung et al., 2014) and long short-term memory (LSTM) networks (Hochreiter and Schmidhuber, 1997), which process data sequentially, Transformers allow for the parallel processing of input data, significantly enhancing computational efficiency and enabling the handling of long-range dependencies within the sentence. Each word of the sentence is processed simultaneously by a transformer cell and interacts with the other words by relying on a multi-head attention module. In its simplest form, the attention module (Vaswani et al., 2017) computes an attention matrix by taking for each query (word for which we are trying to find relevant information from the rest of the sentence) the dot-product between the query and all keys (words used to measure the relevance when compared to the query), followed by applying a softmax function across the key dimension. This attention matrix is then used to filter the values. Mathematically, the attention module can be written as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d}}\right)V, \tag{2.1}$$

where $Q \in \mathbb{R}^{n\times d}$, $K \in \mathbb{R}^{n\times d}$, and $V \in \mathbb{R}^{n\times d}$ are the matrices of queries, keys, and values, respectively, n the sentence's length and d the feature dimension of each word. The term $\frac{1}{\sqrt{d}}$ avoids pushing the softmax function into a saturated regime, resulting in small gradients for training. In practice, several attention heads are used in parallel, and for each head, queries, keys, and values are linearly projected. The output of the h-th attention head AH_h is defined as:

$$\text{AH}_h(Q, K, V) = \text{softmax}\left(\frac{QW_Q^h(KW_K^h)^\top}{\sqrt{d_k}}\right)VW_V^h, \tag{2.2}$$

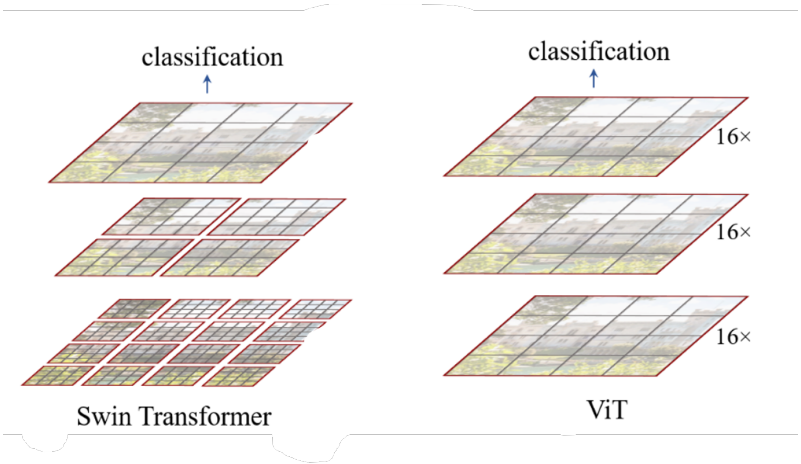


Figure 2.3: Illustration of the Swin Transformer architecture compared to the Visual Transformer architecture. The Swin architecture builds hierarchical feature maps by merging image patches in deeper layers, while ViT builds its feature map from a single low-resolution embedding. Modified version of the figure from Liu et al. (2021)

where $W_Q^h \in \mathbb{R}^{d\times d_k}$, $W_K^h \in \mathbb{R}^{d\times d_k}$ and $W_V^h \in \mathbb{R}^{d\times d_v}$ are learned projection matrices. The outputs of the H-individual attention heads are then concatenated and multiplied by another learned projection matrix $W_O \in \mathbb{R}^{Hd_v\times d}$. There exist different variants of multi-head attention. One of them, the multi-head self-attention is when $Q = K = V = X \in \mathbb{R}^{n\times d}$ and is defined by:

$$\text{SelfAH}_h(X) = \text{softmax}\left(\frac{XW_Q^h(XW_K^h)^\top}{\sqrt{d_k}}\right)XW_V^h. \tag{2.3}$$

Multi-head attention modules enable the model to capture diverse contextual relationships in the input. Transformers input is a vector, a sentence in NLP applications, where each element of the vector is a word. By analogy, applying Transformers to images would result in considering the image as a vector, in which each unit is a pixel. However, applying Transformers on pixels directly is too computationally expensive.

A solution introduced by Dosovitskiy et al. (2020) is to separate the image into 16×16 pixels patches, project these flattened patches into a subspace, and use the result as the input of the Transformer (Fig. 2.2). An extra class embedding is concatenated to the projected patch representations, and the resulting vector is fed to L Transformer blocs (right side of Fig. 2.2), which serve as feature encoder. At the end, the updated class token is fed into a classification head. The remarkable accuracy achieved by Vision Transformer on image classification has challenged the CNN's hegemony. Nowadays, Transformer's architectures are state-of-the-art for multiple computer vision tasks.

Swin. Unlike traditional Vision Transformers (ViT) that process the entire image as a sequence of patches, Swin Transformer (Liu et al., 2021) introduces a hierarchical structure as in CNNs. To do so the Swin Transformer leverages the use of non-overlapping local windows within which self-attention is computed. These windows are shifted in subsequent layers to enable cross-window connections, effectively capturing both local and global context (Fig. 2.3). This shifted window approach addresses the limitations of fixed partitioning in traditional ViTs, allowing for better modeling of fine-grained details and larger spatial relationships.

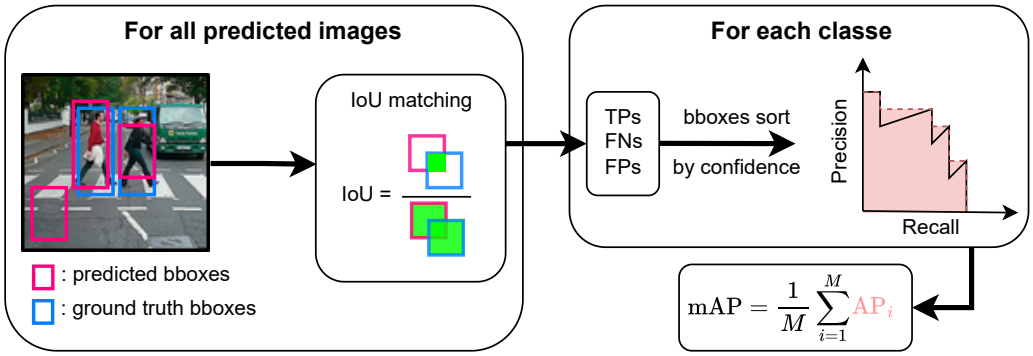


Figure 2.4: Illustration of the mAP calculation pipeline.

In recent years, image classification has become a fundamental yet simplified example for studying neural network behavior. However, its practical applicability is constrained by the limitation of predicting only a single label per image. This limitation becomes problematic when an image contains multiple relevant objects, leading to ambiguous and incomplete predictions. To address this issue, object detection has emerged as an alternative to its capability of identifying the location and labeling multiple objects within an image. Object detection provides a better scene understanding, allowing more precise depth estimation or action on objects.

2.1.2 Object detectors

Object detection aims to identify and locate multiple objects within an image, providing their categories and location in the image. It combines both semantics and regression through image classification and object localization. In object detection, for each object of interest in the image, a bounding box is drawn around the object and a class label is assigned to the bounding box.

Mean average precision (mAP) metric. The most common metric used to evaluate the performance of object detection models is the mean average precision (mAP). mAP is the average of the average precision (AP) over all classes considered. The AP is calculated separately for each class. It is the area under the convex envelope of the precision-recall curve for that class. The precision-recall curve plots precision against recall for different bounding box confidence levels. AP can be viewed as a global metric that measures the balance between precision and recall of the model's predictions. For boxes assigned to the same class, precision is the proportion of correctly predicted boxes (TPs) out of all predicted boxes (TPs + FPs), while recall is the proportion of correctly predicted boxes (TPs) out of all actual ground truth boxes (TPs + FNs). To consider a predicted box as belonging to a ground truth box, the Intersection over Union (IoU), which measures the overlap between predicted and actual bounding boxes, must be greater than a predefined threshold (usually $\text{IoU} = 0.5$). The mAP is then computed according to a given IoU threshold. Figure 2.4 illustrates the mAP calculation process.

As explained earlier, object detection combines semantic and regression tasks. In the literature, two types of architectures have been developed to solve such a task. Either a set of boxes is proposed first by a region proposal module, and then these boxes are refined and assigned to a class in a second stage, or the box proposals and the class assignment is

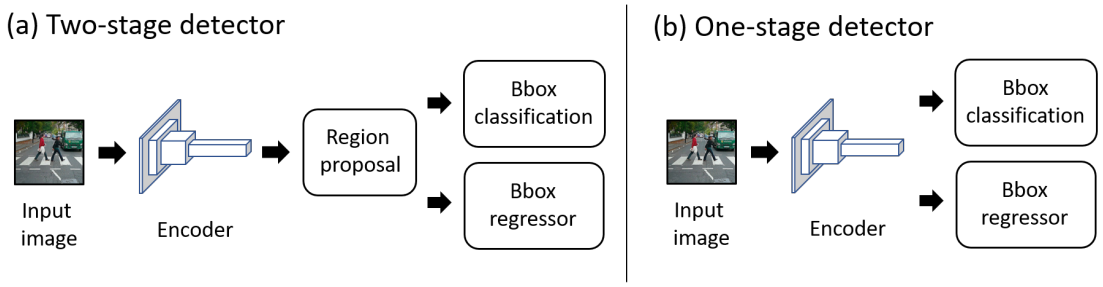


Figure 2.5: Illustration of two-stage and one-stage object detector.

done simultaneously in a one stage (see Fig. 2.5).

Region-based convolutional neural networks (RCNN). Faster R-CNN (Ren et al., 2015) is the third object detector from the Region-based Convolutional Neural Network (R-CNN) family. To understand in more detail how Faster R-CNN works, it is useful to first introduce its predecessors, R-CNN (Girshick et al., 2014) and Fast R-CNN (Girshick, 2015). The different architectures are represented in Figure 2.6.

R-CNN (Girshick et al., 2014) is one of the first papers using CNNs to perform object detection in two steps. First, a nondeep learning region proposal module, Selective Search (Uijlings et al., 2013), is used to generate a set of 2000 object proposals (boxes which may include the objects). These proposals are parts of the image that are cropped and fed independently to the CNN, which extracts a feature vector for each object proposal. Class-specific support vector machines (SVM) use these feature vectors to perform bounding box classification. Boxes proposals made by Selective Search is refined using a class-specific bounding-box regressor. Despite showing good improvement over pioneering methods like HOG (Dalal and Triggs, 2005), R-CNN has drawbacks. The Selective Search procedure and the one-by-one feature map extraction are slow and expensive in terms of memory (each proposal has to be independently proceeded by the CNN and stored). Moreover, the training procedure is complex and time consuming (takes several days).

To mitigate the slow processing speed of R-CNN, in Fast R-CNN, Girshick (2015) proposed to produce a single feature map by extracting the features of the entire image with the CNN and then to use a Region of Interest (RoI) pooling method to extract cropped features from the region proposals (see Fig. 2.6 (b)). These cropped features are used as inputs for the classification and regression module. Instead of multiple specific SVMs, the authors integrate the classification and bounding box regression steps into a single network, leveraging the end-to-end training of the backbone and the task network (detection head). The introduction of the RoI module and single-task network makes Fast R-CNN much faster (146x faster over R-CNN) and more accurate than R-CNN. However, as R-CNN, Fast R-CNN relies on external region proposal methods to generate candidate object proposals. This dependency can be computationally expensive and limits the real-time performance of the model, as these methods are not integrated into the neural network and require separate processing.

Faster R-CNN (Ren et al., 2015) becomes one of the first end-to-end object detection networks by integrating the external region proposal module into the network through the Region Proposal Network (RPN) (see Fig. 2.6 (c)). RPN is a CNN that generates bounding box proposals directly from the features extracted by the backbone. RPN predicts bounding box locations alongside an objectness score for each cell location in the feature

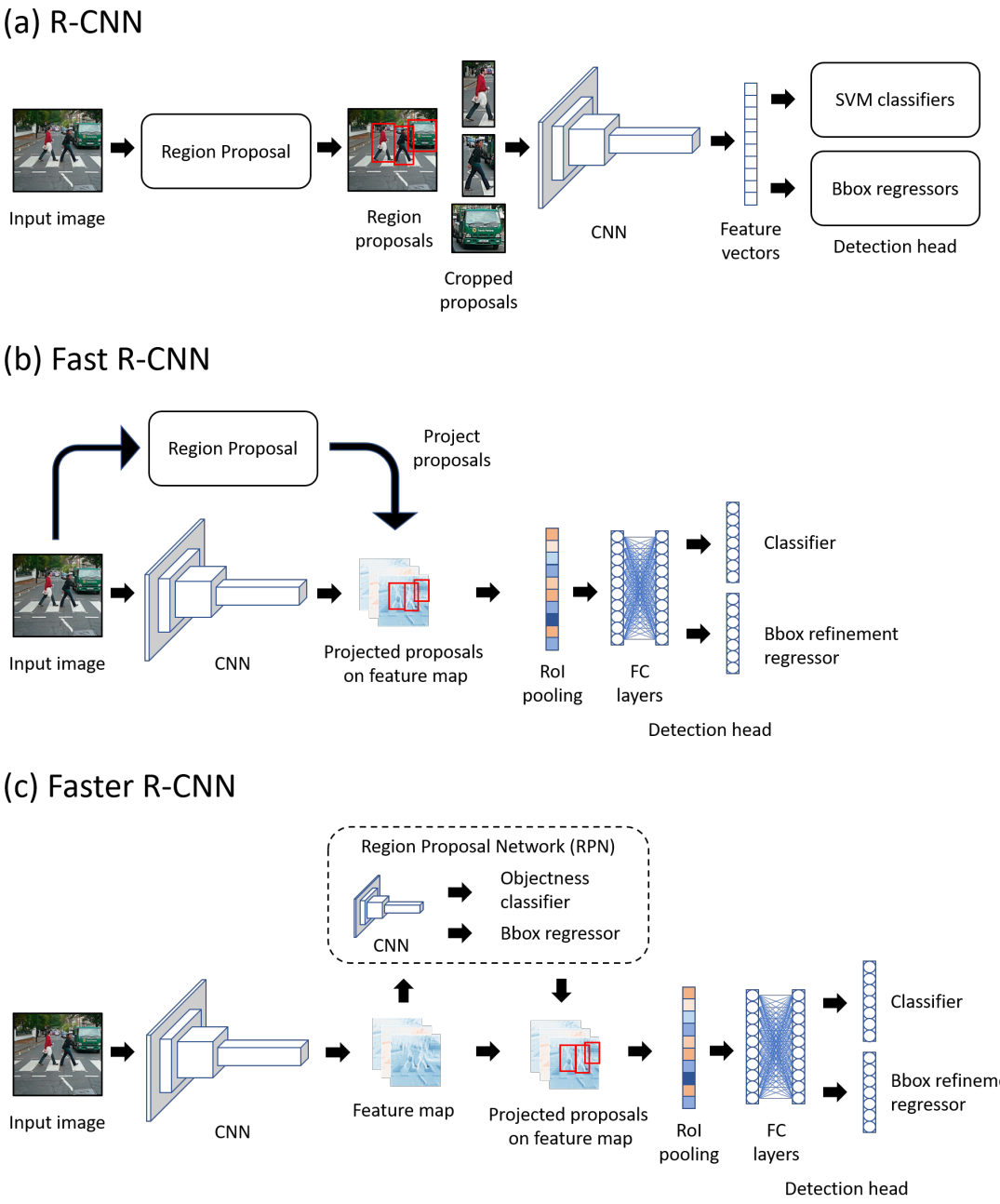


Figure 2.6: Illustration of R-CNN based networks. (a) First, a set of object proposals is generated by a region proposal module. These proposals are used to crop the region of the input image. The cropped proposals are then fed independently to a CNN, which extracts a feature vector for each cropped proposal. SVM classifiers and Bbox regressors then use the feature vector for object classification and localization. (b) In Fast R-CNN object proposals, coordinates are projected to the feature map coordinates using an RoI module. This RoI pooling module is then used to produce a fixed-size feature vector fed to fully connected layers to perform Bbox classification and regression. (c) Object proposals are no longer produced by a non-deep learning module, which processes the input image, but rather by a deep learning module, which processes the feature map produced by the CNN. Papers: R-CNN (Girshick et al., 2014), Fast R-CNN (Girshick, 2015) and Faster R-CNN (Ren et al., 2015).

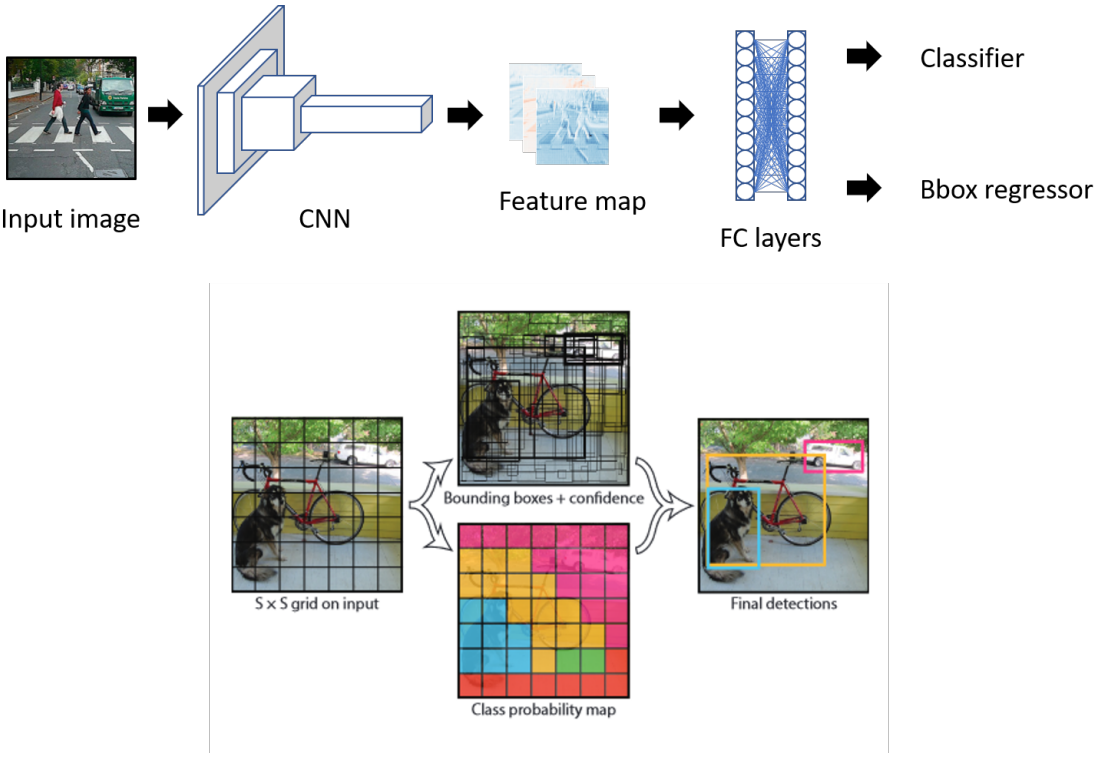


Figure 2.7: YOLOv1 network performs object detection in a one-stage manner, extracting features and performing Bbox regression (i.e., prediction of location) and classification without using proposal generation. First, the image is divided into an $S \times S$ grid. YOLOv1 predicts B bounding boxes for each grid cell, a confidence score for each box, and C class probabilities. Source: Redmon et al. (2016)

map (box probability of containing an object). It also utilizes anchor boxes of different sizes and aspect ratios to handle varying scales and shapes. Each proposal generated by the RPN is processed using RoI pooling, which converts the features inside a proposal into a fixed-size feature map. This is achieved by dividing the proposal into a fixed number of bins and using max pooling to extract the features within each bin. The RoI-pooled feature map is fed into fully connected layers for classification and bounding box refinement. The advantage of Faster R-CNN is that RPN and the fully connected layers share the same features extracted by the CNN backbone, allowing the simultaneous optimization of both tasks. This integrated approach enhances the accuracy and speeds up the inference.

However, Faster R-CNN still faces limitations regarding real-time performance required for industrial applications, such as video surveillance and autonomous vehicles. These limitations are due to the two-stage process. This limitation led to the development of new models that aim to achieve high-speed object detection with accuracy.

YOLO. To mitigate the limitation of previously introduced RCNN-based approaches, the You Only Look Once (YOLO) object detector (Redmon et al., 2016) is a new architecture enabling the real-time object detection.

Among the models achieving real-time object detection, the You Only Look Once (YOLO) object detector (Redmon et al., 2016) stands out for its novel approach. YOLO reframes the two-stage object detection pipeline into a single-step regression task: predicting bounding boxes and class probabilities directly from full images in one stage. To do so, YOLO divides an input image into a $S \times S$ grid, where in each cell YOLO predicts

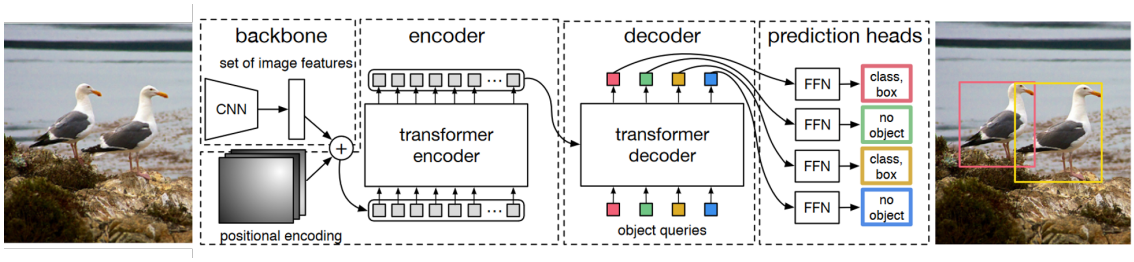


Figure 2.8: DETR detection pipeline. DETR uses a CNN backbone to extract a set of features. These features are flattened and supplemented with a positional encoding before being passed into a Transformer encoder, which further improves feature representations. These refined feature representations are fed to a Transformer decoder, which uses them to update a small fixed number of learned positional embeddings called object queries. Each object query is then passed to a shared feed-forward network that predicts either a detection (class and bounding box) or a *no object* class. Source: Carion et al. (2020)

coordinates for B bounding boxes, as well as confidence scores indicating the likelihood of the boxes containing an object (mathematically defined as $\mathbb{P}(\text{object}) * \text{IoU}^{\text{truth}}_{\text{pred}}$). For each grid cells, YOLO predicts also C conditional probabilities $\mathbb{P}(\text{class}_i|\text{object})$. These probabilities are predicted regardless of the number B of predicted boxes per cell. At test time, the conditional class probabilities and the individual box confidence predictions are multiplied to obtain a class-specific confidence score for each box:

$$\mathbb{P}(\text{class}_i|\text{object}) * \mathbb{P}(\text{object}) * \text{IoU}^{\text{truth}}_{\text{pred}} = \mathbb{P}(\text{class}_i) * \text{IoU}^{\text{truth}}_{\text{pred}}. \tag{2.4}$$

These scores encompass the probability of that class appearing in the box and how well the predicted box fits the ground truth box. The output tensor size is $S \times S \times (B * 5 + C)$. Figure 2.7 represents the architecture and the overall pipeline.

Over the past years, YOLO has emerged as a predominant architecture in real-time object detection. Many works are interested in developing new modules or learning strategies to improve the accuracy or inference time of YOLO architecture. At the time of writing this manuscript, at least ten different YOLO versions have been developed (Redmon and Farhadi, 2017, 2018; Reis et al., 2023; Wang et al., 2024).

Faster R-CNN and YOLO output a set of boxes over the image where multiple boxes can be assigned to the same object. A post-processing method called Non-Maximum Suppression (NMS) is often used to filter out potential duplicate box proposals. NMS takes as input a list of B box proposals associated with confidence scores S and an overlap threshold $\lambda \in [0, 1]$. NMS selects the object proposal with the highest confidence score and iteratively removes from B the proposals with an IoU greater than some threshold λ. The performance (mAP) of Faster R-CNN or YOLO is then measured for pre-selected NMS and IoU thresholds.

DETR. DEtection TRansformer (Carion et al., 2020) is a new one-stage object detector architecture that views object detection as a direct set prediction problem, removing the need for NMS or anchor generation encoding prior knowledge. To enable direct set prediction, DETR introduces the use of Transformer architecture (Vaswani et al., 2017) in object detection and a set-based global loss that forces unique predictions via bipartite matching. One-to-one matching between the predictions and the ground truth is achieved by using the Hungarian algorithm. An additional *no object* class is added to

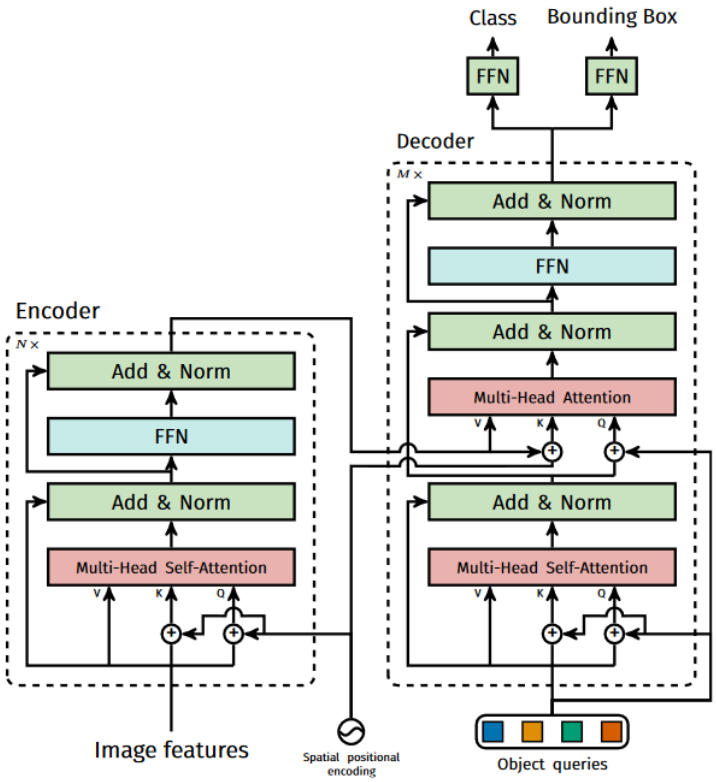


Figure 2.9: DETR's encoder and decoder. The encoder uses multi-headed self-attention to capture object interactions across the feature map. The decoder uses this feature map to update object queries after being fed to a feed-forward network to perform object detection. Source: Carion et al. (2020)

correctly match the right number of actual objects and object queries (DETR's outputs). The model then computes a loss based on this optimal assignment, combining classification and bounding box regression losses. The DETR architecture uses three modules: a CNN encoder extraction features from images, a Transformer encoder, and a Transformer decoder (Fig. 2.8).

As the Transformer encoder takes a sequence as input, features extracted by the CNN are flattened along the spatial dimension, from $z \in \mathbb{R}^{d \times H \times W}$ to $z \in \mathbb{R}^{d \times HW}$ where d is the number of filters and H and W the feature's height and width. The Transformer encoder aims to refine the features extracted by the CNN's encoder. The encoder's self-attention mechanism allows each feature vector to consider the information from all other regions of the image, effectively modeling global context and dependencies. This is particularly important for object detection, as it enables the model to understand complex spatial relationships and interactions between objects and their surroundings.

The Transformer decoder follows the standard architecture of the Transformer, transforming N input embeddings of size d through attention modules. The input embeddings are named object queries in the paper (Carion et al., 2020); they represent potential objects in the image. Initialized randomly and optimized during training, these object queries also include a positional encoding and are added to each attention layer's input. Through the decoder layers, object queries are refined through self-attention and cross-attention mechanisms (Fig. 2.9). The self-attention layers enable the model to relate different object queries to one another, allowing it to capture dependencies and interactions between po-

tential detected objects. Cross-attention layers allow the object queries to attend to the encoder's output, integrating detailed information from the image. This attention mechanism ensures that each query is contextualized from all image regions, enhancing the model's ability to focus on relevant areas for object detection.

Finally, the N refined object queries are passed through a three layers feed-forward network to predict class probabilities and bounding box coordinates for each object query.

All the object detectors discussed so far share a common component: a Convolutional Neural Network (CNN) backbone extracting features. This backbone is not trained from scratch for the specific task of object detection. Instead, it is a fine-tuned version of a pre-trained backbone, initially designed and trained for image classification tasks. Pre-training the CNN backbone, or any backbone, involves extensive training on large-scale image classification datasets, such as ImageNet (Deng et al., 2009). This allows the backbone to learn a rich set of visual features expected to generalize well across different tasks. When these pre-trained models are later fine-tuned for object detection, they start from a strong foundation of learned features, enabling them to localize and identify objects within images more effectively.

2.1.3 Network learning strategies

Another pillar of neural network performance and success is the learning strategies used to train them. In this section, we describe the different learning paradigms and also the training recipes used to learn network weights. We consider the standard notation where $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, ..., n$, are samples drawn from a joint distribution of random variables X and Y. As we are considering an image classification problem, we suppose the input is sampled from $\mathcal{X} = \mathbb{R}^{h \times w \times c}$, where $h \times w$ are the image dimensions and c is the number of channels and where the output is sampled from a set of M labels $\mathcal{Y} = \{y_1, ..., y_M\}$. Let us define by $F_\theta : \mathcal{X} \to \mathcal{P}(\mathcal{Y})$ a deep neural network parameterized by $\theta \in \Theta$ where θ are the network's parameters, Θ the parameter's space and $\mathcal{P}(\mathcal{Y})$ a probability distribution over the labels. As explained in the Section 2.1.1, the functional architecture of neural network classifiers usually follows an encoder-decoder schema. We denote by f the encoder part of F_θ and by T the decoder or the task head, which is a simple linear layer for classification (we omit the parameters notations for f and T). The encoder f maps $x \in \mathcal{X}$ to the feature space $\mathcal{S} = \mathbb{R}^{H \times W \times C}$, where H and W are respectively the height and the weight of the feature tensor and C its number of channels. Features are reduced to a feature vector by applying an average pooling layer over the spatial dimensions. The feature vector is then used by the task head to perform classification, $T : \mathbb{R}^C \to \mathcal{P}(\mathcal{Y})$, $T(z) = \text{softmax}(Wz + b)$. The objective is to learn the optimal parameters θ^* of F_θ that minimize the empirical error $\hat{E}(\theta) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{\hat{y} \neq y(x_i)}$, where $y(x_i)$ is the label associated with the sample x_i and $\hat{y} = \arg\max_{m \in \mathcal{Y}} F_\theta(x_i)_m$.

Supervised learning. Supervised learning involves training F_θ, i.e., both f and T, using the labels. As $\hat{E}$ is not differentiable, the training relies on a surrogate loss, which is generally the cross-entropy loss for classification. The goal is to find the optimal parameters θ^* by solving the following optimization problem:

$$\theta^* = \arg\min_{\theta \in \Theta} \hat{R}(\theta) = \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_{\text{CE}}(F_\theta(x_i), y_i) \tag{2.5}$$

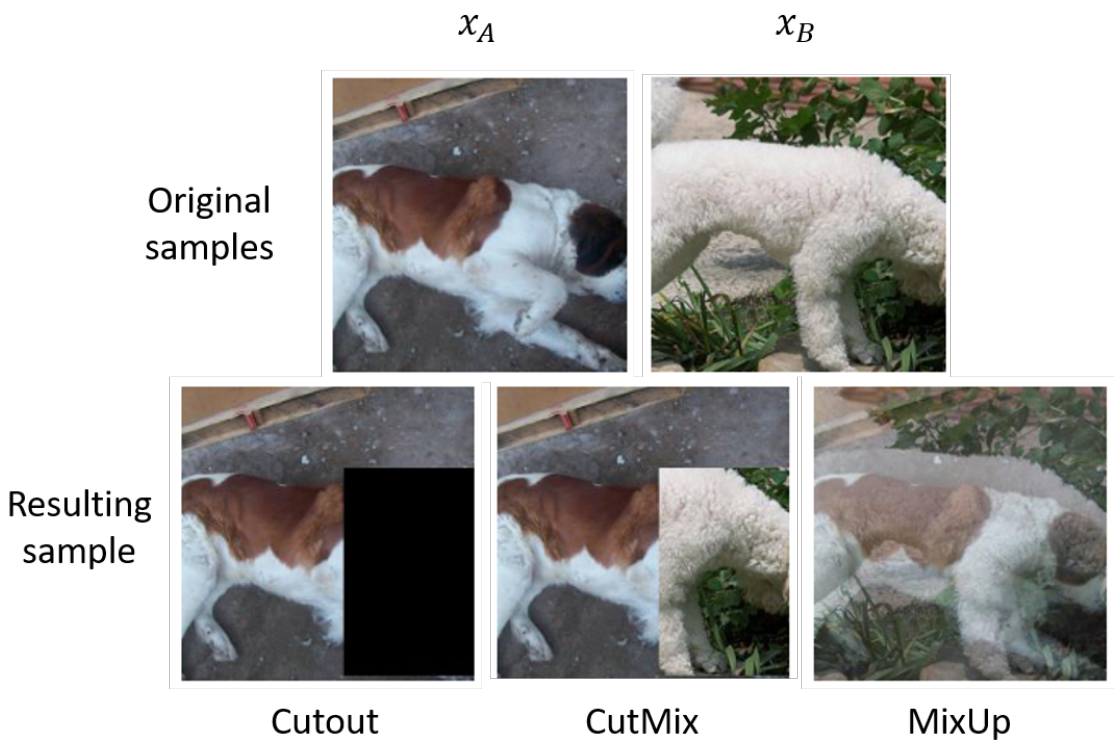


Figure 2.10: Example of image mixture methods. From left to right: Cutout (DeVries and Taylor, 2017) involves masking out a random square region of the sample, CutMix (Yun et al., 2019) combines two images by inserting a crop an image into the other and MixUp (Zhang et al., 2017) makes a convex pixels interpolation between two samples. Source: Yun et al. (2019)

In practice, this optimization is performed using gradient-based methods such as stochastic gradient descent (SGD). The parameters θ are iteratively updated according to the rule:

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta \hat{R}(\theta_t) \tag{2.6}$$

where η is the learning rate, and $\nabla_\theta \hat{R}(\theta_t)$ is the gradient of the empirical risk with respect to θ at the t-th iteration. The training continues until the network converges to a set of parameters θ^* that minimizes the loss over the training data. We obtain an "optimal" encoder and classification head that minimizes the loss over the training data and that can be used to classify new images. Once trained, the image classifier encoder can initialize an object detector encoder.

Training recipes. Different image augmentations, image mixtures, or better optimization procedures may be used to increase the performance, generalization, and robustness of neural networks. Together, they form what we call a training recipe.

Image augmentations consist of changing the visual appearance of an image while keeping its semantics unchanged. It includes the following augmentations: random cropping, random color distortions, random Gaussian blur, random color jittering, random horizontal or vertical flip, random grayscale conversion, random multi-crop, and many more[1] (Shorten and Khoshgoftaar, 2019). These augmentations can be applied alone or can be composed. Another more complex set of image augmentations is image mixture.

[1] https://pytorch.org/vision/stable/transforms.html

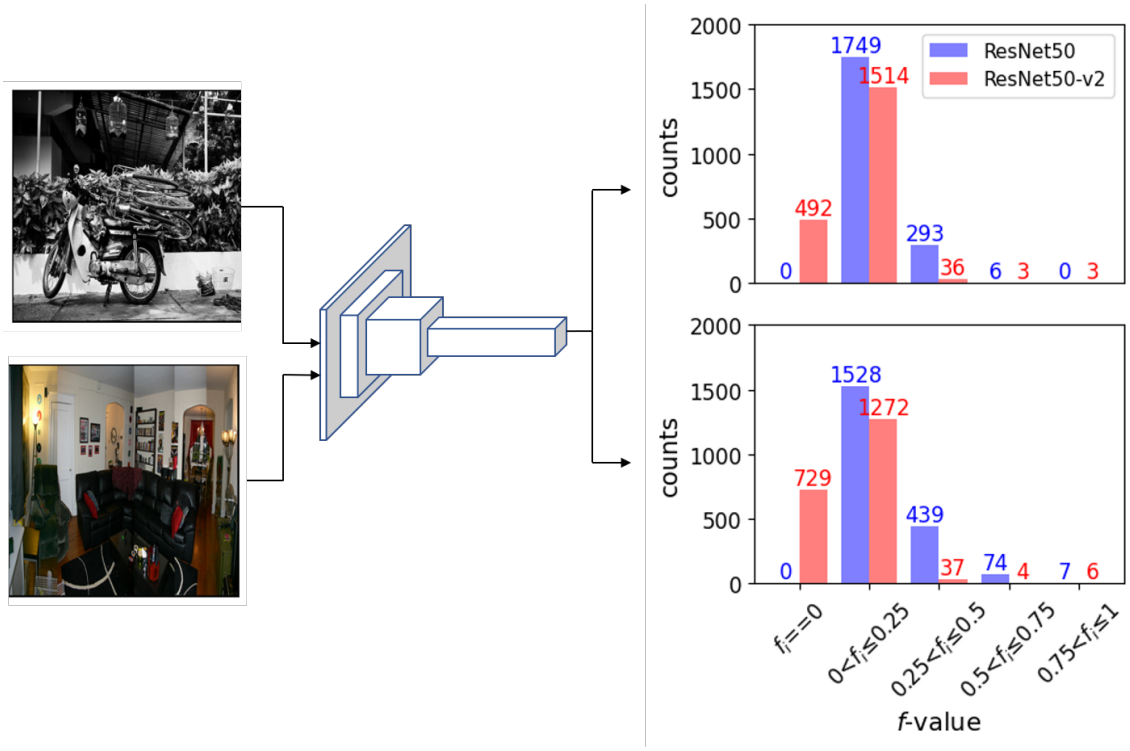


Figure 2.11: Histogram of the feature activations for two images. The features are extracted by feeding the two images independently to ResNet50 and ResNet50-v2. A spatial average pooling is applied, and the features are normalized channel-per-channel by the maximum channel value obtained on a dataset.

Image mixtures construct new images from several images (Fig. 2.10). Cutout (DeVries and Taylor, 2017) is a data augmentation technique that involves masking out a random square region of an image. Removing parts of the image forces the neural network to focus on the remaining features, thus encouraging the model to learn more robust representations. This can prevent the network from becoming overly reliant on specific features or parts of the image, which might lead to overfitting. CutMix (Yun et al., 2019) is a more advanced augmentation technique combining elements from two images. Instead of simply masking a region, as in Cutout, CutMix replaces the masked region with a patch from another image. Images and labels are mixed using the following equations:

$$\begin{aligned} x &= M \odot x_A + (1 - M) \odot x_B \\ y &= \lambda y_A + (1 - \lambda) y_B, \end{aligned} \tag{2.7}$$

where $M \in \{0, 1\}^{h \times w}$ is a binary mask, x_A and x_B are the mixed images and y_A and y_B their associated labels, $\lambda \sim \mathcal{U}[0, 1]$ and $\odot$ is the element-wise multiplication. The one value of the binary mask M is a box whose location is randomly sampled over the image and its cropped area ratio is $\frac{b_h b_w}{hw} = 1 - \lambda$. MixUp (Zhang et al., 2017) takes a different approach by linearly blending pixels of two images and their labels. This technique helps regularize the network to favor linear behavior between training samples.

These image augmentations and mixtures form the data side of training recipes. The optimization procedure can also be considered as a recipe, for example, by optimizing the learning rate using a scheduler, fine-tuning weight decay, and employing Exponential Moving Average (EMA) to update network weights.

Using a training recipe rather than another may result in different accuracy and network behavior. As an example, the repository torchvision released a new version of ResNet-50 (project made by Vasilis Vryniotis) obtained by updating the learning recipe of the original paper (He et al., 2016). This new version of ResNet50, named ResNet50-v2, shows an accuracy improvement of 4.728% (from 76.494 % to 80.858%). Although these two networks share the same architecture, the extracted features are very different. For example, if we compare the feature activations at the encoder output after applying the spatial average pooling, the two networks exhibit two different dynamics. Feature activations of the original ResNet50 are dense, i.e., all the features are activated, while ResNet50-v2 has sparse feature activations, i.e., a large fraction of the features are zero (Fig. 2.11). This observation is completely independent from the fact that these two networks may have extracted different features from the images. The most commonly used paradigm has been supervised learning, but recently self-supervised learning methods have emerged and shown competitive results in learning accurate networks.

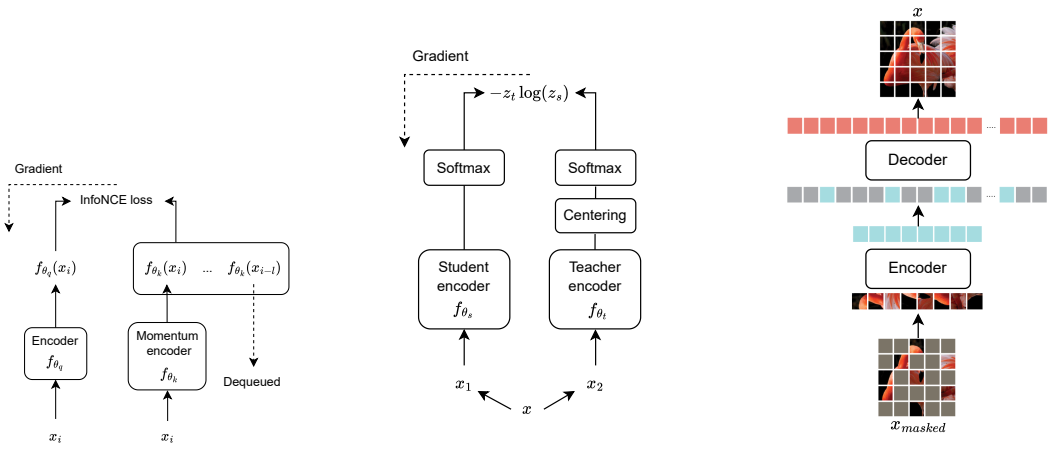
Self-supervised learning. While supervised learning has proven to be highly effective in tasks where large amounts of labeled data are available, it is often limited by the need for extensive manual annotation. In many real-world scenarios, collecting labeled data can be expensive and time-consuming. In addition, human annotations can sometimes be biased, leading to incorrect inference by the network. Self-supervised learning (SSL) has recently emerged as an alternative to learn computer vision networks. In this learning paradigm, the intrinsic structure of the data itself is used to generate a supervisory signal called a *pretext* task. This supervisory signal enables the encoder f to be trained. Instead of training f to extract relevant features for image classification, as in supervised learning, the supervisory signal of SSL methods aims at learning an encoder that may enjoy a property. SSL training is only concerned with the quality of features obtained from this supervisory signal. The quality of features is usually measured by using the feature extractor as a backbone for different downstream supervised tasks. For example, in image classification, a linear layer is added to the encoder and either the entire network or the linear head (linear-probing) can be fine-tuned in a supervised way. In the following, we will introduce three SSL methods which rely on three different SSL frameworks: contrastive learning (He et al., 2020), distillation (Caron et al., 2021) and reconstructive learning (He et al., 2022) (see Fig. 2.12). For more details on SSL the reader can refer to Ozbulak et al. (2023) survey or to this blog[2].

Contrastive learning. Contrastive learning aims to learn an encoder f to distinguish similar and dissimilar pairs of data. Given an input image x_i, a positive sample x_i^+ (which is semantically similar to x_i, e.g., a rotation of a same image), and a set of negative samples $\{x_1^-, x_2^-, ...x_N^-\}$ (which are dissimilar to x_i), the goal is to learn f by minimizing the following loss:

$$\mathcal{L}_{\text{InfoNCE}} = -\log \frac{e^{\text{sim}(f(x_i), f(x_i^+))}}{e^{\text{sim}(f(x_i), f(x_i^+)) + \sum_{j=1}^{N} \text{sim}(f(x_i), f(x_j^-))}} \tag{2.8}$$

where sim is a similarity metric, e.g., the cosine similarity. By minimizing this loss, the encoder f pulls $f(x_i)$ and $f(x_i^+)$ closer while pushing $f(x_i)$ and $f(x_j^-)$ (for all j) farther apart, thereby learning a discriminative feature space. The InfoNCE loss (Eq.

[2]https://lilianweng.github.io/posts/2021-05-31-contrastive



(a) MoCo leverages a dynamic dictionary with a queue and contrastive learning to learn visual representations. Modified from: He et al. (2020)

(b) DINO uses self-distillation to align representations between two networks, a student and a teacher. Modified from: Caron et al. (2021)

(c) MAE learnes the encoder to reconstruct images from partially masked ones. Source: He et al. (2022)

Figure 2.12: Illustrations of the three self-supervised learning techniques described.

2.8), is a popular constrastive loss, that can be interpreted as a log loss of $(K+1)$-way softmax-based classifier that tries to classify $f(x_i)$ as $f(x_i^+)$. Two significant ingredients of contrastive learning are multiple data augmentation and large batch size. Proper data augmentations are essential for creating positive samples from the source sample without modifying the semantics. At the same time, a large batch size enables a more diverse collection of negative samples, forcing the model to learn meaningful representation to distinguish different examples. However, large batch sizes can be extremely computationally expensive. An alternative way to reduce the computation time is to store the representation in memory to trade off data staleness.

Momentum Contrast (MoCo) (He et al., 2020) is an SSL method that leverages a dynamic dictionary with a queue to learn visual representations (see Fig. 2.12a). The dictionary is structured as a large First In First Out (FIFO) queue of encoded representations of data samples. As before, the input image x_i is processed by the encoder f_{θ_q}, named query encoder in the paper, while the positive and negative samples are fed to a momentum encoder f_{θ_k}. The momentum encoder is updated with the moving average of the query encoder's weights: $\theta_k \leftarrow m\theta_k + (1-m)\theta_q$, where m is a momentum coefficient. The dictionary is constructed by representations fed to f_{θ_k}. The query encoder is updated using the InfoNCE loss (Eq. 2.8) modified by a temperature hyperparameter. The MoCO method enables f_{θ_q} to learn representations by contrasting positive pairs with many diverse negative pairs.

Distillation with No Labels (DINO). While MoCo focuses on learning from a large set of negative samples, DINO (Caron et al., 2021) takes a different approach by using self-distillation to align representations between two networks, a student and a teacher, without the need for negative samples (see Fig. 2.12b). Given an input image x, two

different augmented views, x_1 and x_2, are generated using a series of data augmentation techniques. These augmented views are passed through a teacher-student architecture where the teacher network f_{θ_t} and the student network f_{θ_s} produce output representations $z_t = f_{\theta_t}(x_1)$ and $z_s = f_{\theta_s}(x_2)$, respectively. The objective is to minimize the loss $\mathcal{L}(\theta_s) = -z_t \cdot \log z_s$, which encourages the student to match the teacher's output distribution. The teacher network is updated using an exponential moving average (EMA) of the student network's weights, i.e., $\theta_t \leftarrow \lambda\theta_t + (1-\lambda)\theta_s$, where λ is a momentum coefficient. In practice, x_1 is a set of different views constructed by different distorted views or crops, and x_2 is two global views of x. The set of different views is passed through the student, while only the global views are passed through the teacher, encouraging "local-to-global" correspondences and forcing the student to learn rich features from local information. The student network is then used as the backbone.

Reconstructive learning. Another framework in SSL is reconstructive learning which includes famous methods such as Autoencoder (Hinton and Salakhutdinov, 2006) and Denoising Autoencoder (Vincent et al., 2010). In reconstructive learning, the focus is on learning two networks: one encoder that projects raw data into a low dimensional space and one decoder that from this encoded feature reconstructs the raw data. Mathematically, given an input x, the model learns an encoder f that maps the input to a latent representation $z = f(x)$, and a decoder g that reconstructs the input from this representation, $\hat{x} = g(z)$. The objective is to minimize the reconstruction loss, typically measured by a distance metric between the original input x and its reconstruction $\hat{x}$, such as the mean squared error:

$$\mathcal{L}_{\text{rec}} = \|x - \hat{x}\|^2 = \|x - g(f(x))\|^2. \tag{2.9}$$

By minimizing this loss, the model encourages f to capture the essential information needed to reconstruct the input, leading to representations that retain the most significant features of the data, which can be useful for various downstream tasks.

A recent version of the autoencoder principle is **Masked Autoencoders (MAE)** (He et al., 2022) in which networks are learned to reconstruct images from partially masked ones (see Fig. 2.12c). MAE takes advantage of the recent ViT architecture and, contrary to previous works which use networks to process entire images, MAE applies ViT only to the unmasked patches. The masked patches are removed, no mask tokens are used in the encoder. The masked patches are reintroduced in the decoder resulting in a full set of tokens. Each mask token is a shared, learned vector that indicates the presence of a missing patch to be predicted. A positional embedding is added to the mask token to add information about its location. After training, the decoder is removed and the encoder is applied to unmasked images.

2.1.4 Training dataset

In addition to the neural network architecture and learning strategies, the training data also plays a crucial role in influencing the network's weights during the learning process. However, the training datasets are fixed in computer vision and adhere to established benchmarks such as ImageNet (Deng et al., 2009) for image classification. These standardized datasets provide a consistent foundation for training and evaluating models, ensuring comparability across different approaches.

To summarize this related work section, neural networks are first characterized by their architecture. One of the key components of the architecture is the feature encoder.

This encoder can be trained by using various learning paradigms and training recipes. These learning strategies significantly influence the network's behavior and performance. Even with an identical architecture, using two different strategies may produce networks exhibiting distinct dynamics. However, regardless of the encoder training, networks exhibit vulnerability to adversarial attacks. While an attacker can design its attack to disrupt the feature extraction, historically, attacks were primarily designed to target the decoder. The rationale behind this approach is that disrupting the decision-making process of neural networks aligns with the primary objective of malicious actors seeking to compromise model outputs. In the following section, we will delve into adversarial noises and adversarial patches, two specific types of attacks that manipulate input images to deceive well-trained and safely-trained networks.

2.2 Adversarial attacks

This section presents attacks that manipulate input images to deceive non-corrupted networks. These attacks can be grouped into two types. Either the image is captured and afterward digitally corrupted, which is usually known as *adversarial example* (Szegedy et al., 2013), or an object designed to perturb the system is placed into the scene in order to be captured and is referred to *adversarial patch* (Brown et al., 2017). Both methods exploit vulnerabilities in neural networks, leading to incorrect predictions. To more clearly highlight the differences in image modifications between the two attacks, we will adopt the term *adversarial noise* rather than *adversarial example* in the following. This distinction will help provide a more precise understanding of the underlying mechanisms. Historically, the first discovered attack was an adversarial noise.

The study of adversarial noises can be traced back to Biggio (2013) and Szegedy et al. (2013) works, who first highlighted the vulnerability of well-trained ML systems to small, intentional input perturbations. The perturbed input (Szegedy et al., 2013), denoted by x_{adv} in the following, is defined for each benign input x as

$$\begin{aligned} x_{adv} = x + \delta(x) \quad \text{s.t} \quad & \arg\max_m F_\theta(x_{adv})_m \neq \arg\max_m F_\theta(x)_m \\ \text{and} \quad & d(x, x_{adv}) \leq \varepsilon, \end{aligned} \tag{2.10}$$

where $\delta(x)$ denotes the intentional perturbation, F_θ the image classifier returning class probabilities, $d(x, x_{adv})$ a distance, and ϵ a strictly positive constant. As defined in Eq. 2.10, the objective of the attacker is to identify a perturbation, represented by the function $\delta(x)$. Applying the perturbation to the input image x will result in a modified image, denoted as x_{adv}, which will be misclassified by the classifier. In this section, we will focus on attacks targeting neural networks performing classification, but attacks can be extended to another type of classifiers (e.g., KNNs, decision trees, ...) (Biggio, 2013; Papernot et al., 2016) and can be defined to target other tasks (e.g., object detection, semantic segmentation,...).

The norm constraint on $\delta(x)$ can ensure the imperceptibility of the attack and preserve the semantics of the input. Preserving the semantics of the input is essential to consider that an attack is successful. For instance, ambiguity arises if an image initially classified as a dog is altered so that even humans can no longer recognize it. The definition of a distance allowing to encode such properties is challenging and is still an active field of work (Sharif et al., 2018; Wong et al., 2019). A proxy adopted by the community is to rely on ℓ_p-distance where $p \in \{0, 1, 2, \infty\}$ defines the norm-constraint on δ. The ℓ_0-norm

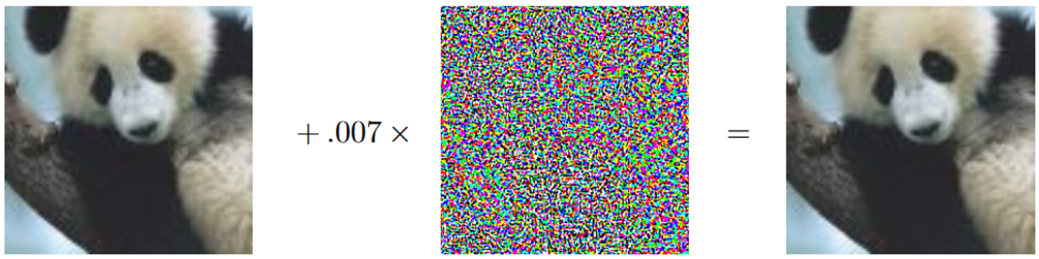


Figure 2.13: Pipeline perturbation using an adversarial noise. By adding a small adversarial noise to the image, the resulting image is now classified as *gibbon*. The 0.007 magnitude corresponds to the smallest bit change of an 8 bit image encoding. The adversarial noise is generated using the FGSM method (Eq. 2.14). Source: Goodfellow et al. (2014b)

($p = 0$) measures how many pixels are modified in the image, the ℓ_2-norm computes the Euclidean distance between x_{adv} and x while the ℓ_1-norm measures the absolute distance and finally the ℓ_∞ considers the maximum pixel value difference between attacked and clean images. The main optimization schemes that have been proposed to compute the perturbation $\delta(x)$ are presented in the following section.

2.2.1 Adversarial noise generation

Let us introduce some notations. The goal is to construct a perturbation $\delta(x)$ that, when added to x, makes the resulting image, $x_{adv} = x + \delta$ misclassified by the network F_θ (see Fig. 2.13). In the following, we will condense the notation from $\delta(x)$ to δ. The initial approach for generating adversarial perturbations deviated slightly from the definition in Eq. 2.10. In their work, Szegedy et al. (2013) generated adversarial noises by seeking the minimal perturbation, unconstrained by distance, through solving the following optimization problem:

$$\begin{aligned} \arg\min_{\delta} \quad & ||\delta||_p \\ \text{s.t} \quad & \arg\max_{m} F_\theta(x+\delta)_m = y_{tgt} \\ & x + \delta \in [0,1]^{h \times w \times c}, \end{aligned} \tag{2.11}$$

where y_{tgt} denotes the target class that the attacker wants the model to output. In practice, solving directly the Eq. 2.11 is challenging, leading the authors to propose the following relaxation:

$$\begin{aligned} \arg\min_{\delta} \quad & c||\delta||_p + \mathcal{L}_{\text{CE}}(F_\theta(x+\delta), y_{tgt}) \\ & x + \delta \in [0,1]^{h \times w \times c}, \end{aligned} \tag{2.12}$$

where $c > 0$. The constraint $x + \delta \in [0,1]^{h \times w \times c}$ ensures that the perturbed image remains an the image value range ($x + \delta \in [0, ..., 255]^{h \times w \times c}$ for unnormalized images). However, such a loose constraint may result in modifying the semantics of the perturbed image. Rather than seeking the smallest adversarial perturbation as in Eq 2.12, the optimization

objective changes for a predefined ε-budget optimization. This ε-constrained design is the current standard benchmark to create adversarial noises, and benchmarks are settled for different ε-values depending on the ℓ_p-norm. The new objective becomes finding an adversarial perturbation δ within a budget of ε usually formalized as follows:

$$\arg \min_{\delta:||\delta||_p \leq \varepsilon} \mathcal{L}_{\text{CE}}(F_\theta(x+\delta), y_{tgt}). \tag{2.13}$$

Since Szegedy et al. (2013) revealed the adversarial noises vulnerability, numerous methods have been developed to improve their effectiveness. The next section describes the key methods from the literature.

Fast Gradient Sign Method (FGSM). To create ℓ_∞-bounded adversarial noises (see Fig. 2.13), Goodfellow et al. (2014b) introduced a one-shot optimization technique named the Fast Gradient Sign Method (FGSM), which is defined by the following equation:

$$\delta = \varepsilon \text{sign}(\nabla_x \mathcal{L}_{\text{CE}}(F_\theta(x), y_{tgt})), \tag{2.14}$$

where sign is the sign function returning either 1 or -1. By applying the sign of the gradient, each image value is perturbed by the maximum possible amount along the direction of the gradient.

Building on the FGSM approach, Kurakin et al. (2018) proposed an enhanced iterative version called the Iterative Fast Gradient Sign Method (I-FGSM). In I-FGSM, FGSM is applied multiple times with a smaller step size α, updating the perturbation iteratively and clipping the pixel values at each step to ensure they remain within the ε-neighborhood of the original image. The update rule is given by:

$$\delta^{(j+1)} = \text{clip}_\varepsilon \left(\delta^{(j)} - \alpha \text{sign}(\nabla_{\delta_{(j)}} \mathcal{L}_{\text{CE}}(F_\theta(x+\delta_{(j)}), y_{tgt})) \right), \tag{2.15}$$

where j denotes the current iteration step and $\delta^{(0)}$ is initialized to zero. This method creates a more disruptive adversarial noise than FGSM by refining the perturbation over multiple iterations.

Projected Gradient Descent (PGD). I-FGSM is a particular case of PGD (Madry et al., 2017). While I-FGSM is specifically designed to generate ℓ_∞-perturbations, PGD can also generate ℓ_1 and ℓ_2 perturbations. Like I-FGSM, PGD updates the input iteratively in the gradient direction and includes an additional projection step with respect to the considered ℓ_p-distance. Additionally, PGD involves random initialization of the perturbation within the norm-ball, which may avoid poor local minima. PGD is a strong baseline for evaluating the system's robustness.

Adaptive Projected Gradient Descent (APGD). Croce and Hein (2020) identified several limitations in the PGD method that may result in local minima attack performance. These limitations include the use of a fixed step size, which may not be optimal across all iterations; a budget-agnostic optimization that does not adequately account for the total perturbation budget; and a trend-unaware optimization, which fails to adapt the direction of updates based on the progress of the attack. To address these issues, the authors proposed the Adaptive Projected Gradient Descent (APGD) method (Croce and Hein, 2020). The APGD is based on a two-phase scheme: an initial exploration phase is used to find

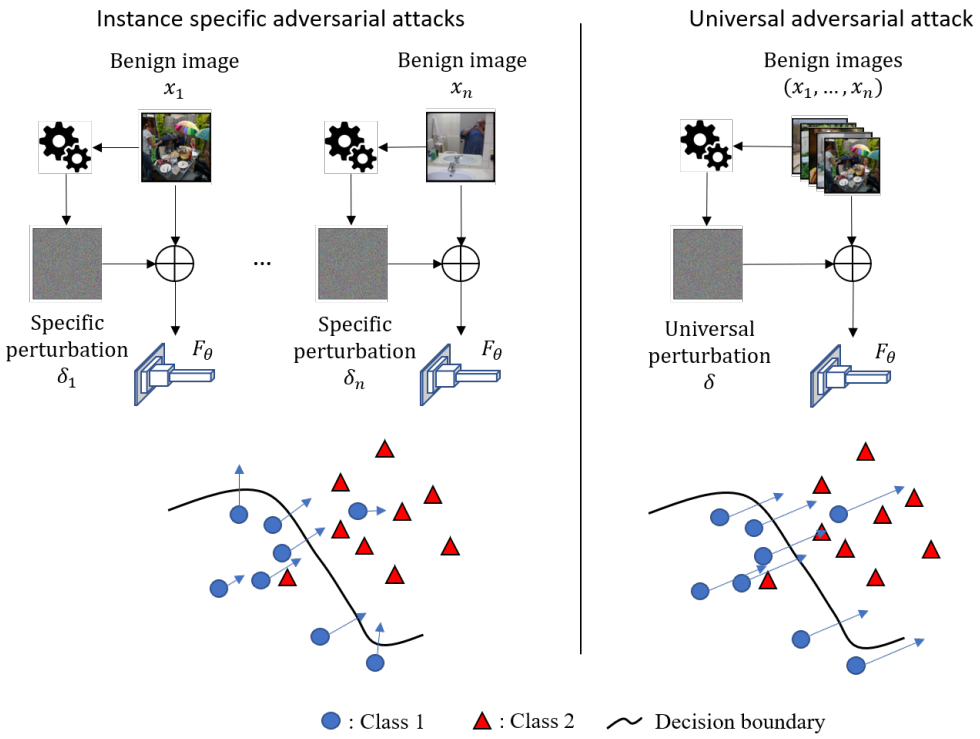


Figure 2.14: Diagram of instance specific attacks and universal attacks and schematic illustration of their impact. Instance-specific attacks consist of independently finding an adversarial perturbation $\delta(x)$. This adversarial perturbation is designed to perturb an unique sample x by crossing the nearest decision boundary. A universal attack is designed from a set of images. The universal attack seeks a common direction that fools the set of images. The bottom diagrams illustrate an example where perturbations are designed to push blue points to be classified as red.

good initialization points; it followed by an exploitation phase to optimize the perturbation using the knowledge accumulated during the exploration. The knowledge exploration allows a dynamic learning rate tuning during the exploitation and a restart strategy allowing to roll back to the best points if the attack does not improve over some iterations.

2.2.2 Attack properties and knowledge

This section describes the various properties that characterize an attack. They are intrinsically linked to the attacker's objectives and the extent of the existing knowledge about the targeted system.

Instance specific vs. universal. Instance specific attacks consist in designing an attack tailored to a particular input sample. Mathematically, for each input sample x, the goal is to find an adversarial perturbation $\delta(x)$, often generated by one of the previously mentioned methods. For example, if the attacker is presented with a set of n target images, they must generate n distinct perturbations, one for each image independently (refer to Fig. 2.14, left). In contrast, designing an universal attack consists of finding an unique perturbation δ that fools the model over multiple input samples (Moosavi-Dezfooli et al.,

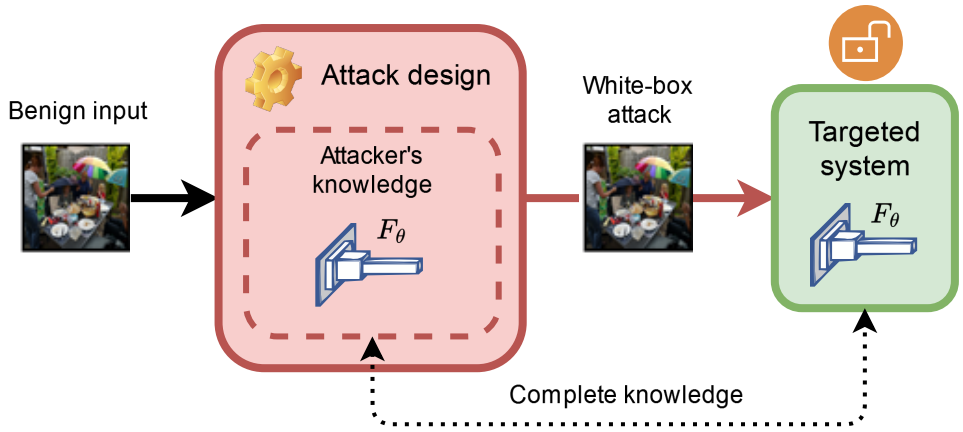


Figure 2.15: Diagram of the white-box attack scenario. The attacker has full knowledge of the target system and can adapt its attack to increase its harmfulness.

2017). The universal perturbation is usually obtained by solving:

$$\arg \min_{\delta:||\delta||_p \leq \varepsilon} \mathbb{E}_x \left[\mathcal{L}_{\text{CE}}(F_\theta(x+\delta), y_{tgt}) \right]. \tag{2.16}$$

The perturbation impact of instance-specific attacks and universal attacks is different. Instance-specific attacks often seek the nearest class decision boundary to cross, while universal attacks search common gradient directions to cross the class decision boundary (Fig. 2.14).

Untargeted vs. Targeted. One of the fundamental distinctions in the adversarial objective is whether attacks are untargeted or targeted. In untargeted adversarial attacks, the attacker aims to cause the model to produce an incorrect prediction without steering it toward a particular wrong label. The focus is simply on inducing an error, such as getting the model to misclassify a cat as anything other than a cat. Untargeted attack performance is then measured by the error rate. On the other hand, targeted adversarial attacks aims to fool the model to predict a predefined label (called the target class). For example, if the target class is *dog*, the perturbed image must be classified as a dog to consider the attack successful. The performance of targeted attacks is usually measured by the targeted success rate (tSuc) metric defined as follows:

$$\text{tSuc} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_{\arg\max_m F_\theta(x_i+\delta)_m = y_{tgt}}, \tag{2.17}$$

where N is the number of attacked images. Generating targeted attacks can be significantly more challenging depending on the target class. For example, changing the network's decision from *cat* to *toaster* is more difficult than changing it to *dog* or a similar animal.

White-box vs. black-box. Attacks can be divided into two categories, white-box and black-box, depending on the knowledge available about the target system.

In the **white-box** scenario, attackers could access the model, its architecture, parameters, and the input data (see Fig. 2.15). With this information, the attacker can adapt and exploit the system to cause the most harm. However, the white-box scenario is less likely

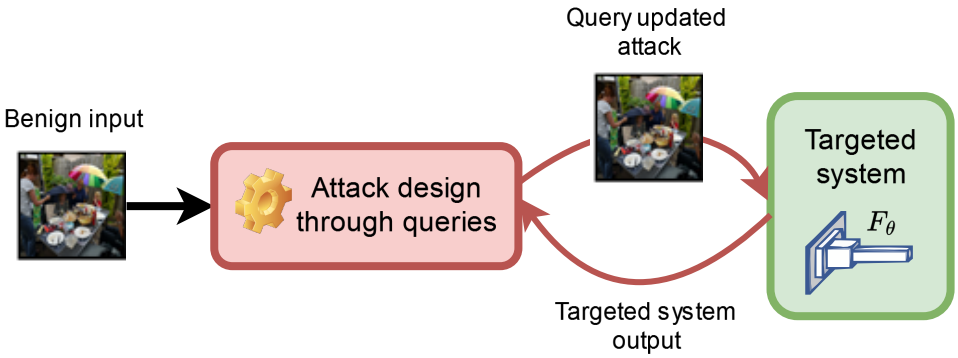


Figure 2.16: Diagram of the query-based attack scenario in which the attacker iteratively updates its attack through querying the targeted model. The targeted model itself is not available.

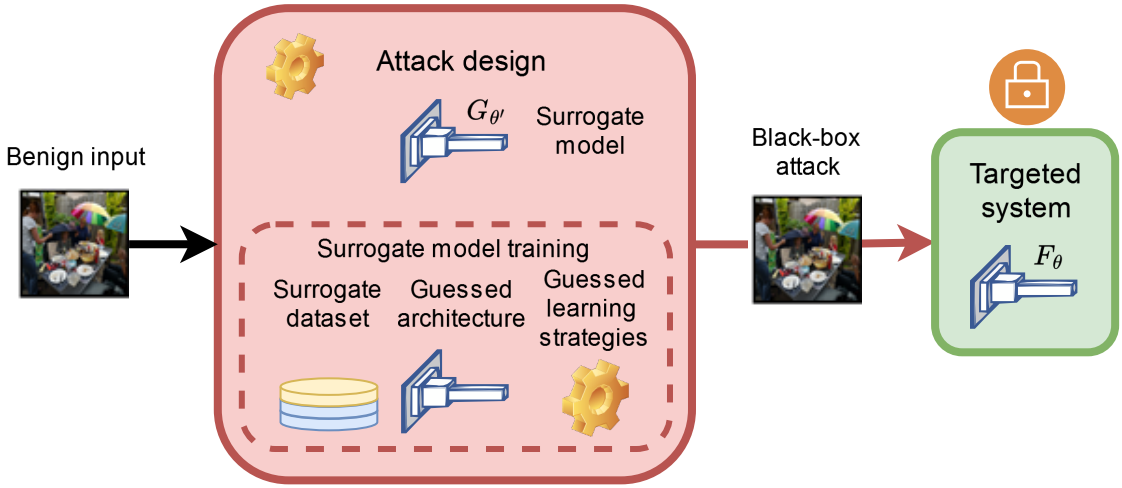


Figure 2.17: Diagram of the transfer-based attack scenario in which the target system is unavailable. The attack is built on a surrogate model, which may be trained using some possible knowledge (training dataset, architecture, learning strategies).

as the developers should hide information about the target system. This scenario is often used as a worst-case scenario to evaluate new defenses. In addition, white-box scenarios provide a controlled environment for testing new attack methods and evaluating their effectiveness before their extension to black-box settings which are more representative of real-world conditions.

In the **black-box** setting, attackers have much more limited access to knowledge about the network. In general, the attacker does not know the model architecture, its parameters, and the training data beyond what is publicly available. Two major categories of black-box attacks are query-based attacks and transfer-based attacks. Query-based attacks (Ilyas et al., 2018) assume that the attacker can query the target model several times and uses this information to iteratively improve the attack (Fig. 2.16). These attacks are particularly suited to target online APIs such as Amazon or Google Cloud Vision. However, their effectiveness can be mitigated by limiting the number of queries or introducing a delay between successive queries.

Transfer-based attacks (Papernot et al., 2016) refer to a class of attacks that are designed on a *surrogate* or a *source* model and then used to fool a target model (Fig. 2.17). In this setting, the target model cannot be queried. The ability of an adversarial attack

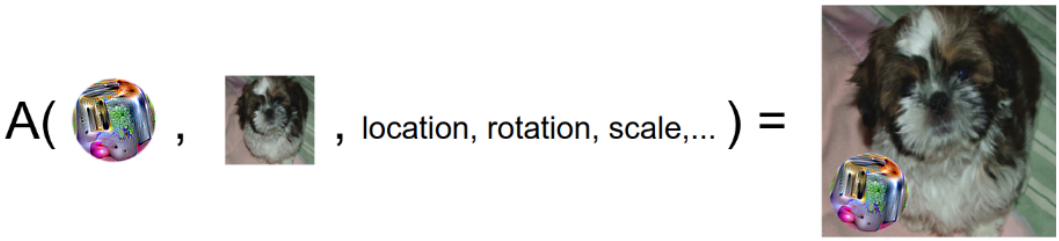


Figure 2.18: Illustration of the patch application operator. The operator takes as input a patch, an image, a location, and a patch transformation (such as scale and rotations) and applies the transformed patch to the image at the given location. Source: Brown et al. (2017)

to fool several models without being trained on them is called *attack transferability* or *transferability*. In their pioneering work, Szegedy et al. (2013) showed that adversarial noises exhibit cross model and cross training-set generalization. A significant portion of these adversarial noises can mislead several networks, even when these networks have different architecture, parameters, or are trained on different datasets. This finding underscores the transferability of adversarial noises, revealing that they are not merely artifacts of overfitting to a specific model or training set, but rather a broader vulnerability inherent to many machine learning models (Papernot et al., 2016). The study of adversarial attack transferability is crucial, not only for preventing security vulnerabilities, but also for gaining deeper insights into the behavior of deep learning models. The effectiveness of transfer-based attacks can be evaluated by varying the "proximity" between the surrogate and target models. By studying different levels of similarity, such as identical architectures with varying initializations or training recipes, researchers can gain valuable insights into the factors that influence the transferability of adversarial noises and the underlying behavior of deep learning models. Deeper explanations and details about transfer-based attacks are presented in Chapter 4.

So far, we have introduced how to generate adversarial noises and what may be their properties. While these numerical attacks can be effective in specific contexts, such as fooling API systems, their effectiveness is limited when targeting real-world systems like autonomous vehicles, where the attacker would need direct access to the system's inputs. In contrast, adversarial patches represent a more practical physical attack, as they rely on inserting visually detectable patterns or objects into a scene, making them well-suited to disrupt real-world systems without the access to the target system.

2.3 Adversarial patch attack

Physical attacks were first explored in the work of Sharif et al. (2016) and Eykholt et al. (2017), who demonstrated networks vulnerability to adversarial inputs in the physical world. These studies showed that modified eyeglasses or small stickers, could deceive real-world systems like facial or stop sign recognition, respectively. These perturbations were obtained using Equation 2.12 and constraining δ to a small part of the image as either a sticker or eyeglasses. Additional losses were added to enhance the perturbations physicality. While these works enabled the generation of physical perturbations, they were still norm-constrained, limiting their attack optimization.

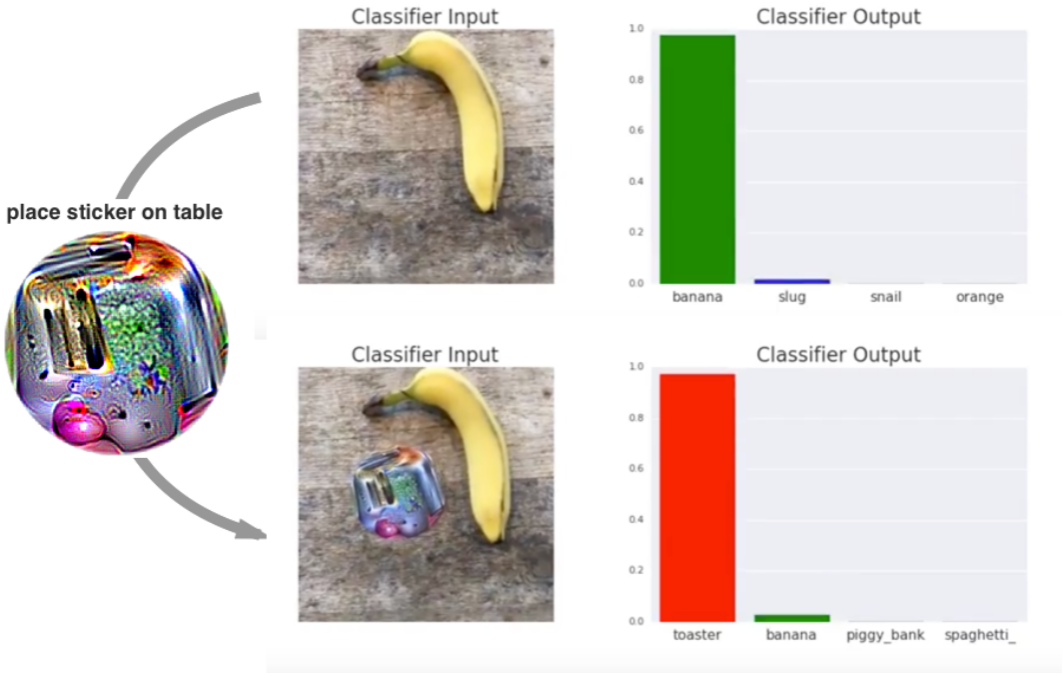


Figure 2.19: Example of a real-world attack against an image classifier. The benign image is classified as a *banana* with 97 % confidence (top plot). When the patch is physically placed on the table, the resulting image is classified as a *toaster* with 99 % confidence (bottom plot). See the following video for a full demonstration: here. Source: Brown et al. (2017)

Instead of focusing on small additive adversarial noise, Brown et al. (2017) constrained the optimization procedure to a small region of the image, similar to the approach of Sharif et al. (2016) and Eykholt et al. (2017), but allowed the perturbations to be unconstrained in magnitude, leading to more flexible and potentially stronger attacks. This type of attack, constrained in space but unconstrained in magnitude, is called adversarial patch attacks (APAs).

Let $\mathcal{T}$ be a distribution over transformation (e.g., rotations, scaling, blur, ...) and E a distribution over locations. To obtain a patch (called GAP in the paper), Brown et al. (2017) modify Equation 2.12 to solve:

$$\arg\min_{\delta} \mathbb{E}_{x\sim X, t\sim\mathcal{T}, e\sim E}\left[\mathcal{L}_{\text{CE}}\left(F_\theta(A(\delta, x, e, t)), y_{tgt}\right)\right], \tag{2.18}$$

where X is the image distribution and $A(\delta, x, e, t)$ the patch applicator operator in an image x where δ is the patch, t is a ensemble of patch transformations and e is the patch location in the image x. Figure 2.18 illustrates how the patch is digitally inserted in images by the patch application operator A. The expectation over transformations and locations is a variant of the Expectation over Transformations (EoT) from Athalye et al. (2018) and is designed to make the patch more robust to physical transformations while the expectation over images (as in Eq 2.16) enhances the patch universality. This regularization strategy encourages the trained patch to work regardless of the scene in which the patch is placed. Brown et al. (2017) showed that the resulting patch was capable of fooling several ImageNet classification models. They demonstrated the real-world threat of their attack by printing the patch and placing it near objects. When placed in the scene, the patch consistently misled the image classifier to categorize the object as a toaster -

precisely as intended, since the patch had been specifically designed to cause the model to output *toaster* as the predicted label (Fig 2.19).

Other works studying APAs designed to fool image classifiers develop new methods either to increase the fooling effectiveness of the patch (Karmon et al., 2018) or its inconspicuousness (Liu et al., 2019; Doan et al., 2022; Casper et al., 2022). However, real-world systems such as autonomous vehicles rely on more sophisticated computer vision tasks like object detection and semantic segmentation. Consequently, research on APAs should go beyond image classification and study whether these more complex tasks are vulnerable to APAs.

2.3.1 APAs for object detection

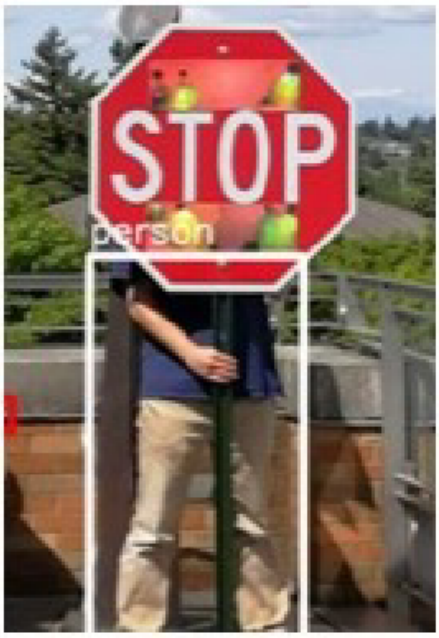
Attacking object detectors are explored in several works considering the different attack scenarios. In the first published studies, patches are directly applied on the targeted object. The first works on patch-based attacks against object detectors target stop signs detection. Song et al. (2018) develops stickers which are applied on stop signs, and fool YOLOv2 to detect correctly. They show the real-world applicability of their inhibition attack in indoor and outdoor settings (Fig. 2.20a). Another stop sign attack (Chen et al., 2019) designs APAs to change the detection label of the stop sign. The authors use the change-of-variable attack described in Carlini and Wagner (2017) and the Expectation over Transformation technique (Athalye et al., 2018) to change the red background of stop signs to fool Faster-RCNN. The new stop sign is then classified as *person* or *sports ball* (Fig. 2.20b). Both of these works illustrate the effectiveness of their patch when facing different physical conditions.

Instead of creating a patch that hides a stop sign, Thys et al. (2019) creates a patch causing the disappearance of people when they wear it (see Fig. 2.21). This particular type of patch is frequently denoted as *invisible cloak* in the APA's literature. Designing a patch allowing to hide people is much more challenging than designing a patch hiding stop signs. The person class diversity is richer; people's appearance can vary from one individual to another, however stop signs remain practically the same. Moreover, the image context can change significantly compared to stop signs that are always placed near a road.

Depending on the context, suppressing detection only on objects close to the patch can be too constraining. In a video surveillance setting, it is an issue if the hacker can become invisible because of a patch. However, in autonomous driving, the attacker's goal may not be to become invisible to the vehicle but rather to manipulate the system in other ways. For instance, a patch on a wall near a pedestrian crossing could pose significant risks by hiding people at long range. This example illustrates that, in some contexts, the main issue is the contextual effect of the patch.

2.3.2 Contextual adversarial patches

A *contextual adversarial patch* refers to a patch designed to disrupt object detection without directly overlapping with the targeted object itself. The attack is achieved by exploiting the network's reliance on background information. The contextual attack scenario presents a more complex challenge than the invisible cloak setting. Ideally, neural networks should not rely on background information when detecting large objects in images, which inherently limits the potential effectiveness of contextual attacks.



(a) Example of the stop sign inhibition attack in real-world conditions. The APA is designed to fool the detector to detect stop signs. Source: Song et al. (2018)

(b) Example of the stop sign targeted attack in real-world conditions. The APA is designed to sway the detector to output another class, here *person*. The unattacked left stop sign is correctly detected, while the attacked one on the right is detected as *person*. Source: Chen et al. (2019)

Figure 2.20: Examples of two adversarial patch attacks developed to fool stop sign detectors

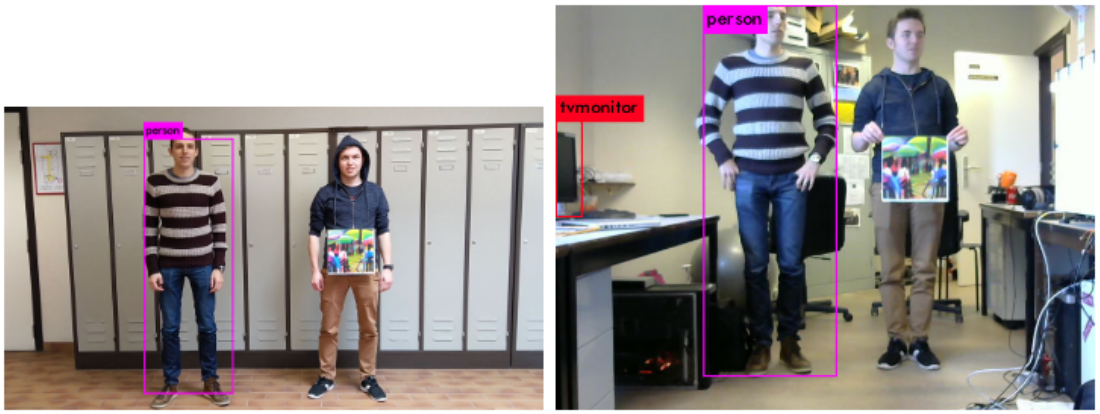


Figure 2.21: Two examples of real-world footage perturbed by an adversarial patch attack. The person wearing the patch is hidden from the object detector while the other is detected. The authors provided a demonstration of their attack here. Source: Thys et al. (2019)

Contextual patch attacks were first explored by Liu et al. (2018). Instead of designing a loss from scratch for the attack, they use the YOLOv2 loss by redefining the ground truth at the patch localization. They train the patch to minimize the redefined YOLOV2 loss, causing the disappearance of objects. This causes also the detection of the patch as an object of interest. However, their patches were never clipped to the image range, resulting in *NaN* values on several pixels of the patch. Lee and Kolter (2019) study the Dpatch attack in feasible physical conditions and compared it to their new attack. Considering a maximization problem of the YOLOv2 loss over the ground truth, they outperformed the Dpatch method and showed real-time attack success. To only measure the contextual effect of patches, Saha et al. (2020) introduce the idea of removing false positives on the patch. Consequently, only the patch contextual effects are measured by the mAP. They also develop attacks and a defense mechanism for contextual adversarial patches. An universal blindness attack targeting one chosen class is also proposed alongside an objectness attack fooling bounding box proposals, and a targeted attack fooling an object to be classified as another class.

Another line of work on APAs studies if a patch can perturb dense prediction tasks like semantic segmentation or monocular depth estimation. Nesti et al. (2022) designed a patch that reduces the baseline model accuracy of a semantic segmentation model. However, they show that the effect of the patch is mitigated when physically implemented. Yamanaka et al. (2020) introduces a patch that successfully targets a monocular depth estimation model. The security of automated lane centering and infrared-based systems have been studied in Sato et al. (2021) and Wei et al. (2023), respectively.

2.4 Conclusion

This chapter provides the necessary background for understanding the following chapters, which detail our contributions. Below, we summarize the key points to keep in mind:

- Neural networks for image classification and object detection are built from several blocks. These blocks can be convolutional layers, transformer blocks, or a combination of both. The image encoder is the core of modern computer vision networks. It is designed to extract useful features to perform the task at hand;

- Different learning strategies can be used to learn the image encoder. These learning strategies could have a strong impact on the performance and behavior of the resulting network;

- Safely and well-trained networks are vulnerable to adversarial image modification. Depending on the attack knowledge and objective, several properties can qualify the attack. The attack can be untargeted or targeted, instance specific or universal. The targeted system can be accessible, and the attacker uses this knowledge (white-box scenario), or it can be unavailable (black-box scenario). In the black-box transfer-based scenario, attacks are designed on a source model and then used to fool a target model;

- Adversarial patch attack relies on printing and adding a small object to the scene. By its physical aspect, this attack is well-suited to target physical systems;

- APA against object detectors can be characterized as either an invisible cloak, applied on the targeted object, or contextual, applied without directly overlapping with the targeted object itself.

In the following chapters, we provide chapter-specific material to better introduce our contributions. In Chapter 3, we will define what makes a patch critical for a real-world system.

Chapter 3

Definition of a critical adversarial patch attack

The objective of this chapter is to define what is a critical patch, outline its key properties, and establish a process for evaluating them. Following a discussion of the underlying motivation, we provide a definition of the properties that characterize a critical patch. We detail the evaluation criteria used to assess these properties and, consequently, determine the level of patch criticality. Then, we review the state-of-the-art techniques enhancing patch criticality and characterize them according to our evaluation criteria. This evaluation procedure allows us to identify the limitations of previously developed APAs. This chapter is based on our contribution Labarbarie et al. (2022) published at the workshop AI Safety hosted at the IJCAI-ECAI 2022 conference.

3.1 Motivation and definition

There exists a growing literature on patch-based adversarial attacks. These works focus on designing APAs for different targeted tasks like: image classification (Brown et al., 2017), object detection (Thys et al., 2019) and semantic segmentation (Nesti et al., 2022). Several attack scenarios are also studied, e.g., a patch applied on the struck object (Thys et al., 2019) and contextual patch (Saha et al., 2020)). Different evaluation conditions such as numerical and physical patches could also be considered. We have found that this wide variety of patch evaluations does not provide a clear understanding of the proper level of patch criticality. However, it is of utmost importance for concerned researchers and industries to be able to define and verify the expected properties of such critical APAs.

Although APA's design results from numerical optimization, its significance is based on its physicality through real-world applications (e.g., printing and placing on stop signs). Figure 3.1 schematizes the APA's attack pipeline against a real-world system. Once the patch is numerically designed, the first step is to print it. This process can introduce color discrepancies, resulting in the first potential source of attack mitigation. The printed APA is then placed in an environment to deceive the chosen targeted real-world system running in this environment, e.g., the patch is placed near the road when targeting an autonomous vehicle. The targeted environment may be volatile, introducing contingencies, e.g., different weather conditions leading to a second source of attack mitigation. Finally, to be processed by the system, the patch should be placed in the receptive field of the system's sensors and may be adapted to various angles. The recording conditions and the camera characteristics may also diminish the patch's effectiveness. These transformations,

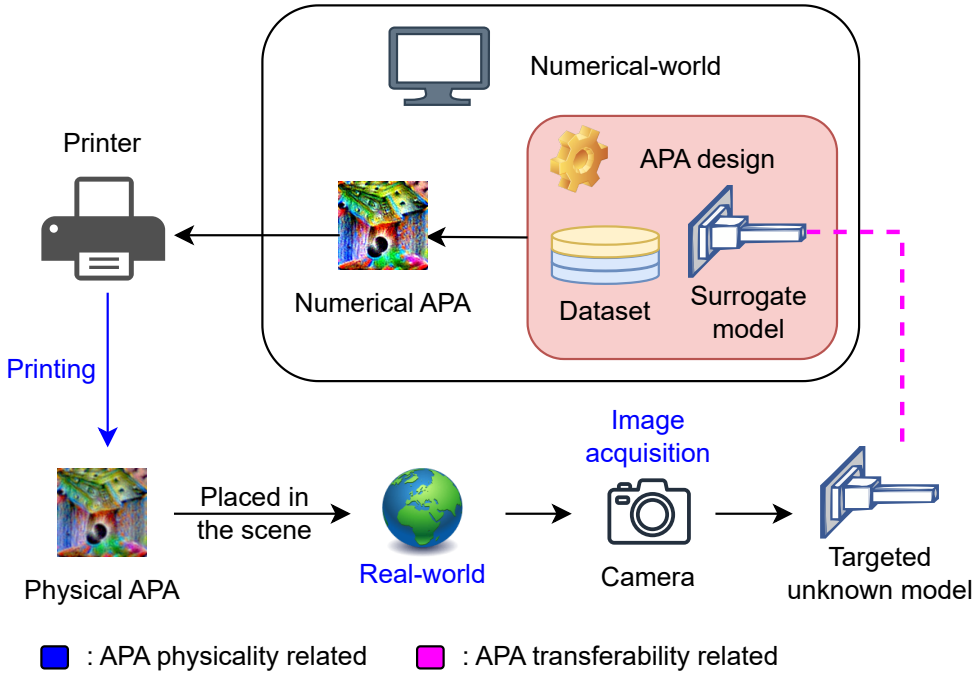


Figure 3.1: Diagram of APA's attack pipeline against a real-world system. In blue are represented transformations acting on APA's physicality and in magenta is represented the possible lack of knowledge about the targeted system.

ranging from printing to image acquisition, are collectively categorized as physical transformations (represented in blue in Fig. 3.1). The patch's robustness, according to physical transformations, is thus one of the main properties that define a critical patch. Robustness according to physical transformations can be resumed by the notion of *patch physicality*.

A real-world system, such as a computer vision network deployed in an autonomous vehicle, is usually targeted in a particular attack scenario. As these systems are designed and developed by companies, they usually maintain a high level of security, implying very restricted access to the system component. Thus, the attacker has no knowledge about the targeted system. Since we consider a real-world system, not an API, the attack can not be designed by querying the model. So, the attack must be designed in the black-box transfer-based attack scenario (represented in magenta in Fig. 3.1). The second expected property of a critical patch is *transferability* (see Figure 2.17 for a reminder).

We therefore define a *critical patch* to be physical and transferable. The following section details the numerical transformations and processes used to assess APA physicality and transferability.

3.2 Evaluation processes to assess APA criticality

To evaluate APA's criticality, we introduce a range of numerical transformations and evaluation processes. These tools assess independently patch physicality and transferability, providing insights into APA's limitations. To illustrate the applicability of our approach, we contextualize these evaluation tools by using the example of an APA designed to target the computer vision systems used in an autonomous vehicle.

Table 3.1: Evaluation settings by category and their brief description.

Property	Category	Transformation/Process	Example
Physicality	Radiometric	Varying weather conditions	Brightness, snow, rain, ...
		Filters	JPEG transformation
	Geometric	Rescaling	* * *
		Crop	* * *
		Affine transformations	Rotations
		Distance wrt. learning position	Shift from learning position
Transferability	Detector sensitivity	Learning paradigm	Supervised, contrastive, ...
		Training recipe	Data augmentations, optimization, ...
		Initialization	Different seeds
	Detector generalisation	Varying architectures	e.g. ResNet to ViT

3.2.1 APA physicality

The physicality of an adversarial patch refers to its robustness against the previously mentioned physical transformations: printing, real-world volatility, and image acquisition. To model these transformations, we introduce a set of numerical transformations that, when combined, provide a robust proxy for evaluating an APA's physicality. These numerical transformations can be categorized into radiometric and geometric transformations and are shown in the first row of Table 3.1.

Radiometric transformations. Radiometric transformations are transformations used to measure patch robustness against environmental changes like illumination and weather conditions and photometric changes like filters. They measure patch robustness when radiometric phenomena modify the entire image or the patch. Regarding our example, an attacker would design a patch resilient to the day's weather or illumination on the patch.

Geometric transformations. Geometric transformations are designed to capture the robustness of a patch when subject to geometric transformations. Contrary to radiometric features, geometric transformations concerns the patch itself and not the whole image. We can distinguish two types of geometric transformation. The transformations of the patch itself, such as the effect of a zoom or an ablation of one of the parts of the patch, and the transformations of the patch in its physical environments, such as affine transformations, rotations, and displacements with respect to its training position. Regarding our self-driving car attack example, an attack could be efficient regardless of its position in the image.

3.2.2 APA transferability

Transferability evaluation processes assess patch robustness based on changes to the neural network triptych defined in Chapter 2: network architecture, training strategy, and learning dataset. Regarding the above-mentioned autonomous vehicle example, we should be able to answer questions like: will an attack succeed on one YOLO (Redmon et al., 2016) capable of attacking any YOLO? Or will an attack fooling YOLO's detector be able to sway a Faster R-CNN (Ren et al., 2015)? If the attacker has the knowledge of the targeted architecture but not its weights, is it more effective to design the attack on another network from the same architecture or on a network from a different architecture?

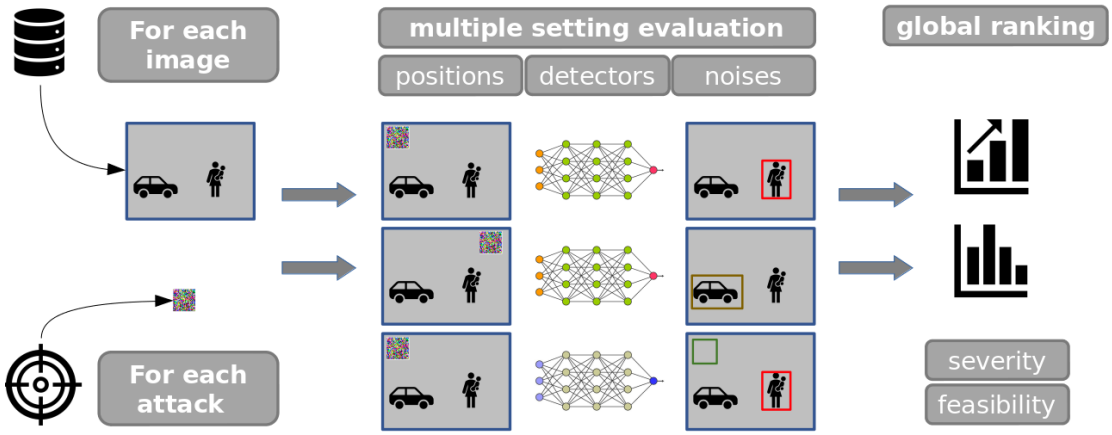


Figure 3.2: Structure of the proposed pipeline to evaluate APAs. Given a dataset, a network and a patch, we evaluate multiple settings. The resulting targeted success (tSuc) rate for image classification or the resulting average precision (AP) for object detection are used to rank APAs for each setting. The global rating measures the impact in real-world conditions of each APA.

Another point of view is to consider these transferability evaluation processes to measure the patch robustness according to several attacking knowledge scenarios. For example, attacking with the single knowledge of the targeted system architecture or attacking with this architecture's knowledge and training dataset knowledge.

From a defense perspective, measuring patch effectiveness against different network learning strategies help to understand the robustness of an architecture against APA. Similarly, a learning paradigm or a training recipe can be studied from a robustness perspective and help developers to design more robust systems.

Table 3.1 summarizes and describes all the settings related to a given property. Note that certain transformations can be applied only on the patch or the image as a whole for radiometric and geometric settings.

3.2.3 Evaluation pipeline

We suggest, in this thesis, a pipeline (see Figure 3.2) allowing the evaluation of the criticality of patch-based adversarial attacks. It is based on the evaluation processes detailed previously. The first step of the pipeline consists of selecting an attack strategy, a source model, and a dataset on which we may train the patch following the selected attack strategy. Once a patch is designed and learned, we evaluate its performance on the selected setting. The reported evaluation criteria are the targeted success rate (tSuc, defined in Eq. 2.17) for image classification or the mean average precision (mAP, defined in the beginning of Section 2.1.2) for object detection. Those metrics are computed before (clean) and after the application of the patch on the data (perturbed). APA's physicality and transferability are assessed by comparing attacking performance when the patch is placed in the same condition as during its training. When evaluating contextual effects, we ensure that no object of interest intersects with the patch and that false positive detections are removed on the patch if they are localized at the same position as the patch. This evaluation pipeline allows a better overview of the patch's criticality.

3.3 Beyond numerical APA

This section uses the previous evaluation pipeline to assess the criticality of three APA methodologies (Brown et al., 2017; Lee and Kolter, 2019; Saha et al., 2020). These methodologies are chosen because they are strong baselines. Rather than evaluating each transformation in each property, we design and evaluate patches regarding four of them (two per property). This study aims to provide a comprehensive overview of the current limitations of APAs that may reduce their criticality. The evaluated patches are designed to fool an image classifier to output a chosen class (targeted attack) or prevent an object from being detected (*person* class here). The APAs against object detectors are evaluated in the contextual and invisible cloak modes. We also compare the effect of the Expectation over Transformations (EoT) technique Athalye et al. (2018) on the resulting patch, an evaluation strategy explained in the following.

3.3.1 Expectation over Transformations technique

The aim of EoT technique (Athalye et al., 2018) is to make an attack more resilient to a chosen distribution of transformations (e.g., image rotations). To do so, the EoT technique consists of training the attack by sampling the set of transformations at each training iteration according to the considered distribution. For example, the two following equations detail an image classification patch training without and with patch translation in the image:

$$\begin{aligned} &\arg\min_{\delta} \mathbb{E}_{x\sim X}\left[\mathcal{L}_{\text{CE}}\left(M\odot\delta+(1-M)\odot x, y_{tgt}\right)\right] \quad \text{(fixed patch training)},\\ &\arg\min_{\delta} \mathbb{E}_{x\sim X, e\sim E}\left[\mathcal{L}_{\text{CE}}\left(F_\theta(A(\delta,x,e)), y_{tgt}\right)\right] \quad \text{(training w/ translations)}, \end{aligned} \tag{3.1}$$

where M is a fixed binary mask defining the patch position in the image, A is the patch applicator operator defined in Eq. 2.18 (see also Fig. 2.18), e is the patch location in the image x and δ is the actual patch. EoT for patch training usually considers the following transformations: rotation, homography, translation, resizing, random crop, brightness and contrast changes, and Gaussian noise, applied either on the patch or on the whole image. These transformations depend on fixed ranges of hyperparameters, e.g., patch x-axis rotation from -5°to +5°.

3.3.2 Experimental setup

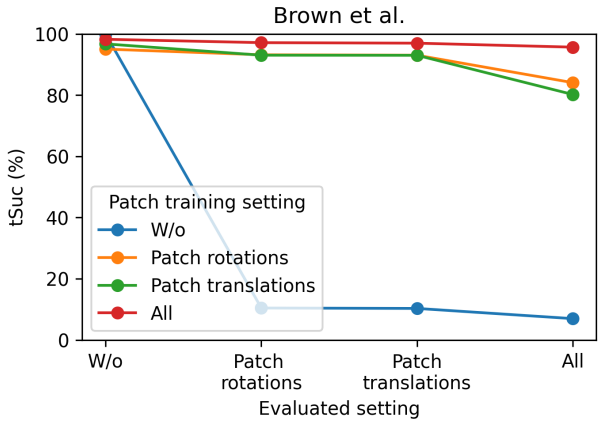
Once learned, the obtained patches are evaluated in four settings: patch shift from learning position, patch rotations, various model initializations and architectures. We evaluate the GAP attack for image classification (Brown et al., 2017), and attacks from Lee and Kolter (2019) and Saha et al. (2020) for object detection. We solve each attack's optimization problem and clip the patch to $[0, 1]$ (images are normalized). Clipping the patch ensures that patch values remain in the image range and do not produce *NaN* values. Patch size is set to 10% of the image size. We launch the optimization process with an all-zeros patch and apply the optimizer used in the corresponding article. The patch optimization is run over 100 epochs (1 epoch equals 1000 iterations, as in Lee and Kolter (2019)) with a batch size of 50 images for image classification and 1 for object detection (as in Lee and Kolter (2019) and Saha et al. (2020)). Optimizations are launched for three different learning rates, and finally we select the APA with the minimum loss for evaluation.

To study the effect of the EoT method (Athalye et al., 2018) and consequently the resulting patch physicality, we optimize patches with different transformations during patch training: no transformation (named *W/o* in the following), patch rotations without patch translation (named *Patch rotations*), patch translations in the image without patch rotation (named *Patch translations*), and combinations of translations and rotations (named *All* in the following) as described in Section 3.3.1. When training the patch without patch translation, we follow Saha et al. (2020) and place the patch at the top-left corner, $i.e.$, pixel $(5, 5)$ in the image. On the other hand, we randomly placed the patch in the image for *Patch translations* and *All* training scenarios. During patch evaluation, we measure patch robustness when facing different scenarios. For example, when evaluated in the *All* scenario, the corresponding transformations (rotations, Gaussian noise ...) are applied to the patch. However, the patch is no longer translated into the image during evaluation. It is either evaluated at the top-left corner for the *W/o* and *Patch rotations* transformations or at the bottom-right corner for the *Patch translations* and *All* transformations. To study the transferability property, we consider the single-source model evaluation process (patches are designed using a single model) and test attack transferability to other models.

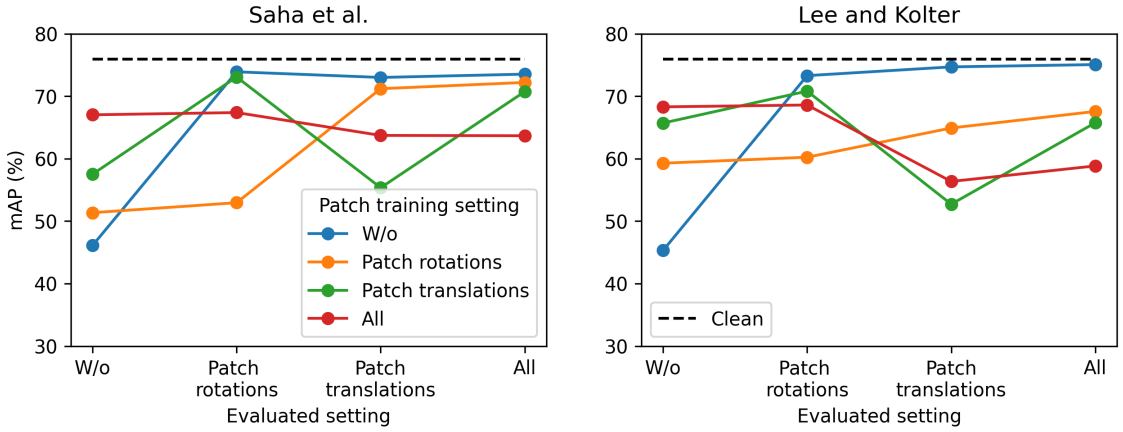
Image classification. To apply these evaluation process for attacking image classification, we use the ImageNet-1K (Deng et al., 2009) dataset and the following models: ResNet50, ResNet50-v2 (He et al., 2016), Swin-T (Liu et al., 2021) from Pytorch Model Zoo[1] and DeiT-T (a ViT trained using distillation) (Touvron et al., 2021) from Timm Model Zoo[2]. We randomly selected three target classes (eft, birdhouse, and starfish) and generated a patch for each class. Patches are designed to mislead the network into incorrectly classifying inputs as the respective target class. We split the ImageNet-1K validation set into a training set of 40000 images on which we train patches and a test set of 10000 images on which we evaluate their impact. Reported tSuc are averaged over the classes.

Object detection. PASCAL VOC (Everingham et al., 2008) dataset is used to apply the evaluated settings for patch attack against object detection. We evaluate the Lee and Kolter (2019) attack and the blindness attack from Saha et al. (2020). We recall that the attack from Lee and Kolter (2019) involves designing a patch to maximize the detector loss over the ground truth. In contrast, the blindness attack from Saha et al. (2020) creates a patch that prevents a class from being detected by reducing the probability of the predicted boxes that match the ground truth. We consider the blindness attack against the person class. We evaluate patches against two YOLOv2 models obtained using the same learning strategy but with a different weight initialization. As explained in Section 3.2.3, we extract images wherein targeted objects do not overlap with the patch when designing and evaluating contextual patches. These images are extracted from PASCAL VOC (Everingham et al., 2008) and split the set into two parts, one for training of the patch and the other for patch evaluation. The Lee and Kolter (2019) attack and the blindness attack from Saha et al. (2020) are designed, evaluated, and compared when designing contextual patches. We design the invisible cloak mode patch using the blindness attack from Saha et al. (2020). When training invisible cloak APAs, patches are scaled and placed at the center of targeted class boxes. Patches scale are defined with respect to the patch scale ratio which is the length ratio between the patch and the corresponding bounding box in

[1]https://pytorch.org/vision/stable/models.html
[2]https://timm.fast.ai/



(a) Robustness curves for image classification patches. Patches are designed using the Brown et al. (2017) attack. Results are averaged over three targeted classes. A higher tSuc indicates a more effective attack.



(b) Robustness curves for contextual adversarial patches targeting object detectors. Patches are either designed using the blindness attack (Saha et al., 2020) to hide a person from being detected or using the attack from Lee and Kolter (2019). The dashed black line represents YOLOv2's clean performance. A lower mAP indicates a more effective attack

Figure 3.3: Robustness curves of patch physicality across various evaluation scenarios. The patches are trained under different physical transformations, and their attack effectiveness is assessed across multiple physical evaluation settings. These scenarios include: (1) no transformations (*W/o*), (2) rotations applied to the patch without translations (*Patch rotations*), (3) translations applied to the patch within the image without rotations *Patch translations*), and (4) the full set of EoT transformations (*All*). These curves illustrate the impact of each transformation on patch attack robustness.

which the patch is placed. Following standards (Thys et al., 2019; Huang et al., 2023), we set the patch scale ratio at 0.15 during patch training and evaluation. In evaluation mode, we set the YOLOv2 confidence threshold at $5e-4$, the non-maximum suppression at 0.45, and the IOU at 0.5.

3.3.3 Results

Patch physicality. Figure 3.3 gathers the results of the evaluation processes used to assess patch physicality. Training a patch, without considering any transformation, results in a highly effective attack when evaluated with the same setting as the one applied for training. However, patch effectiveness drops sharply when the patch is translated in the image (even by a few pixels), rotated, or modified by Gaussian noise, making the attack ineffective in physical conditions (blue line in the three plots). When targeting an object detector, designing a patch without rotation results in lower attack performance when the patch is translated in the image and vice versa. As expected, training patches using the full range of EoT transformations (*All* training setting, red lines) leads to more robust patches when faced with the broadest set of transformations (*All* evaluation setting). Qualitatively, the *W/o* setting generates pixelated patches while the *All* setting generates smoother patches that are more physically realizable (see Fig. B.1 in the Appendix).

Patch transferability. Table 3.2 summarizes the transferability results for APAs designed for image classification task, contextual patches, and invisible cloak patches. Without using the EoT method, the generated patches for image classification do not transfer across architectures or ResNets. When trained with transformations, patches trained on the first version of ResNet50 (named R50 in the table) exhibit a small capacity to transfer to ResNet50-v2 and vice versa. One possible explanation is that EoT, as a patch regularization, reduces the patch overfitting on the source model, thus enhancing a little patch transferability. Nonetheless, even when using EoT, none of the patches show a large scale transferability across architectures. The low transferability capacity is also observed for contextual and invisible cloak patches, although patches are transferred to a model from the same architecture. We observe that the drop in the mAP or the person AP is due to the saliency of the patch producing false positives near the patch (see Fig. 3.4). The black-box model still succeeds in detecting and correctly classifying objects of interest. These results highlight that patches fail to transfer between models, limiting their potential criticality.

To conclude, we would like to show in this section that training a patch without considering any physical transformations results in a patch that is not robust to physical transformations. The EoT method (Athalye et al., 2018) can improve patch robustness according to physical transformations. In spite of this, no patch designed by the considered approaches demonstrates a capacity to transfer between models.

Table 3.2: Transfer results between models for APAs against image classification and object detection (contextual patches and invisible cloak patches). Transferability is evaluated for two patch training settings: no transformations (*W/o*) and the full set of EoT transformations (*All*). For image classification, results are averaged over classes and are measured by the targeted success rate (tSuc (%), a higher value indicates a better attack). Control is considering a sample of the dataset corresponding to the target class as the patch. For contextual patches, false positives on the patch are removed, and results are measured by mAP ((%), a lower value indicates a better attack). For the invisible cloak patch, patches are resized and placed in the middle of targeted objects (here, person detection). Results are measured by the person AP ((%), a lower value indicates a better attack).

Image classification (tSuc %)						
Attack	Patch training setting	Target / Source	R50	R50-v2	DeiT-T	Swin-T
Control			2.85	1.59	1.57	0.93
GAP (Brown et al., 2017)	W/o	R50	100	0.5	0.	0.1
		R50-v2	0.1	100	0.1	0.1
		Swin-T	0.1	0.1	0.1	100
	All	R50	96	12	0.2	3
		R50-v2	10.5	78	0.2	2
		Swin-T	1	2	0.1	31

Contextual patches (mAP %, without f.p.)				
		Target / Source	YOLOv2-1	YOLOv2-2
Clean			75.9	80.52
Saha et al. (2020)	W/o	YOLOv2-1	46.1	76.75
	All		63.67	75.22
Lee and Kolter (2019)	W/o	YOLOv2-1	45.31	78.13
	All		58.84	75.14

Invisible cloak (person AP %)				
		Target / Source	YOLOv2-1	YOLOv2-2
Clean			78.64	80.07
Saha et al. (2020)	All	YOLOv2-1	22.27	63.15

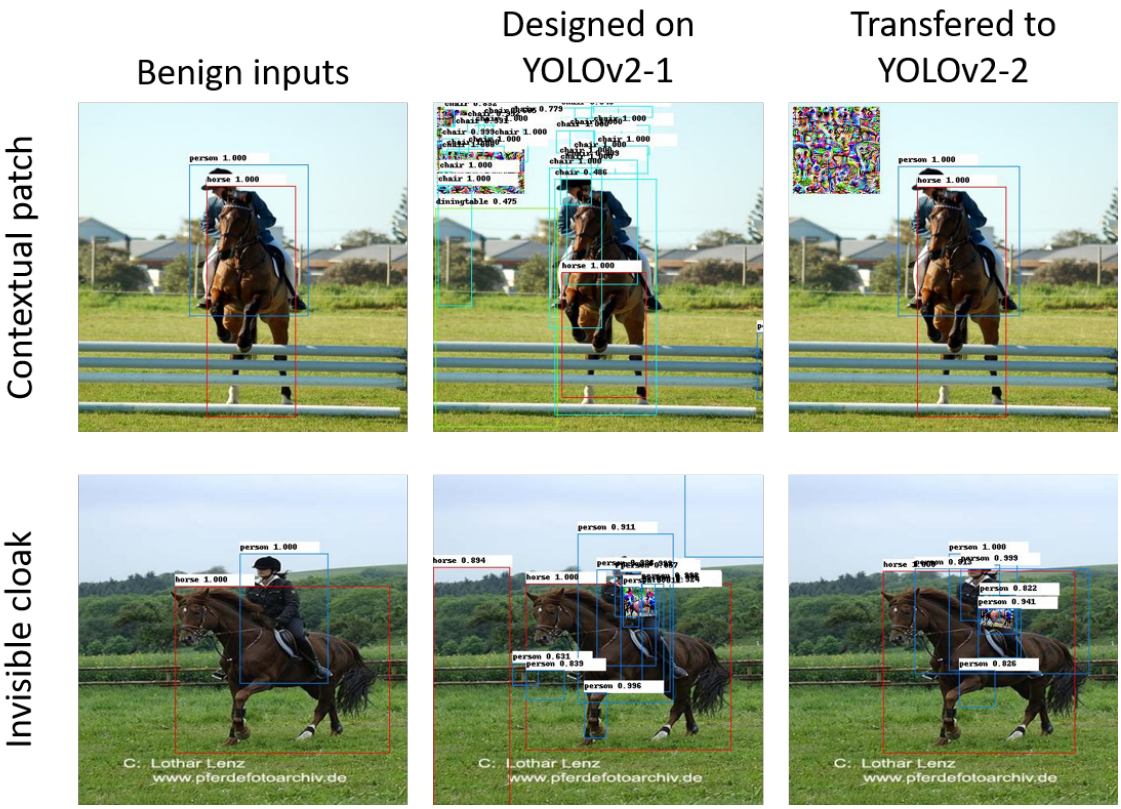


Figure 3.4: Example of white-box and black-box transfer scenarios for the contextual patch and the invisible cloak attack. Attacks are designed on a first YOLOv2 model (YOLOv2-1) and transferred to YOLOv2-2, the same model trained using the same learning strategy, but with weights initialized from a different seed. Patches are designed to prevent persons from being detected.

(a) Patch physicality is enhanced by using the EoT technique from (Athalye et al., 2018), the TV loss, and the non-printability loss. Source: Thys et al. (2019)

(b) Patch physicality is enhanced by using the EoT technique from (Athalye et al., 2018), the TV loss, and a Thin Plate Spline (TPS) based method. Source: Xu et al. (2020)

(c) Patch physicality is enhanced by using the EoT technique from (Athalye et al., 2018), the TV loss and the Toroidal-Cropping-based Expandable Generative Attack. Source: Hu et al. (2022)

Figure 3.5: Examples of patches causing the disappearance of people when applied on them. The first two patches are used as the central pattern in a T-shirt, while the last is a dress.

3.4 State-of-the-art APAs

In this section, we review the different techniques used to enhance the physicality and the transferability of patches. As explained in the previous section, patch physicality is often increased using the Expectation over Transformations (EoT) technique from Athalye et al. (2018). The Total Variation (TV) loss is also used to enhance patch physicality further:

$$TV(x) = \sum_{i,j} |x_{i+1,j} - x_{i,j}| + |x_{i,j+1} - x_{i,j}|, \tag{3.2}$$

where i and j are the height and width indices in the image, respectively. This loss encourages the patch to be smoother, i.e., and neighbor pixel values to stay close. A very pixelated patch is less successful in physical conditions due to the physics of the image acquisition (diffraction). The EoT technique and TV loss form the baseline toolkit to generate physical APA. Some works (Sharif et al., 2016; Thys et al., 2019) use another loss, called the non-printability loss, to represent how well a common printer can represent the colors in the patch. However, this loss is not useful when combined with the TV loss (Nesti et al., 2022).

Recent works have further improved the physicality of patches, especially in scenarios where the patch is a T-shirt worn by a person (Thys et al., 2019). This problem is challenging because T-shirts are non-rigid objects, while stop signs, eyeglasses, and cardboard are. Xu et al. (2020) develop a Thin Plate Spline (TPS) based method to model the deformation of a t-shirt for a moving person (e.g., wrinkles). However, the adversarial pattern is placed in the middle of the front side of T-shirts, limiting its criticality for different camera viewing angles. To generate a multi-angle patch attack, Hu et al. (2022) introduce Toroidal-Cropping-based Expandable Generative Attack (TC-EGA), a generative method developed to craft a patch with repetitive structures. This method involves finding, cropping, and repeating the best adversarial pattern to create a texture independent of the camera viewing angle. They use their method to generate several pieces of cloth, e.g.,

(a) Example of the flower attack (TnT method). The latent representation of the generator is modified to generate an adversarial flower that deceives the image classifier to output a pre-chosen class. Source: Doan et al. (2022)

(b) Example of a people disappearance attack. The patch is generated by applying small modifications to a benign image. Source: Hu et al. (2021)

(c) Example of a people disappearance attack. The latent representation of the generator is modified to generate an adversarial dog that deceives the object detector in detecting persons. Source: Huang et al. (2020)

Figure 3.6: Examples of naturalist-oriented patches.

T-shirts, skirts, and dresses, and show effectiveness in the real world for several physical conditions. Figure 3.5 shows examples of patches generated by the previously explained methods.

Another line of work focuses on designing more naturalistic and inconspicuous patches to prevent humans from detecting them. Liu et al. (2019) (PS-GAN method) train a GAN to generate a background-harmonious patch that enhances both the patch's visual fidelity and attacking ability. Instead of training a GAN, Doan et al. (2022) and Casper et al. (2022) use a pre-trained GAN directly. To change the generated flower to an adversarial flower, Doan et al. (2022) (TnT method) modify the latent representation of the generator. The same procedure is used by Hu et al. (2021) to generate naturalistic adversarial dogs to fool an object detector. Casper et al. (2022) perturb the latent representation at some chosen generator layer. The generated patches are used as interpretability tools to inspect network decision mechanisms. Instead of using a GAN, Huang et al. (2020) initialize the patch as a natural image and apply small modifications to make the image adversarial. Figure 3.6 shows examples of naturalistic-oriented patches generated by the previously explained methods. However, these natural-looking patches show limitations. A dog-head-like patch on a T-shirt's front is natural-looking but not robust to multiple camera viewing angles. On the other side, repeating the dog everywhere on the t-shirt increases the robustness of the patch according to multiple camera angles but makes the T-shirt look weird, limiting its inconspicuousness. To create a more natural-looking patch that is robust to multiple camera viewing angles, Hu et al. (2023) introduce a novel 3D augmentation method combining topologically plausible projection (TopoProj) and thin plate spline (TPS). They generate more natural-looking camouflage clothes that look like military camouflage equipment. Currently, this method is state-of-the-art in generating physical patches.

Whereas the physicality of patches has been explored and improved in previous works, patch transferability remains less studied. Some of the above papers contain a section discussing the transferability of their produced patches, and they usually show that their

patches transfer poorly. To increase patch transferability, Huang et al. (2023) introduce a set of techniques and optimization tricks: they adjust the patch learning rate and add a scheduler. They propose to use the Shakedrop technique (Yamada et al., 2018) to generate variants of the source model during patch training; they randomly crop the patch to avoid the attack performance relying on a particular region of the patch. These techniques prevent the patch from overfitting on the source model. Nevertheless, in Section 5, we show that a misusage of the mAP metric causes a biased account of transferability.

3.5 Conclusion

This chapter defines a critical patch as being physically realizable and transferable. We developed an evaluation pipeline with various numerical transformations and evaluation processes, allowing us to assess patch criticality. We demonstrated that a trained patch may not be physically feasible without considering specific training techniques. To improve physicality, we showed that the EoT method and TV loss can be effective baselines for generating physical patches. While several approaches have been proposed to enhance patch physicality further, less attention has been given to improving transferability. We found that current patches do not effectively transfer across models.

The following chapters are dedicated to enhancing the transferability of APA. To ensure that the generated patches remain physically realizable, we will employ the EoT method and TV loss. Chapter 4 focuses on improving patch transferability across image classifiers and introduces a novel strategy. In Chapter 5, we examine the transferability of APAs across object detectors, further expanding the scope of this analysis.

Chapter 4

A transferable targeted patch on image classification

In Chapter 3, we show that despite the good attacking performance of current APA in white-box configuration (applied on the same model that they have been learned), their effectiveness is mitigated when transferring to unseen models, i.e. in black-box configuration (see Paragraph 2.2.2 for a reminder about white-box vs black-box). In this chapter, we move towards building more transferable APAs. With a focus on image classification, we first review the different APAs proposed in the literature and their transferability. Then, we present several methods used to improve the transferability of adversarial noise, adapt these methods to build APAs, and study their limitations. We propose a new optimal transport-based method to improve the transferability of APAs for classification. Through numerical, hybrid, and physical experiments, we show the superiority of our method. Finally, we study the perturbation mechanism of our method and other APAs. This chapter is based on our contribution Labarbarie et al. (2024) published at the ICLR 2024 conference.

4.1 APA transferability

We consider the standard notation where $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, ..., n$, are samples drawn from a joint distribution of random variables X and Y. As we are considering the problem of an APA against image classifiers, the input is sampled from $\mathcal{X} = \mathbb{R}^{h \times w \times c}$, where $h \times w$ are the image dimensions and c is the number of channels and the output is sampled from a set of M labels $\mathcal{Y} = \{y_1, ..., y_M\}$. Let $\mathcal{T}$ be a distribution over transformation (e.g., rotations, scaling, blur, ...), E a distribution over locations and $A(.,.,.,.)$ the patch applicator operator (see Fig. 2.18 for a reminder). Our goal is to create an APA that perturbs the process of classifying any image x_i as the target class y_{tgt} by any classifier, $F : \mathcal{X} \to \mathcal{P}(\mathcal{Y})$, that has been previously trained to do so. This goal can be mathematically formalized as follows:

$$\arg\max_{\delta} \mathbb{E}_{F\sim\mathcal{F},x\sim X,t\sim\mathcal{T},e\sim E} \left[\mathbb{1}_{\arg\max_m F(A(\delta,x,e,t))_m = y_{tgt}}\right], \tag{4.1}$$

where $\mathcal{F}$ denotes the space of neural networks that we want to fool. Sampling according to $\mathcal{T}$ and E enhance patch physicality by using the EoT method (see Section 3.3.1 for a reminder), sampling according to X increases patch universality (see Paragraph 2.14 for a reminder). Sampling according to $\mathcal{F}$ is expected to increase patch transferability.

However, estimating this expectation over $\mathcal{F}$ is intractable in practice: we have only few realizations of this distribution. Moreover, this distribution is very diverse, encompassing a variety of network architectures, learning strategies (such as learning paradigms and training methodologies), and training datasets (see the beginning of Chapter 2 for a reminder on the triptych that characterizes a network). In Section 4.4.1, we show that utilizing the few available realizations to form an ensemble of models for APA design is computationally expensive and may lead to suboptimal attacking results. The objective of transferability can be redefined as a search for a strategy that, when employed to learn an APA on a single source model, results in an APA that exhibits an impact on many other networks.

The following section provides an overview of the transferable attacks designed to fool image classifiers.

4.2 Low transferability of current APAs

This section describes the various APAs proposed in the literature, focusing on their transferability capabilities. We highlight the limitations of the design strategy used to design these APAs. While methods to improve APA transferability have received relatively little attention, the transferability of adversarial noise has been more extensively studied. We then review the existing approaches for enhancing adversarial noise transferability and discuss their limitations.

4.2.1 APA for image classification

APAs for image classification were first introduced by Brown et al. (2017). They design a patch (called GAP) by minimizing, under Expectation over Transformations (EoT) (Athalye et al., 2018), the cross-entropy loss of a selected target class. To increase the fooling effectiveness of the patch, Karmon et al. (2018) (LaVAN method) add a new term to the loss criterion initially proposed by Brown et al. (2017) as:

$$\arg\min_{\delta} \mathbb{E}_{x\sim X, t\sim\mathcal{T}, e\sim E}\Big[\underbrace{\mathcal{L}_{\text{CE}}\left(F_\theta(A(\delta,x,e,t)),y_{tgt}\right)}_{\text{Brown et al. (2017)}} - \underbrace{\mathcal{L}_{\text{CE}}\left(F_\theta(A(\delta,x,e,t)),y_{src}\right)}_{\text{added term}}\Big], \quad (4.2)$$

where y_{src} can be either the image ground truth to ensure the image miss-classification or a chosen target to sway the network to not predict this label. Other methods propose to use generative-based approaches: Liu et al. (2019) (PS-GAN method) train a GAN to generate a background-harmonious patch that enhances both the visual fidelity and attacking ability of the patch. Instead of training a GAN, Doan et al. (2022) and Casper et al. (2022) use a pre-trained GAN directly. To change the generated object (flower in their work) to an adversarial object, Doan et al. (2022) (named TnT method in the following) modify the latent representation of the generator:

$$\begin{aligned}\arg\min_{z} \mathbb{E}_{x\sim X, t\sim\mathcal{T}, e\sim E}\Big[&\mathcal{L}_{\text{CE}}\left(F_\theta(A(G_{\theta'}(z),x,e,t)),y_{tgt}\right) - \\ &\mathcal{L}_{\text{CE}}\left(F_\theta(A(G_{\theta'}(z),x,e,t)),y_{src}\right)\Big],\end{aligned} \quad (4.3)$$

where z is the latent representation and $G_{\theta'}$ is the pre-trained GAN. Casper et al. (2022) perturb the latent representation at some chosen generator layer.

Table 4.1: Transfer results when changing the linear classifier while the encoder remains fixed (variation of tSuc (%)). Patches are designed to fool the Swin-T model (encoder, and linear classifier from Pytorch Model Zoo). The transferability is measured when targeting a new network (same encoder, different linear classifier). Results are averaged over classes.

Method	Variation of tSuc (%)
GAP (Brown et al., 2017)	- 61.4
LaVAN (Karmon et al., 2018)	- 42.6

Evaluation of transferability. The previously presented APAs Brown et al. (2017); Karmon et al. (2018); Liu et al. (2019); Doan et al. (2022); Casper et al. (2022) use different evaluation settings in their experimentation. Some of these works measure the transferability of their patch to unknown models. Brown et al. (2017) and Doan et al. (2022) measure their patch transferability but only consider dated CNNs like ResNet trained with a less generalizing learning policy (ResNet50-v1) than recent ones (ResNet50-v2). The PS-GAN generated patch (Liu et al., 2019) shows transferability among unseen models. However, the evaluated models are not state-of-the-art, and the experiments are conducted on the small GTRSB dataset (Stallkamp et al., 2011). Focused on the interpretability of decision-making of deep networks, Casper et al. (2022) do not provide quantitative results on network transferability.

Limitations of decision boundary-based methods. The previous works on APA for classification employed the APA learning strategy of influencing the network to output a target class with high confidence. (Brown et al., 2017; Karmon et al., 2018; Liu et al., 2019; Doan et al., 2022; Casper et al., 2022). This strategy involves pushing the deep representation of images to cross the nearest decision boundary of the source model. Such a strategy has two drawbacks: it is highly dependent on the model on which the attack is based, and the patch may push the corrupted image representations into unseen regions of the representation space. To illustrate that decision boundary-based methods learn a patch that tends to overfit on the source model classifier, we consider the transfer not between 2 different models but between 2 models sharing the same encoder but different classifiers. We design APAs using Brown et al. (2017) (GAP) and Karmon et al. (2018) (LaVAN) methodologies and following the same optimization procedure as in Section 3.3.2 (patches targeted three classes, trained using EoT, best patches reaching the minimum loss among three learning rates). Patches are trained to attack the source model Swin-T (Liu et al., 2021). On top of this Swin-T encoder, we train a new linear classifier from scratch on the ImageNet train set (Deng et al., 2009). This new linear classifier reaches the same level of clean accuracy as the previous classifier (from Pytorch (Paszke et al., 2019)) while being different. We measured the patch performance when targeting this new network (same encoder, different linear classifier). As expected, the transferability of decision boundary-based patches drops drastically (nearly by half) (see Tab. 4.1). This poor APA transferability does not result from a bad APA learning since the white-box patch's tSuc is nearly 100%.

While methods to improve APA transferability have received little attention, various methods have been introduced for adversarial noises to prevent attack overfitting on the source model classifier.

Table 4.2: White-box and transfer results between ResNet50-V1 and Swin-T (tSuc (%)) for the L2 method (Inkawhich et al., 2019). Three different target points are evaluated. Results are for the source model ResNet50-V1 and Swin-T, for the target class *australian terrier* and for patches of size 60×60. Patches are placed randomly in the image but not at the center of the images.

		Targeted network	
Source		ResNet50-v1	Swin-T
ResNet50-v1	Target point 1	0.4	0.1
	Target point 2	57.6	0.45
	Target point 3	40.4	0.9
Swin-T	Target point 1	0.51	1.2
	Target point 2	9.81	12.79
	Target point 3	0.1	0.1

4.2.2 Transferable adversarial noises

Methods enhancing adversarial noises transferability can be grouped into the following categories: data augmentation-based techniques, optimization-based techniques and loss-based techniques which are described briefly in the following. The reader can refer to Gu et al. (2023) for a complete survey.

Data augmentations-based. Data augmentations-based techniques consist of applying stochastic input transformations during adversarial noise optimization. For example, Xie et al. (2019) generate several versions of the targeted input x using random data augmentations and design the adversarial noise to deceive the transformed inputs. While these data augmentations-based techniques are useful to enhance the transferability of instance-specific attacks (see Fig. 2.14 for a reminder), their usefulness is limited when designing universal attacks like APAs. The universal perturbation designed by iterating under the expectation over samples is already a form of data augmentation regularization.

Optimization-based. Optimization-based techniques enhance adversarial noise transferability by refining optimization methods. For instance, Dong et al. (2018) introduce the momentum iterative fast gradient sign method (MI-FGSM), which improves transferability of I-FGSM attacks by adding a momentum term to the input gradient. While these methods boost transferability, we focus on loss-based techniques to address decision boundary issues by design.

Loss-based. Instead of blindly maximizing the probability of a target class, research on adversarial noises suggests considering the feature space of deep networks (Rozsa et al., 2017; Zhou et al., 2018; Inkawhich et al., 2019; Lu et al., 2020; Inkawhich et al., 2020). For example, Inkawhich et al. (2019) propose optimizing the adversarial noises to match the feature representations of an existing target image:

$$\arg\min_{\delta} \mathbb{E}_{x \sim X} \left[||f(x+\delta) - f(x_i)||_2^2 \right], \tag{4.4}$$

where f is the encoder part of F_θ. The role of the attack is to push the corrupted image features $f(x+\delta)$ to be close to the features of $f(x_i)$ (feature representation of a chosen

Table 4.3: Typology of Adversarial Patch Attacks (APA) papers and transferable adversarial noises works depending on the requirement to illustrate a transferable physical APA. Each column represents an essential characteristic to demonstrate the real-world criticality of an APA. The symbols ✓, ≈, and ∅ represent "measured," "ambiguous," and "not evaluated," respectively.

	Transfer		Defended networks	APA
	Narrow	Broad		
GAP (Brown et al., 2017)	✓	∅	∅	✓
LaVAN (Karmon et al., 2018)	∅	∅	∅	✓
PS-GAN (Liu et al., 2019)	≈	∅	∅	✓
TnT (Doan et al., 2022)	✓	∅	✓	✓
Casper et al. (2022)	≈	∅	≈	✓
Inkawhich et al. (2019)	✓	∅	✓	∅
TTP Naseer et al. (2021)	✓	∅	✓	∅
M3D Zhao et al. (2023)	✓	∅	✓	∅
Ours	✓	✓	✓	✓

sample corresponding to a pre-selected class) and make $f(x+\delta)$ to be classified as y_i. The rationale for such a strategy is that the feature space is expected to capture the useful information in an image more universally, allowing better attack transferability between models.

However, this point-wise strategy (named L2 in the following) presents several drawbacks. First, optimizing when the objective is to push multiple points to a unique point is likely to fail. When the optimization succeeds, the power of the attack depends highly on the choice of the target point. Furthermore, a single feature point can be well-classified by one network but misclassified by another, thus limiting the transferability of the attack to multiple networks. To measure the stability of the L2 method (Inkawhich et al., 2019), we launch APA's optimization for three randomly selected target points. Patches are designed to sway ResNet50-v1 or Swin-T which will output the *Australian terrier* class. Patches are trained using EoT, and we select the patch that obtained the lower loss among the three learning rates. Table 4.2 reports the transfer results of the obtained patches. Although the optimization has converged for the first target of the L2 method for ResNet50-v1, the obtained patch is harmless. Even if the APA works, its attacking capacity depends on the considered target point. For example, on Swin-T, APA's transferability can decrease drastically. Additional experiments on the feature point method instability are provided in Appendix C.2.

Rather than independently train each adversarial noise by iterative methods, recent methods (Poursaeed et al., 2018; Naseer et al., 2019a, 2021; Zhao et al., 2023) train a GAN to transform clean images to adversarial noises. Naseer et al. (2021) (called TTP method) learn their GAN by minimizing the Kullback-Leibler divergence between the probability-class distribution of adversarial noises and the probability-class distribution of target class images. From a generalization error bound for black-box targeted attacks, Zhao et al. (2023) derive an algorithm to train their GAN that minimizes the maximum model discrepancy (M3D method) between two models. However, all these previously presented works study only the transferability of adversarial noises, and it is not obvious that they work for APA. Moreover, these methods rely on the classifier decision boundary, and we show that such rationale may result in poor transferable patches.

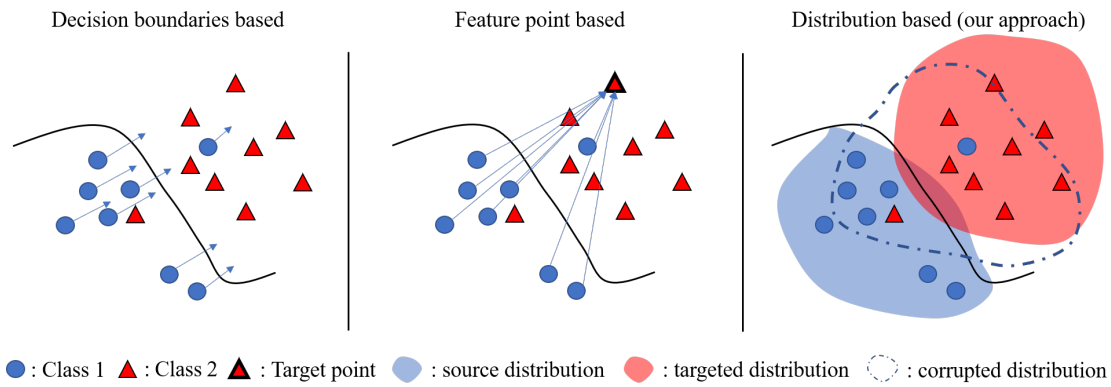


Figure 4.1: Three different strategies for designing patch attacks. Left: The attack pushes multiple samples to the other side of a decision boundary defined for a particular model. Middle: The attack matches a given point in the feature space that is expected to represent a sample from a different class. Right: Our strategy narrows the distribution gap between the samples corrupted by the patch and another misleading distribution in feature space. It does not depend on the decision boundaries nor on the choice of a specific target point in the feature space.

In the following section, we introduce a new method that follows the principle that the attack is designed with an optimization metric defined in the feature space rather than in the decision space (Inkawhich et al., 2019). This new method generalizes the L2 method proposed by Inkawhich et al. (2019). Our new method bridges the gap between the transferability approaches for adversarial noise design and adversarial patch attacks (see Tab. 4.3).

4.3 Feature-based APA learning for transferability

To overcome the dependence of the attack on a single decision boundary of the source model (left scheme in Figure 4.1) and to relax the specificity of the target feature point selection (middle scheme in Figure 4.1), we propose a distribution-oriented approach (right scheme in Figure 4.1). The learning principle of our patch attack is to globally alter the feature distribution of a set of images from a particular class to match another known distribution to be taken from another class. The role of the patch, when placed in the scene, is to push the feature distribution towards this known misleading class distribution (right scheme in Figure 4.1). Such a global strategy in feature space is expected to allow a better transferability capability, as it is independent of the decision boundary constructed by the classifier and the choice of the target point (as proposed by Inkawhich et al. (2019)). The technique used to push the feature distribution towards the target distribution is based on Optimal Transport (OT). The theoretical foundation of this technique and how it is adapted to our problem is detailed in the next section.

4.3.1 Background on optimal transport

The theory of optimal transport (Peyré et al., 2019; Villani et al., 2009) provides several techniques for efficient computation of distances between distributions. It has been shown that optimizing with respect to the Wasserstein distance has various practical benefits

over the KL-divergence (Peyré et al., 2012; Frogner et al., 2015; Arjovsky et al., 2017; Gulrajani et al., 2017). Unlike the KL-divergence and its related dissimilarity measures (e.g. Jensen-Shannon divergence), the Wasserstein distance can provide a meaningful notion of closeness between distributions supported on non-overlapping low dimensional manifolds.

Let $\mathcal{P}_p(\mathbb{R}^d) = \{\mu \in \mathcal{P}(\mathbb{R}^d) : \int_{\mathbb{R}_d} ||x||^p \mathrm{d}\mu(x) < \infty\}$ be the set of probability measures on $\mathbb{R}^d$ with finite moment of order p, with $p \in [1, +\infty)$. The p-Wasserstein distance is defined as

$$\mathbf{W}_p(\mu, \nu)^p = \inf_{\pi \in \Pi(\mu,\nu)} \int_{\mathbb{R}^d \times \mathbb{R}^d} ||x - y||^p \mathrm{d}\pi(x, y), \tag{4.5}$$

where $\mu, \nu \in \mathcal{P}_p(\mathbb{R}^d)$, $||.||$ is the Euclidean norm and $\Pi(\mu, \nu)$ is the set of probability measures on $\mathbb{R}^d \times \mathbb{R}^d$ whose marginals with respect to the first and second variables are given by μ and ν respectively. The quantity $\mathbf{W}_p(\mu, \nu)$ in not analytically avaible in general. To solve Eq 4.5, the standard methods are linear programs and have a worst-case computational complexity in $\mathcal{O}(n^3 \log(n))$, where n is the number of samples (Peyré et al., 2019).

To leverage the computational efficiency of Eq 4.5, Rabin et al. (2012) and Bonneel et al. (2015) works define a new metric named Sliced-Wasserstein distance. This new metric is based on the fact that for one-dimensional probability measure, the p-Wasserstein distance (4.5) has the following closed-form

$$\mathbf{W}_p(\mu, \nu)^p = \int_0^1 |Q_\mu(s) - Q_\nu(s)|^p \mathrm{d}s, \tag{4.6}$$

where Q_μ and Q_ν are the quantile functions of μ and ν respectively. Let $\mathbb{S}^{d-1}$ be the d-dimensional unit sphere and σ the uniform distribution on $\mathbb{S}^{d-1}$. For $\theta \in \mathbb{S}^{d-1}$, we define the linear form for all $x \in \mathbb{R}^d$ by $\theta^*(x) = \langle \theta, x \rangle$. The Sliced-Wasserstein distance is then defined by

$$\mathbf{SW}_p(\mu, \nu)^p = \int_{\mathbb{S}^{d-1}} \mathbf{W}_p^p(\theta^*_\sharp \mu, \theta^*_\sharp \nu) \mathrm{d}\sigma(\theta), \tag{4.7}$$

where $\mu, \nu \in \mathcal{P}_p(\mathbb{R}^d)$, $p \in [1, +\infty)$ and $\theta^*_\sharp \mu$ and $\theta^*_\sharp \nu$ are the push-forward by θ^* of μ and ν respectively. In practice, Eq 4.7 is approximated with a standard Monte Carlo method. We denote by $\mathbf{SW}_p(\mu, \nu)^p_K$ its numerical approximation where K is the number of random projections. Since $\theta^*_\sharp \mu$ and $\theta^*_\sharp \nu$ are univariate distributions, the resulting complexity of the approximation is generally more efficient than resolving Eq. 4.5. The corresponding computational complexity is $\mathcal{O}(Kdn + Kn \log(n))$. We show in section C.1 of the Appendix that the empirical computation time of our approach remains similar to other methods.

4.3.2 Optimal Transport based loss

In this section, we describe our OT-based methodology used to design transferable APA. We denote by f the encoder part of F_θ, where f is composed by a set of J layers; $\mathcal{L} = \{l_1, ..., l_J\}$. Except for the last layer, which usually results directly from an spatial average pooling layer, we apply a spatial average layer to obtain a feature vector. For all $l \in \mathcal{L}$, $f^{(l)}$ maps $x \in \mathcal{X}$ to the feature space $\mathcal{S}^{(l)} = \mathbb{R}^{c_l}$, where c_l is the number of channels for a given layer l.

For a given target class y_{tgt}, we denote by $\nu^{(l)}_{y_{tgt}}$ the conditional multivariate target distribution of $f^{(l)}(X)$ when the class of X is y_{tgt}. The principle of our proposed method

is to design a patch by moving the corrupted image distribution towards the target $\nu_{y_{tgt}}^{(l)}$ and solve:

$$\arg\min_{\delta} \mathbb{E}_X \left[\sum_{l \in \mathcal{L}} OT(\mu_{X_\delta}^{(l)}, \nu_{y_{tgt}}^{(l)}) \right], \tag{4.8}$$

where μ_{X_δ} is the estimated feature distribution of the corrupted source images X_δ (which we will define hereafter) and OT could be $\mathbf{W}_p^p$ or $\mathbf{SW}_p^p$.

In practice, we solve a regularized version of Eq. 4.8 using Expectation over Transformations (EoT) from Athalye et al. (2018). Our patch is trained to optimize the following objective:

$$\arg\min_{\delta} \mathbb{E}_{X,t\sim\mathcal{T},e\sim E} \left[\sum_{l \in \mathcal{L}} OT(\mu_{A(\delta,X,e,t)}^{(l)}, \nu_{y_{tgt}}^{(l)}) + TV(\delta) \right], \tag{4.9}$$

where TV is the total variation loss discouraging high-frequency patterns. We will denote by $(\mathbf{W}_p^p)^{(N)}$ and $(\mathbf{SW}_p^p)^{(N)}$ when we attack N layers by solving the standard or the sliced version of the Wasserstein distance respectively. We choose by convention that for $N = 1$, $l = l_J$, $i.e.$, we are attacking the last layer of f. In Section 4.4.4, we study our APA when designed to attack multiple layers.

L2 method generalization. Herein, we demonstrate that our method generalizes L2-based method. The Wasserstein distance can be interpreted through a probabilistic point of view. If we name (X_1, X_2) a couple of random variables over $\mathbb{R}^d \times \mathbb{R}^d$ with $X_1 \sim \mu$, $X_2 \sim \nu$ and $(X_1, X_2) \sim \pi \in \Pi(\mu, \nu)$ we can write

$$\mathbf{W}_2(\mu, \nu)^2 = \min_{(X_1,X_2)} \mathbb{E}_{(X_1,X_2)\sim\pi} \left[||X_1 - X_2||^2 \right]. \tag{4.10}$$

If we consider the 2-Wasserstein distance and if we assume that the target distribution contains a single point, $i.e.$, $\nu = \delta_{x_i}$ a.s., then we have

$$\mathbf{W}_2(\mu, \nu)^2 = \mathbb{E}_{X_1\sim\mu} \left[||X_1 - x_i||^2 \right], \tag{4.11}$$

which coincides with Eq. 4.4. Minimizing with respect to the 2-Wasserstein for a target distribution containing a single point is equivalent to considering the L2-based criterion proposed by Inkawhich et al. (2019). Our method, in some sense, includes and generalizes the L2-based method.

4.4 Experiments

This section evaluates our APA through digital, hybrid and physical world experiments. In all experiments, the objective is to craft an APA with a high targeted success rate (tSuc) (see Equation 2.17 for a reminder).

We consider the single-source model setting and test attacking transferability to other models. Transferability is tested between ImageNet-1K (Deng et al., 2009) models. We cluster these models into several categories depending on their architecture and learning strategies (learning paradigm and training recipe, see Table 4.4).

Table 4.4: Summary of the model used to evaluate APA transferability. Models are clustered according to their architecture and learning strategy (learning paradigm and training recipe). DeiT is a ViT network trained using distillation to speed-up the training (the distillation differs from the one presented in Figure 2.12b). T, S and B stands for Tiny, Small, and Base respectively. AT stands for Adversarially trained, PMZ for Pytorch Model Zoo and TIM for Timm Model Zoo

Model Category	Architecture	Learning paradigm	Training Recipe	Source
CNNs-v1	ResNet 18/34/50-v1 DenseNet 121/161/169/201 VGG19, Inception-V3	Sup	Old	PMZ
CNNs-v2	ResNet 50-V2/50-Self	Sup Self-Sup	Recent	PMZ and Caron et al. (2020)
ENet	EfficientNet B0/B1/B2/B3/B4	Sup	Recent	PMZ
CNext	ConvNext T/S	Sup	Recent	PMZ
DeiT	DeiT T/S/B	Sup	Recent	TMZ
Swin	Swin T/S/B	Sup	Recent	PMZ
AT	ResNet50 ReLU Adv DeiT S Adv	Adv Sup	Recent	Bai et al. (2021)

Evaluated methods. We consider GAP (Brown et al., 2017), LaVAN (Karmon et al., 2018), TnT (Doan et al., 2022) and Casper et al. (2022) as decision boundary-based baselines. Because of its ease of computation compared to Inkawhich et al. (2020), which requires off-line training of multiple specific auxiliary models, we choose to adapt the proposed method by Inkawhich et al. (2019) as a baseline (we name it L2) to craft an APA based on attacking the feature space. We also adapt the recent state-of-the-art works on transferable adversarial noises (Naseer et al., 2021; Zhao et al., 2023) to craft an APA. To do so, we convert generative methods (Naseer et al., 2021; Zhao et al., 2023) to iterative ones where the objective is to create one universal APA and not one for each image (see Paragraph 2.2.2 for a reminder).

Experimental setup. For comparison, baselines and our method are crafted using the same patch training recipes and are describe bellow. To control the balance between the adversarial loss and the total variation loss (see for example Equation 4.9), the gradient of each loss is computed individually, normalized, and combined using a weighted sum. Patch values are clipped into the image range at each iteration. Following prior works Brown et al. (2017); Lee and Kolter (2019); Casper et al. (2022), we choose and fix the sampling distributions from EoT (Eykholt et al., 2018) for all the methods. During training and evaluation, patches are randomly placed to the side of images (to avoid occluding the object of interest, which is usually centered in image classification). Transformations and noises are also applied to the patch to mimic real-world situations. Appendix C.4 evaluates models' robustness according to the position of the patch in the image. We randomly choose nine targeted classes (see Appendix C.1 for details) and design a patch to fool the network targeting each class. We split the ImageNet-1K validation set into a training set of 40000 images on which we train patches and a test set of 10000 images on which we evaluate their impact. The patch optimization is performed using 100 epochs

Table 4.5: Best transfer results from a single model to all others obtained for each method (tSuc (%) higher is better for an attack).

Method	min	mean	max
GAP (Brown et al., 2017)	2.22	15.46	37.33
LaVAN (Karmon et al., 2018)	2.26	8.67	31.4
L2 (Inkawhich et al., 2019)	4.44	13.6	32.78
TnT (Doan et al., 2022)	0.67	2.11	5.84
Casper et al. (2022)	0.33	3.81	14.85
TTP Naseer et al. (2021)	2.33	13.77	31.87
M3D Zhao et al. (2023)	0.84	5.19	17.11
Ours $(\mathbf{SW}_2^2)_{500}^{(1)}$	**8.93**	**22.56**	**45.31**
Ours $(\mathbf{W}_2^2)^{(1)}$	**8.09**	**21.14**	**49.1**

(1 epoch equals 1000 iterations) with a batch size of 50 images. Three different learning rates are tested (*i.e.*, 0.1, 0.5, 1). We choose for our method $p = 2$ and $K = 500$ as hyperparameters for our method (reasons are explained in Appendix C.7). We evaluate the APA with the best training loss among the three learning rates, leading to one patch per method and per class. Finally, reported tSuc is the average over the classes and patch sizes (from 70×70 to 90×90), which is the standard setting considered in the literature (Brown et al., 2017; Poursaeed et al., 2018; Doan et al., 2022; Casper et al., 2022)).

4.4.1 Numerical experiments

Transferability among networks. We select from the previously defined families the following models: ResNet34, ResNet50-V1, ResNet50-V2, ResNet50-self, EfficientNet-B0, ConvNext-S, DeiT-S, Swin-T, Swin-S, Swin-B. We design patches to attack one of these source models. Then, we measure the attacking transferability when the resulting patch is used to fool the remaining models (target models).

For example, patches trained and tested on the Swin family provide three patches. Each is trained individually on Swin-T, Swin-S or Swin-B, and is evaluated against the two other Swin models. Table 4.5 summarizes, for each method, the best transferring attack performance. According to the method, we select and report the results of the source family producing the highest mean targeted success rate (tSuc, rate at which the attacked images are classified as the patch target label). Our method shows the best transferability capacity: highest mean, min and max tSuc.

Table 4.6 details transferability results for all source families. From this table, we can make several conclusions: Networks trained with older training recipes (CNNs-v1) seem more vulnerable to attacks regardless of the attacking procedures. These networks are more sensitive to salient patches present in the image. As presented in Bai et al. (2021), new training recipes (scheduler, augmenting training data like RandAug and Mixup, ...) make models more robust for convolutional networks and transformers.

Table 4.6: Transfer results (tSuc (%), higher is better) between categories of models. Results are averaged over classes, over patch sizes, and over networks within a category. Patches are placed randomly in the image without object overlapping. Physical transformations ($e.g.$, noise, rotations) are applied to patches. Control is inserting a real object of the corresponding class as a patch.

		Target						mean / std
		CNNs-v1	CNNs-v2	ENet	CNext	DeiT	Swin	
Method	Clean / Source	0.1	0.1	0.1	0.1	0.1	0.1	0.1 / 0
Control		2.85	1.59	0.86	0.54	1.57	0.93	1.39 / 0.75
GAP (Brown et al., 2017)	CNNs-v1	36.61	9.64	5.54	2.43	4.03	3.05	10.22 / 12.04
	CNNs-v2	15.6	9.57	3.66	2.74	3.51	2.49	6.26 / 4.82
	ENet	37.33	10.11	29.88	2.22	8.91	4.3	15.46 / 13.28
	CNext	0.33	0.77	0.23	0.97	0.43	0.8	0.59 / 0.27
	DeiT	1.43	1.97	0.46	1.25	11.54	3.58	3.37 / 3.78
	Swin	1.46	1.54	0.66	1.33	1.58	6.15	2.12 / 1.83
LaVAN (Karmon et al., 2018)	CNNs-v1	31.4	8.56	4.32	2.26	2.49	3.01	8.67 / 10.38
	CNNs-v2	11.08	9.68	2.33	2.45	2.24	2.13	4.98 / 3.84
	ENet	8.74	4.76	11.31	1.08	3.33	2.58	5.3 / 3.59
	CNext	0.45	0.63	0.26	0.44	0.47	0.87	0.52 / 0.19
	DeiT	2.1	1.53	0.93	0.61	5.84	2.45	2.24 / 1.73
	Swin	1.45	1.41	0.57	1.31	1.29	9.44	2.58 / 3.08
L2 (Inkawhich et al., 2019)	CNNs-v1	14.76	5.06	2.69	0.88	1.78	1.08	4.37 / 4.85
	CNNs-v2	3.25	3.5	0.64	1.44	0.57	1.04	1.74 / 1.19
	ENet	14.33	4.12	13.35	0.79	3.02	1.88	6.25 / 5.47
	CNext	2.46	9.66	0.92	20.2	1.73	10.67	7.6 / 6.81
	DeiT	17.88	10.23	8.15	4.44	32.78	8.1	13.6 / 9.5
	Swin	8.2	8.54	3.22	7.24	5.38	23.24	9.3 / 6.49
TnT (Doan et al., 2022)	CNNs-v1	5.84	1.5	2.12	0.67	1.43	1.08	2.11 / 1.73
	CNNs-v2	1.82	0.69	0.59	0.37	0.52	0.6	0.77 / 0.48
	ENet	2.13	0.92	1.4	0.43	0.71	0.64	1.04 / 0.57
	CNext	0.48	0.49	0.24	0.32	0.3	0.4	0.37 / 0.09
	DeiT	1.12	0.85	0.61	0.58	2.43	1.03	1.1 / 0.63
	Swin	1.41	1.03	0.55	0.81	1.61	1.68	1.18 / 0.42
Casper et al. (2022)	CNNs-v1	12.87	1.62	1.2	0.28	0.33	0.19	2.75 / 4.56
	CNNs-v2	7.8	7.44	1.26	0.83	0.95	0.78	3.17 / 3.15
	ENet	5.37	0.85	14.85	0.33	0.68	0.78	3.81 / 5.23
	CNext	0.42	0.28	0.22	0.45	0.15	0.22	0.29 / 0.11
	DeiT	2.86	1.38	0.98	0.91	10.19	2.22	3.09 / 3.25
	Swin	0.56	0.4	0.35	0.52	0.32	1.87	0.67 / 0.54
TTP (Naseer et al., 2021)	CNNs-v1	35.4	8.41	5.43	1.58	3.46	2.29	9.43 / 11.83
	CNNs-v2	17.55	9.67	3.3	3.87	3.66	3.87	6.99 / 5.21
	ENet	31.87	8.88	27.16	2.33	8.75	3.65	13.77 / 11.47
	CNext	0.49	3.39	0.22	7.87	0.48	2.71	2.53 / 2.67
	DeiT	3.87	3.24	1.51	1.64	13.75	3.85	4.64 / 4.18
	Swin	1.53	1.22	0.53	0.98	1.18	5.54	1.83 / 1.69
Zhao et al. (2023)	CNNs-v1	17.11	6.18	3.59	0.84	1.98	1.43	5.19 / 5.61
	CNNs-v2	7.45	11.77	1.77	2.13	1.51	2.33	4.49 / 3.84
	ENet	11.21	1.97	3.34	0.54	1.0	0.88	3.16 / 3.72
	CNext	0.34	0.36	0.16	0.21	1.78	0.28	0.52 / 0.57
	DeiT	2.39	1.59	0.81	1.07	7.7	2.9	2.74 / 2.33
	Swin	1.85	1.63	0.71	0.82	1.55	3.6	1.69 / 0.95
Ours $(\mathbf{SW}_2^2)_{500}^{(1)}$	CNNs-v1	25.25	6.15	4.73	1.7	5.15	2.61	7.6 / 8.04
	CNNs-v2	16.93	8.67	4.02	4.08	5.77	3.56	7.17 / 4.69
	ENet	22.53	5.83	18.8	2.07	8.49	3.03	10.13 / 7.8
	CNext	3.97	11.62	1.1	29.97	3.14	14.75	10.76 / 9.86
	DeiT	23.65	12.16	7.27	5.21	32.39	9.35	15.01 / 9.77
	Swin	25.2	20.21	8.93	19.54	16.16	45.31	22.56 / 11.3
Ours $(\mathbf{W}_2^2)^{(1)}$	CNNs-v1	39.65	13.01	8.27	2.44	4.89	3.16	11.9 / 12.91
	CNNs-v2	19.0	11.35	3.82	4.51	3.74	4.19	7.77 / 5.69
	ENet	35.12	10.45	32.0	2.27	7.8	3.79	15.24 / 13.25
	CNext	3.47	12.2	0.92	25.14	2.04	15.12	9.82 / 8.64
	DeiT	22.26	11.43	10.18	5.29	39.51	9.25	16.32 / 11.59
	Swin	20.55	17.89	8.09	17.7	13.55	49.1	21.14 / 13.12

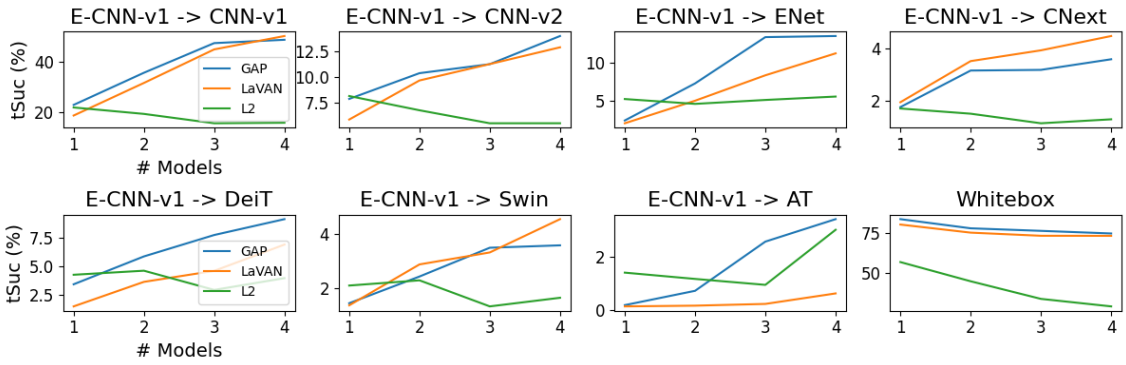


Figure 4.2: Transfer and white-box results for patches built on an ensemble of models (tSuc (%)). Results are averaged over classes and patch sizes. Patches are placed randomly in the image but not at the center of the images.

Baseline methods could not create a patch that can strongly transfer to CNext and Swin models even when these patches are learned using the same model category. This indicates that rather than catching the useful common information in deep networks, baseline methods produce a patch that tends to overfit on the specific weights of the model. This is particularly the case for decision boundaries-based APAs (GAP (Brown et al., 2017), LaVAN (Karmon et al., 2018), TnT (Doan et al., 2022) and Casper et al. (2022)). Naseer et al. (2021) method seems to create a patch that better captures the source model decision boundaries. This method leads to better results than decision boundary-based APAs on the DeiT family of networks (networks with more complex representations than CNNs-V1 networks). Creating a patch that minimizes the maximum discrepancy between two models (Zhao et al., 2023) is unstable and generally results in a patch that is not transferable. A possible explanation is that the APA induces a higher shift in the feature space than adversarial noises.

A patch resulting from an optimization defined in the feature space reduces the patch overfitting and increases the transferability to other networks (see table lines of the L2 and our method in Table 4.6). This suggests that the patch has learned more about the common information to model the different classes rather than trying to cross the decision boundaries. However, the L2 methodology (Inkawhich et al., 2019) is unstable and is highly dependent on the choice of the target point, resulting in lower performance than our method (see Appendix C.2). Our two methods (exact and sliced version) outperform the previous methods in terms of transferability. We remark that patches learned using Swin or CNext seem more universal as they can transfer to multiple models. When crafted on Swin models, we produce a patch capable of transferring uniformly well to all the models.

Ensemble methods. Ensemble methods train a single patch across an ensemble of models simultaneously. By doing so, the patch is expected to exploit the common vulnerabilities in networks, resulting in a more transferable attack. We determine if an attacker building his attack on an ensemble of CNN-v1 models can significantly increase its attacking performance on other models such as CNext or Swin. We consider the following ordered list of models E-CNN-v1 $=$ {ResNet50/34/18-v1, DenseNet121} in which networks are added to the ensemble in this order. Figure 4.2 plots the tSuc as a function of the number of models in the ensemble. Even with the largest ensemble of four models, patches failed to increase their transferability performances on CNext and Swin models

Table 4.7: Transfer results on robustified models by LGS defense (Naseer et al., 2019b) (tSuc (%)). Patches are designed on Swin models.

		Target						
		CNNs-v1	CNNs-v2	ENet	CNext	DeiT	Swin	mean / std
$\lambda = 1.5$	Clean	0.1	0.1	0.1	0.1	0.1	0.1	0.1 / 0
	GAP (Brown et al., 2017)	0.72	0.87	0.35	0.78	1.13	2.34	1.03 / 0.63
	LaVAN (Karmon et al., 2018)	0.56	0.69	0.3	0.69	0.82	2.65	0.95 / 0.78
	L2 (Inkawhich et al., 2019)	4.79	6.44	1.72	7.79	4.79	13.85	6.56 / 3.75
	TnT (Doan et al., 2022)	0.84	0.59	0.52	0.53	0.7	0.85	0.67 / 0.13
	Casper et al. (2022)	0.37	0.4	0.2	0.32	0.25	0.59	0.36 / 0.13
	TTP (Naseer et al., 2021)	0.68	0.77	0.28	0.68	0.76	1.98	0.86 / 0.53
	M3D (Zhao et al., 2023)	0.83	0.81	0.36	0.77	1.17	1.17	0.85 / 0.27
	Ours $(\mathrm{SW}_2^2)_{500}^{(1)}$	**10.56**	**11.86**	**3.81**	**18.9**	**11.67**	**31.68**	**14.75 / 8.75**
	Ours $(\mathrm{W}_2^2)^{(1)}$	**13.23**	**13.4**	**4.37**	**21.42**	**13.84**	**32.08**	**16.39 / 8.58**

significantly. This result suggests that an attacker expecting to sway all the models uniformly should design his attack on Swin models when using our approach. Figure 4.2 also shows that the feature point method becomes unstable with the increased number of models in the ensemble. The following experiments are performed on Swin patches as they lead to a more uniform transferability across networks.

Effectiveness against robust networks. We now consider a more realistic scenario in which the attacked system uses a defense mechanism. We propose to use Local Gradients Smoothing (LGS) (Naseer et al., 2019b), as it is one of the strongest defense mechanisms against patch attacks. The LGS strategy smooths salient regions in images before feeding them to the network. We reproduce the previous experiments for three different smoothing factors $\lambda \in \{1.5, 1.9, 2.3\}$ for LGS while fixing other parameters as in the article (we report here results for $\lambda = 1.5$, see C.9 for other results). For each method, we evaluate the patches designed using Swin architecture against networks made robust by LGS. Our method achieves the best transfer results demonstrating the criticality of our attack even for robust networks (Table 4.7).

We now suppose the target network has been adversarially trained (AT) against adversarial noises. The patch attacks that are not learned on AT models could reduce their accuracy when transferred to these models. However, the AT models do not get fooled by the patch to predict the targeted class. (clean accuracy: 65.44, attacked accuracy: 57.65, tsuc: 0.72). AT models seem to have different class representations and are hard to force to predict a chosen class. When designed on one AT model and transferred to another model, our patches and GAP patches produce the best transfer performances (see Appendix C.5).

4.4.2 Hybrid experiments

In this section, we propose to measure the physicality of patches through a hybrid experiment to simulate the potential effect of patches in the real world. We consider the following steps: printing and digitalization. Scanned patches are placed numerically in images using the same procedure as in the previous section (physical transformations are applied to them). We use patches designed on Swin-T and the results for three different settings (i.e., digital, scan and scan with defense) are reported. Our patch obtains the best transfer results and performs well in scan with defense setting which is more compli-

Table 4.8: Transfer results of digital, scanned, and scanned defended patches (mean tSuc (%)). Patches are designed on Swin models and for the targeted class *birdhouse*.

	Digital	Scan	Scan Defended
Clean	0.1	0.1	0.1
GAP (Brown et al., 2017)	1.3	1.1	0.92
LaVAN (Karmon et al., 2018)	1.53	1.05	0.88
L2 (Inkawhich et al., 2019)	8.65	4.54	4.26
TnT (Doan et al., 2022)	0.78	0.43	0.37
Casper et al. (2022)	1.13	0.49	0.39
TTP (Naseer et al., 2021)	1.06	0.73	0.52
M3D (Zhao et al., 2023)	1.84	1.08	0.68
Ours$(\mathbf{SW}_2^2)_{500}^{(1)}$	**19**	**12.41**	**12.11**
Ours$(\mathbf{W}_2^2)^{(1)}$	**20.04**	**12.59**	**12.41**

cated than the other settings. (see Table 4.8). This result confirms the potentially harmful behavior of our patch in the real-world.

4.4.3 Qualitative physical experiments

Despite our work is focus on increasing APA transferability, we give some qualitative results concerning the physicality of our attack. We select three objects present in ImageNet-1K (banana, cup, keyboard) and record videos of when one patch is placed next to the object or not. During the video, patches are moved around the object. Figure 4.3 shows examples of our patch localized near objects. In the experiments, all the patches were not able to transfer (tSuc lower than 2%), except for L2 and our patches. The transfer results for the L2 method, our first ($(\mathbf{SW}_2^2)_{500}^{(1)}$) and second ($(\mathbf{W}_2^2)_{500}^{(1)}$) methods are 9.3%, 23.4% and 29.3% respectively. These results confirm that real-world classifiers can be swayed without explicit knowledge of their architecture or weights.

Figure 4.3: Examples of frames of our APA close to different objects. Our patch is designed to sway networks to output the targeted class *birdhouse*.

4.4.4 Ablation studies

In this section, we study the impact of the choice of the targeted layers on the patch transferability. We learn our attack ($(\mathbf{W}_2^2)$ version) to push distribution on different layers for Swin models and report results in Table 4.9. The last layer of the encoder ($l = l_J$) seems essential to model and close the gap between the corrupted image distribution and the target distribution. It is coherent since this layer is expected to separate classes before

Table 4.9: Transfer results of digital patches when varying the choice of the targeted layers (tSuc (%)). Patches are designed on Swin models.

$\mathcal{L}$	$\{l_{J-8}, l_{J-2}, l_J\}$	$\{l_{J-2}, l_J\}$	$\{l_{J-2}\}$	$\{l_J\}$
mean	14.66	**23.47**	17.31	21.14

linear classification. We observe that the multi-layer objective leads to better results as it helps the optimization to converge to a better local minimum, leading to a stronger patch (see Fig. C.7 in the Appendix for details about layers). However, most layers fail to model the targeted distribution correctly. In Appendix C.7, we present additional ablation studies, in which we investigate the impact of the power, p, and the number of slices, K.

Through numerical, hybrid, and physical experiments, we show the superiority of our method in designing transferable APA. In the following section, we move towards understanding what phenomena in APA's learning may induce or reduce the resulting APA transferability.

4.5 Study of APA behavior

The different APAs evaluated so far can be classified into two different categories based on their learning principles. Either the APA learning is based on a classifier and its decision boundary (decision boundary based) or the criterion for learning APAs considers the feature space of the network (feature space based). The L2-based method (Inkawhich et al., 2019) and ours are feature space-based while the remaining methods (Brown et al., 2017; Karmon et al., 2018; Doan et al., 2022; Casper et al., 2022; Naseer et al., 2021; Zhao et al., 2023) fall in the first category.

As already introduced in Paragraph 4.2.1, decision boundary-based methods may design a patch that pushes the corrupted image features into unseen regions of the representations space. In the following section, we study how the different APAs influence the features extracted by the encoder. This study may help to understand the underlying attack mechanism of the different APAs.

4.5.1 A feature space study

We propose to evaluate the geometric shift of image features induced by introducing an APA in the image. We compare the GAP method (Brown et al., 2017) and ours as these methods are the most suited to generate transferable APAs for their corresponding category. To quantify the shift in the feature space, we estimate three feature distributions:

- $\mu_{y_{src}}$, the conditional multivariate feature distribution of benign images when the class of images is y_{src}. These benign images are subsequently perturbed using the selected APAs;
- μ_{X_δ}, the conditional multivariate feature distribution of the perturbed images after applying the APAs. We recall that patches are designed to sway the image classifier to output the target class y_{tgt};
- $\nu_{y_{tgt}}$, the conditional multivariate feature distribution of benign images when the class of images is y_{tgt}.

Table 4.10: Feature distribution shift induced by APAs when introduced in images. $\mu_{y_{src}}$ is the feature distribution of benign images when the class of images is y_{src}, μ_{X_δ} is the feature distribution of the perturbed source images and $\mu_{y_{tgt}}$ is the feature distribution of benign images when the class of images is y_{tgt}. Results are averaged over source and target classes.

Network	Method	$\tilde{\mathbf{W}}_2(\mu_{src}, \mu_{X_\delta})^2$	$\tilde{\mathbf{W}}_2(\mu_{X_\delta}, \nu_{y_{tgt}})^2$
Swin-T	Black patch	0.44	1.05
	Random patch	0.42	1.04
	GAP (Brown et al., 2017)	0.66	0.95
	Ours$(\mathbf{W}_2^2)^{(1)}$	0.59	0.78
ResNet50-v1	Black patch	0.25	1.00
	Random patch	0.25	1.00
	GAP (Brown et al., 2017)	0.91	0.97
	Ours$(\mathbf{W}_2^2)^{(1)}$	0.51	0.76
EfficientNet-B0	Black patch	0.27	1.01
	Random patch	0.15	1.00
	GAP (Brown et al., 2017)	0.88	0.94
	Ours$(\mathbf{W}_2^2)^{(1)}$	0.71	0.83
DeiT-S	Black patch	0.43	1.04
	Random patch	0.35	1.02
	GAP (Brown et al., 2017)	0.80	0.96
	Ours$(\mathbf{W}_2^2)^{(1)}$	0.68	0.76

We estimate $\mu_{y_{src}}$ for three randomly selected classes, with each class represented by 1000 randomly sampled images. Similarly, μ_{X_δ} is computed by applying the APA to these images, capturing the distribution of the adversarially perturbed images. For the target class, $\nu_{y_{tgt}}$ is also estimated using 1000 randomly sampled benign images. The target classes are the same as in the previous section (see Paragraph 4.4 for a reminder). For each couple (y_{src}, y_{tgt}), we compute the following two quantities:

$$\tilde{\mathbf{W}}_2(\mu_{src}, \mu_{X_\delta})^2 = \frac{\mathbf{W}_2(\mu_{src}, \mu_{X_\delta})^2}{\mathbf{W}_2(\mu_{src}, \nu_{y_{tgt}})^2}, \quad \tilde{\mathbf{W}}_2(\mu_{X_\delta}, \nu_{y_{tgt}})^2 = \frac{\mathbf{W}_2(\mu_{X_\delta}, \nu_{y_{tgt}})^2}{\mathbf{W}_2(\mu_{src}, \nu_{y_{tgt}})^2}. \tag{4.12}$$

where $\mathbf{W}_2(.,.)^2$ is the 2-Wasserstein distance (see Eq. 4.5 for a reminder). These quantities describe the relative distribution shift induced by APAs. The first quantity encodes how far the perturbed distribution is pushed from the source distribution, and the second quantity encodes how far the perturbed distribution gets closer to the target distribution.

Table 4.10 reports these metrics for several networks. GAP patches (Brown et al., 2017) push the perturbed distribution further away from the source distribution. However, this increased shift does not necessarily bring the perturbed distribution closer to the target distribution. The distribution of images perturbed by GAP remains just as distant from the target distribution as the source distribution (the second metric stays close to 1). Contrary to expectations, designing an APA to cross the decision boundary and force classification into the target class does not simply result in the perturbed images aligning with the target distribution. Instead, it pushes the perturbed distribution into unseen regions of the feature space, far from both the source and target distributions. Pushing into unseen regions may limit the resulting APA transferability as these regions may be network specific. As expected, our method creates an APA that tries to close the gap between the source and

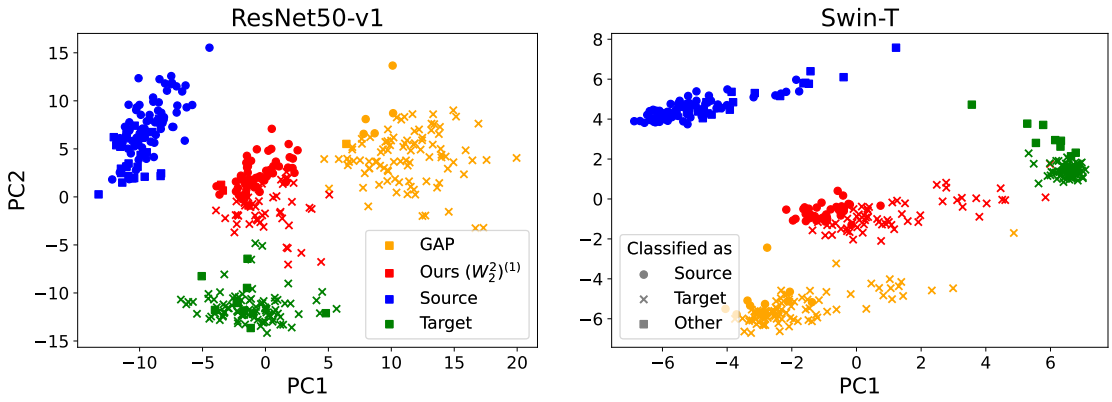


Figure 4.4: First two principal components of the principal component analysis performed on the three distributions. The source class and target class are *Australian terrier* and *eft*, respectively.

the target distribution. For Swin-T (Liu et al., 2021) and ResNet50-v1 (He et al., 2016), the sum of the two metrics is close to 1, indicating, due to the triangle inequality, that the perturbed distribution is close to the geodesic defined between the source and the target distribution.

To qualitatively illustrate our findings, we perform a principal component analysis on the feature points defining the three distributions. Figure 4.4 plots the first two principal components. As previously measured, the GAP perturbations (Brown et al., 2017) cause a significant shift, pushing the perturbed distribution far from the source and target distributions. In contrast, our method seeks to find the shortest path between the two.

4.6 Conclusion

In this chapter, we categorize APAs against image classification models into two groups based on their underlying learning principles. We show that the methods from the first group, decision boundary-based methods (Brown et al., 2017; Karmon et al., 2018; Doan et al., 2022; Casper et al., 2022; Naseer et al., 2021; Zhao et al., 2023), generate an APA that tends to overfit on the source model decision boundaries. This patch overfitting induces the learned APA to push corrupted image features into unseen feature space regions, leading to poor APA transfer capacity. In contrast, the second group, feature space-based methods (Zhou et al., 2018; Rozsa et al., 2017; Inkawhich et al., 2019), relies on the feature space of a neural network to capture universal image information, leading to better attack transferability across models and reducing the risk of overfitting to a specific model's decision boundaries.

Among the feature space-based approaches, the L2 method (Inkawhich et al., 2019) demonstrates better transferability but suffers from numerical instability, with its performance highly dependent on the choice of the targeted feature point. To address these limitations, we introduce a more robust and generalized feature space method. Our novel approach leverages a distribution-oriented framework based on optimal transport for designing APAs. This method mitigates patch overfitting to the source model, significantly enhancing its transferability across both CNNs and Transformer-based architectures.

When applied to Swin models, our patch is the only one capable of effectively fooling multiple architectures from different model families, even in the presence of defense

mechanisms designed to enhance robustness. Furthermore, hybrid and physical-world experiments show that our attack can successfully disrupt real-world classifiers, even without prior knowledge of the system. These results highlight the practical effectiveness and adaptability of our approach in real-world scenarios.

In the following chapter, to study if more complex real-world systems may be vulnerable to physical attacks, we focus on designing a transferable APA against object detectors.

Chapter 5

A transferable inhibition patch to hide objects

In Chapter 4, we introduced a novel APA capable of effectively transferring across a wide range of image classifiers. However, real-world systems, such as vision systems in autonomous vehicles, often involve more complex tasks than image classification. Our goal in this chapter is to design a transferable APA that hides people from being detected by multiple object detectors. We focus on the invisible cloak attack scenario as this scenario is easier than the contextual attack scenario (see Section 2.3.2 for a reminder). First, we review state-of-the-art methods developed to enhance the transferability of APAs for object detectors. We show that their success is due to several limitations and flaws in their evaluation protocol. To mitigate these issues, we propose a new evaluation framework based on multi-label classification. Finally, we examine the performance of white-box and black-box transfer-based attacks using our new evaluation framework.

5.1 A transferable object inhibition APA

We aim to design an APA that can effectively transfer across models to prevent people from being detected. The attack mechanism behind inhibiting detection differs from that used by APAs targeting image classification. For image classification, APAs try to exploit the features used by the network to predict the target class. These features exist as soon as the image classifier can predict the target class. Designing a successful transferable APA for image classification suggests that the APA is based on patterns that different networks extract and use. It was not evident that such a pattern exists because networks can rely on different patterns to predict a class. Nevertheless, Chapter 4 shows that such a pattern exists.

Now, the APA's objective is to disrupt the features used by object detectors to identify the target class. Contrary to image classification, the existence of such an attack has yet to be shown. In fact, object detection is inherently more complex than classification as it is based on more diverse decision mechanisms (textures and shapes identification, use of contrast), thus making it harder to attack or interpret. Herein, we make the hypothesis that some features exist that can inhibit the detection of a class. This assumption may be valid. For small objects, detection can not be performed by recognizing class-related patterns and may rely on contextual information. Such contextual features could be exploited to prevent the detection of a target class. The question of seeking a transferable APA against object detection then becomes: Does a common pattern inhibit the detection of various

object detectors?

The following section presents methods that focus on designing transferable APA against object detection.

5.1.1 Previous transferable APA against object detection

As already introduced at the end of Section 3.4, and through experiments in Section 3.3.3, different invisible cloak APAs developed in the literature show the capacity to reduce the mAP of unknown models when transferred to them.

To increase patch transferability across object detectors, Huang et al. (2023) introduce Transfer-based Self-Ensemble Attack (T-SEA), a set of techniques and tricks. First, they observe that Thys et al. (2019) APA can be improved by modifying the learning rate scheduler and the patch scale ratio (see Paragraph 3.3.2 for a reminder about the patch scale ratio). They propose a new updating rule for learning rate to better handle saddle points during patch optimization. The patch scale ratio is reduced to 0.15 to help the optimized patch to learn more global patterns instead of local patterns. Local patterns are more sensible to patch rescaling, thus limiting their robustness to physical transformations. The other proposed techniques are built on an analogy between patch and model training. As for model training, they use image augmentations to generate new candidates for the patch training. Instead of training the patch to target an ensemble of models, which may require training multiple models, they use Shakedrop (Yamada et al., 2018), a method that randomly drops a subset of layer activations, enabling the training over multiple variants of the same model. Inspired by the use of Dropout (Srivastava et al., 2014) and Cutout (DeVries and Taylor, 2017), they introduce patch cutout, a technique that randomly drops some parts of the patch to avoid the attack to rely on specific patch regions.

When gathered, they show that these techniques and tricks help generate better transferable APA, reducing the mAP on multiple object detectors. However, in the next section, we show that most of these results are due to a misusage of the mAP. Another limitation is that none of the previous works have evaluated the robustness of object detectors that use a Transformer as an encoder. As we showed in Chapter 4, the encoder plays a major role in the prediction process and, therefore, in APA's transferability.

5.1.2 A false sense of inhibition

The objective of an invisible cloak APA is to prevent the detection of a specific target class, typically the *person* class, hence the term invisible cloak. An adversary with such an objective would attempt to influence the detector to refrain from predicting a bounding box for the class *person*. An APA that generates false positives for poodle plants in the vicinity of the patch would have no adverse consequences. In fact, from a global safety perspective, the addition of detections of another class will either be disregarded by the overall system or, in the most unfavorable scenario for the attacker, raise a warning by the system. Detecting more people on or near the patch is obviously counterproductive for the attacker. In summary, the mAP is not an appropriate metric for evaluating the impact of an invisible cloak APA, as it is susceptible to being influenced by other classes false positive.

Person average precision (AP) should be used to avoid false positives from other classes. AP is calculated for a predefined Intersection over Union (IoU) threshold, typ-

Table 5.1: White-box and transfer results for invisible cloak APAs designed either on YOLOv5 or Faster R-CNN (person AP (%)). Clean refers to when there is no attack. Patches are from Huang et al. (2023) work and are available in the github associated with the paper.

	Targeted network		
Source	YOLOv5	Faster R-CNN	DETR
Clean	87.16	85.35	74.6
YOLOv5	1.2	20.53	17.69
Faster R-CNN	11.22	3.57	10.79

ically set to 0.5 in APA-related object detection studies. For instance, this is the setting used in Huang et al. (2023) work. We propose to re-evaluate their state-of-the-art APAs designed in their article and available in their GitHub repository[1]. We select APAs optimized to target a YOLOv5 (Jocher et al., 2022) or a Faster R-CNN (Ren et al., 2015). We evaluate the performance of these APAs when targetting YOLOv5, Faster R-CNN and DETR (Carion et al., 2020) (see Tab. 5.1). We follow Huang et al. (2023) evaluation procedure and measure APAs performance on the INRIA Person test set (Dalal and Triggs, 2005) (see Paragraph 5.2.2 for details about INRIA Person). When targeting YOLOv5 and Faster R-CNN, in both white-box and black-box settings, we measure a comparable attack performance to that reported in the original paper (Huang et al., 2023). When transferred to DETR, patches highly reduce the person's AP.

Nevertheless, this level of attack performance is attributable to an artifact resulting from the IoU threshold process. Examining APA effects on a few images (see Fig. 5.1) reveals two key side effects. First, it generates false positive person detections near the target, which, as previously discussed, is counterproductive for the attacker's goal. Second, it introduces multiple regression errors, causing the predicted height of bounding box to be half of the actual person's height. These regression errors cause the predicted boxes to fail to meet the IoU threshold required to match ground truth boxes. Predicted boxes with an IoU with a ground truth box less than the threshold will be ignored (see the beginning of Section 2.1.2). This would lead to a miss-detection of a person and a false alarm on the person, drastically reducing the AP and, therefore, inducing people to think that the patch correctly hides people from being detected.

We re-evaluate the previous APAs from Huang et al. (2023) work for various IoU thresholds. Figure 5.2 plots the recall according to different IoU threshold values. For both APAs, when the IoU decreases, the recall rapidly increases, indicating that the APA performance falls drastically. In the black-box setting, the recall value matches the clean recall value (clean refers to when there is no attack), suggesting that APAs fail to hide people from being detected. Even in the white-box setting, APA performance is mitigated when the IoU decreases. These results confirm that the attack mechanism relies on introducing bounding box regression errors and fails to hide people.

To ensure that the APA works regardless of the IoU threshold, we introduce a new target objective for invisible cloak APA. The objective is to ensure that no individual is identified in images when applying *multi-label classification*.

[1] https://github.com/VDIGPKU/T-SEA

Figure 5.1: Examples of white-box (designed on YOLOv5 and applied to it) and black-box transfer invisible cloak attack. The patch is designed on YOLOv5 and is from Huang et al. (2023) work. Patches are designed to prevent persons from being detected.

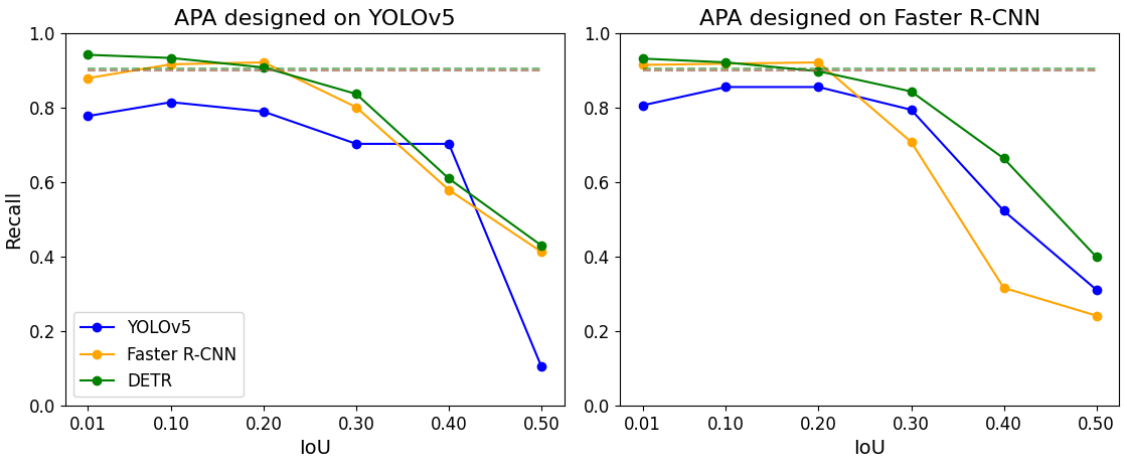


Figure 5.2: Recall as a function of the IoU threshold value used in detection evaluation. Dashed lines represent detectors' clean performance when 0.5 is the IoU threshold. APAs are applied to YOLOv5, Faster R-CNN, and DETR. On the left, the APA is learned on YOLOv5, and on the right figure, the APA is learned on Faster R-CNN. APAs are from Huang et al. (2023) work. Patches are designed to prevent the detection of persons.

5.2 Adressing inhibition using multi-label classification as a surrogate problem

5.2.1 Definition

Unlike object detection, multi-label classification suppresses the regression aspect, focusing only on categorizing the presence of the labels in an image. Multi-label image classification involves multiple binary classification tasks, where each class is treated as a separate binary decision, predicting whether the given label is in the image or not. Consequently, the method leverages binary classifiers that operate in parallel, making decisions independent for each label, which are then aggregated to produce the final set of predictions. By deactivating the objective of predicting the bounding box, multi-label classification de facto eliminates the influence of the IoU threshold on performance.

To create multi-label classifiers, we extract the encoder part from image classifiers or object detectors. On top of this feature encoder, we add a spatial average pooling operation to reduce the spatial dimension to obtain a feature vector. This feature vector is then independently used by the binary classifiers. Figure 5.3 illustrates the architecture of a multi-label classification network.

Once trained, multi-label classifiers are evaluated using the AP metric. As there is a single output for each image and no more multiple proposals, the AP score can not be perturbed by false positives when evaluating APA. The difference between the AP before and after the attack directly encodes the inhibition capacity of the patch.

5.2.2 Experimental setup

Multi-label training. In this section, we detail the different networks selected for multi-label classification. The following sections use these networks to design and test APA performance. Two strategies can be employed to adapt networks: the whole network can be fine-tuned on the new task, or only the multi-label head can be trained as linear prob-

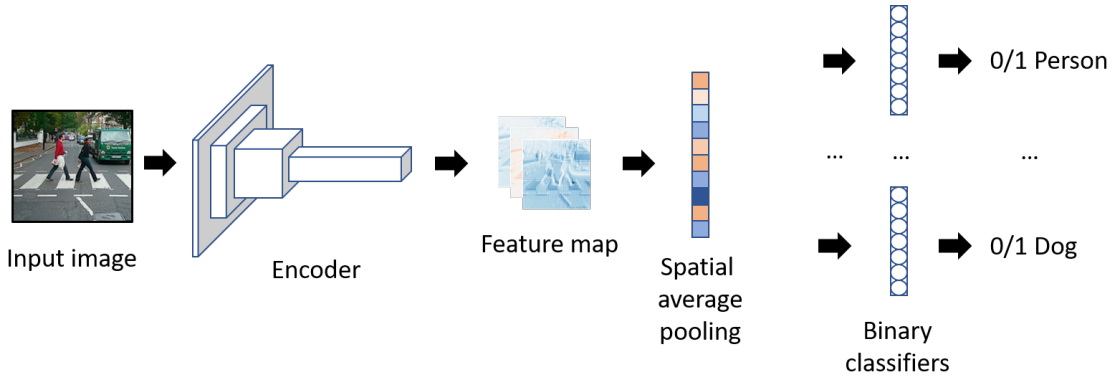


Figure 5.3: Illustration of a multi-label classifier.

ing. We choose to perform linear probing to keep the different encoding space properties unchanged. Keeping the different encoders unmodified helps to study the most suited encoder architecture for designing a transferable APA that hides people. Freezing the encoder also helps to directly compare the behavior of models used in image classification and object detection. It ensures that the fine-tuning does not change the extracted features, impacting the fair comparison between the two networks.

As for image classification, we use binary linear predictors for the output head and train them independently using the following optimization problem:

$$\forall j, \quad \arg\min_{\theta_j} \sum_{i=1}^{n} \mathcal{L}_{\text{BCE}}(T_{\theta_j} \circ f(x_i), y_{i,j}), \tag{5.1}$$

where $\mathcal{L}_{\text{BCE}}$ is the binary cross-entropy loss, f is the image encoder outputing a feature vector, $y_{i,j} \in \{0, 1\}$ is the binary label for the sample n and the label j indicating the presence of the label j in the image x_i, $T_{\theta_j} : \mathbb{R}^C \to [0, 1]$ is the linear head for the label j and θ_j its associated parameters. We will omit the parameter notation in the following.

Selected encoders. We extract encoders from already trained image classifiers and object detectors. We select several ResNet50 encoders (He et al., 2016): ResNet50-v1 and ResNet50-v2 image classifiers from Pytorch Model Zoo, ResNet50-DETR extracted from DETR (Carion et al., 2020), ResNet50-FRCNN extracted from Faster R-CNN (Ren et al., 2015) available in Pytorch Model Zoo, ResNet50-DINO from Caron et al. (2021) and ResNet50-MoCov3 (Chen et al., 2021). We also extract several ViT-B encoders (Dosovitskiy et al., 2020) and a Swin-T encoder (Liu et al., 2021).

Datasets and training recipes. The Common Objects in Context (COCO) (Lin et al., 2014), Pascal VOC (Everingham et al., 2008), and INRIA Person (Dalal and Triggs, 2005) datasets are widely utilized in computer vision research, particularly in object detection and image segmentation tasks. The COCO dataset is a large-scale and diverse set of images containing over 330,000 images with 80 object categories. It includes complex scenes with multiple objects per image. Pascal VOC provides a smaller dataset than COCO, with approximately 20 object categories. It contains less diverse scenes, object types, and sizes than COCO. The INRIA Person dataset is specifically focused on the detection of humans in images. It contains mainly outdoor scenes and is designed for pedestrian detection tasks, offering fewer categories but more detailed attention to person annotation.

Table 5.2: Multi-label clean performance when freezing the encoder and training only the binary classifier (person AP (%)). The multi-label classifier is trained on the COCO training set (Lin et al., 2014).

	Test dataset		
Model	COCO	Pascal VOC	INRIA
Swin-T	98.77	96.38	99.46
ResNet50-v1	98.87	97.11	99.46
ResNet50-v2	99.20	97.80	99.61
ResNet50-DINO	99.04	97.44	99.67
ResNet50-FRCNN	99.04	97.04	99.74
ResNet50-DETR	99.31	98.11	97.87
ResNet50-MoCo	98.99	97.42	99.79
ViT-B-MoCo	98.16	95.76	99.52
ViT-B-MAE	97.39	94.80	98.30
ViT-B-DINO	99.02	97.22	99.59

We train binary classifiers using the COCO training dataset because it offers a more diverse range of images. For optimization, we employ stochastic gradient descent (SGD) with a momentum of 0.9, the batch size is set to 512, and we train the models for 100 epochs. Table 5.2 reports the person's AP of the trained classifiers on COCO (Lin et al., 2014), Pascal VOC (Everingham et al., 2008) and INRIA Person (Dalal and Triggs, 2005) test sets. The different networks reach remarkable performances across the three datasets.

5.2.3 Previous APAs on multi-label classification

In this section, we evaluate our *birdhouse* APA designed in the previous chapter (see Section 4.4.3 for a reminder) and the object detector APAs from Thys et al. (2019) and Huang et al. (2023) works (same as in Section 5.1.2) . APAs are tested on the INRIA Person test set (Dalal and Triggs, 2005) as it offers a more classical framework; persons are centered in images and close to the camera. None of the APAs show the capacity to fool the multi-label classifiers and even when the source and the target network share the same encoder (T-SEA Faster-RCNN on ResNet50-FRCNN, and Ours Swin-T on Swin-T, see Table 5.3). These results highlight the fact that current APAs (Thys et al., 2019; Huang et al., 2023) are not critical for inhibiting the detection of persons in images. In the following section, we develop new APAs against multi-label classification.

5.3 A new APA against multi-label classification

In this section, we seek a method to create a transferable APA against multi-label classification. As shown in the previous chapter (see Paragraph 4.2.2 for a reminder), optimizing an APA using a criterion defined in the feature space helps reduce patch overfitting on the source model, thus improving its transferability. To build on this, we try to adapt our optimal transport-based methodology to create a transferable APA against multi-label classification (see Section 4.3.2 for a reminder). To do so, we need to define a target distribution, which intuitively may be here the *no person* class distribution defined by images that do not contain persons. The goal may be to push the attacked *person* distribution to-

Table 5.3: Transfer results of invisible cloak patches. Difference between the clean performance of the network and its performance when exposed to attacks (person AP difference (%), a higher value indicates a better attack). Patches are resized and placed in the middle of person detection for the INRIA Person test set.

	Method			
	T-SEA (Huang et al., 2023)		Thys et al. (2019)	Ours (Chapter 4)
Source / Target	YOLOv5	Faster-RCNN	YOLOv2	Swin-T
Swin-T	0	0	0	0.01
ResNet50-v1	0.74	0	0.26	0.35
ResNet50-v2	0	0	0	0
ResNet50-DINO	0	0	0	0.04
ResNet50-FRCNN	0.04	0	0.21	0.09
ResNet50-DETR	0.22	0	0.01	0.44
ResNet50-MoCo	0.02	0	0	0.03
ViT-B-MoCo	0	0	0	0
ViT-B-MAE	0.05	0.04	0.08	0.12
ViT-B-DINO	0.17	0	0	0

ward the *no person* distribution. However, this is not feasible due to the structure of the *no person* distribution. Unlike class distributions, the *no person* distribution is highly variable and does not exhibit a consistent structure. The non-presence of features responsible for the person's detection is not encoded uniquely.

An alternative may be to push the attacked *person* distribution toward a target class distribution as in Chapter 4. However, as we have shown in the previous section, this strategy fails. Instead of hiding people, the patch introduces features from the target class without inhibiting the detection of the person. While effective in image classification, the OT-based method is unsuitable for crafting an APA to suppress the detection of a specific class. We thus rely on the decision boundary of the multi-label classifier.

5.3.1 An inhibition APA

In this section, we define the strategy used to attack multi-label classifiers. We consider the standard notation where $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, ..., n$, are samples drawn from a joint distribution of random variables X and Y. As we are considering multi-label classification problem, the input is sampled from $\mathcal{X} = \mathbb{R}^{h \times w \times c}$, where $h \times w$ are the image dimensions and c is the number of channels. Let $\mathcal{T}$ be a distribution over transformation (e.g., rotations, scaling, blur, ...) and E a distribution over locations. For a given source class y_{src}, that we seek to inhibit, our patch is trained to optimize the following objective:

$$\arg\min_{\delta} \mathbb{E}_{X,t\sim\mathcal{T},e\sim E} \left[F_{y_{src}}(A(\delta, X, e, t)) + TV(\delta) \right], \tag{5.2}$$

where $F_{y_{src}} = T_{y_{src}} \circ f$ and $T_{y_{src}}$ is the binary classifier associated with the class y_{src}, $A(\delta, x, e, t)$ the patch applicator operator in an image x where δ is the patch (see Fig. 2.18 for a reminder), t is a ensemble of patch transformations and e is the patch location in the image x. As decision boundary-based methods tend to create an APA that pushes attacked feature representations into unseen regions, we propose to reduce the dimension

of the feature space. It is expected that this reduction will limit APA's overfitting in those uninformative regions. We project feature representations using a Principal Component Analysis (PCA) (Jolliffe and Cadima, 2016) and learn the binary classifier from this reduced feature space. The binary classifier is learned using the same learning recipe as in Section 5.2.2. Then, we optimize the patch using the following objective:

$$\arg\min_{\delta} \mathbb{E}_{X,t\sim\mathcal{T},e\sim E}\left[\tilde{F}_{y_{src}}(A(\delta,X,e,t)) + TV(\delta)\right], \tag{5.3}$$

where $\tilde{F}_{y_{src}} = T_{y_{src}} \circ PCA \circ f$. We also considered the targeted version of the two previous losses by adding a term to the loss to sway a binary classifier to output another target class. For example, the targeted version of the non-reduced inhibition is:

$$\arg\min_{\delta} \mathbb{E}_{X,t\sim\mathcal{T},e\sim E}\left[\underbrace{F_{y_{src}}(A(\delta,X,e,t))}_{\text{inhibition term}} - \underbrace{F_{y_{tgt}}(A(\delta,X,e,t))}_{\text{targeted term}} + TV(\delta)\right], \tag{5.4}$$

where the targeted term may limit the APA to push the attacked feature representations into unseen regions.

5.3.2 APA evaluation

Experimental setup. In this section, we evaluate four APA methods developed in the previous section. We consider the same setting as in Section 5.2.2: same selected encoders (ResNet (He et al., 2016), ViT (Dosovitskiy et al., 2020) and Swin (Liu et al., 2021)) and multi-label classifiers are learned on the COCO training dataset (Lin et al., 2014). For the reduced-Attacks, which involve projecting features using PCA, we select the first twenty principal components. This selection is based on the observation that including additional components does not increase the AP for the *person* class. We reduce the dimension from 2048, 1536 and 768 to 20 for ResNets, ViT and Swin respectively. The different APAs are evaluated on the non-reduced feature space.

As in Chapter 4, the gradient of the adversarial loss and the total variation loss are computed individually, normalized, and combined using a weighted sum (see Paragraph 4.4 for a reminder). Patch values are clipped into the image range at each iteration. We fix sampling distributions from EoT (Eykholt et al., 2018) for all the methods. Patches are trained on the INRIA Person train set (Dalal and Triggs, 2005). The patch optimization is performed using 100 epochs with a batch size of 15 images. Two different learning rates are tested (i.e., 0.5, 1), and we evaluate the APA with the best training loss among the two learning rates.

Results. We evaluate the white-box attack performance and the transferability of the different APAs and report the results in Table 5.4-top for the non-reduced attacks and in Table 5.4-bottom for the reduced attacks. For the unmodified inhibition attack (named *In* in Table 5.4), only a small fraction of the APA successfully fool their source model in the white-box setting (R50-v1, R50-FRCNN, R50-DETR, VIT-B-DINO). The other models seem to be more robust. Even when succeeding in the white-box setting, the corresponding APAs fail to transfer to other models, even for models sharing the same architecture. For most of the models, adding a targeted term in the loss degrades the white-box performance and the transferability (*In-Tgt* in Table 5.4). Reducing the feature dimension

Table 5.4: Transfer results of invisible cloak patches. Difference between the clean performance of the network and its performance when exposed to attacks (person AP difference (%), a higher value indicates a better attack). Patches are resized and placed in the middle of person detection for the INRIA Person test set. *In* stands for the inhibition attack (Eq. 5.3), *In-Tgt* stands for the inhibition targeted attack. *R-In* stands for the reduced inhibition attack (Eq. 5.4), *In-Tgt* stands for the reduced inhibition targeted attack.

Method	Target / Source	Swin-T	R50						ViT-B		
			v1	v2	DINO	FRCNN	DETR	MoCo	MoCo	MAE	DINO
In	Swin-T	**1.8**	0.7	0.5	0.0	0.0	2.0	0.0	0.0	1.3	0.0
	R50-v1	0.0	**37.3**	0.2	0.0	0.0	2.4	0.0	0.0	1.3	0.0
	R50-v2	0.2	1.2	**7.6**	0.2	0.1	1.9	0.0	0.0	1.6	0.0
	R50-DINO	0.3	1.8	0.0	2.8	0.2	3.6	0.1	0.1	1.9	0.1
	R50-FRCNN	0.5	3.5	0.6	0.7	**8.1**	5.0	0.3	0.2	1.6	0.2
	R50-DETR	0.0	2.7	0.0	0.0	0.0	**10.4**	0.0	0.0	0.0	0.0
	R50-MoCo	0.4	2.5	0.2	0.7	0.7	3.2	3.5	0.2	1.9	0.1
	ViT-B-MoCo	0.2	0.8	0.0	0.2	0.0	2.3	0.0	0.0	1.5	0.0
	ViT-B-MAE	0.0	0.6	0.0	0.0	0.0	1.2	0.0	0.0	**2.0**	0.0
	ViT-B-DINO	0.2	0.8	0.0	0.0	0.1	2.6	0.0	0.1	1.4	**31.5**
In-Tgt	Swin-T	1.6	0.4	0.0	0.0	0.0	1.8	0.0	0.0	1.2	0.0
	R50-v1	0.2	23.5	0.6	0.7	1.0	4.0	0.1	0.0	1.3	0.4
	R50-v2	0.6	1.2	4.4	0.9	0.4	2.5	0.1	0.0	1.5	0.0
	R50-DINO	0.4	3.8	1.0	**6.6**	1.7	4.0	0.5	0.1	1.5	0.0
	R50-FRCNN	0.5	2.7	0.9	0.9	7.7	5.0	0.5	0.2	1.5	0.1
	R50-DETR	0.0	4.8	0.0	0.0	1.2	7.9	0.0	0.0	0.0	0.0
	R50-MoCo	0.5	1.5	0.9	1.0	1.7	3.2	**5.3**	0.2	1.7	0.0
	ViT-B-MoCo	0.1	1.2	0.0	0.0	0.0	2.5	0.0	0.0	1.7	0.0
	ViT-B-MAE	0.0	0.2	0.0	0.0	0.0	1.4	0.0	0.0	1.6	0.0
	ViT-B-DINO	0.3	2.3	0.3	0.5	0.8	3.0	0.2	0.0	1.6	14.9
R-In	Swin-T	0.3	0.5	0.4	0.0	0.0	1.8	0.0	0.0	1.6	0.0
	R50-v1	0.1	3.8	0.0	0.0	0.0	2.3	0.0	0.0	1.6	0.0
	R50-v2	0.0	0.9	5.2	0.0	0.0	2.1	0.0	0.1	1.5	0.0
	R50-DINO	0.4	0.9	0.2	1.9	0.2	2.8	0.0	0.1	1.8	0.0
	R50-FRCNN	0.4	3.2	0.8	0.8	7.5	4.9	0.5	0.2	1.5	0.2
	R50-DETR	0.0	1.4	0.0	0.0	0.0	8.7	0.0	0.0	0.0	0.0
	R50-MoCo	0.5	1.8	0.4	0.5	0.2	2.6	1.1	**0.3**	2.0	0.1
	ViT-B-MoCo	0.1	1.1	0.0	0.0	0.0	2.0	0.0	0.0	1.5	0.0
	ViT-B-MAE	0.0	0.0	0.0	0.0	0.0	1.2	0.0	0.0	0.5	0.0
	ViT-B-DINO	0.1	5.0	0.1	1.1	1.2	4.4	0.0	0.0	1.6	6.4
R-In-Tgt	Swin-T	0.2	0.8	0.1	0.1	0.0	2.8	0.0	0.0	1.5	0.0
	R50-v1	0.1	19.1	0.4	0.3	0.5	3.5	0.0	0.0	1.3	0.0
	R50-v2	0.2	1.3	3.1	0.1	0.0	2.0	0.0	0.1	1.7	0.0
	R50-DINO	0.4	2.3	0.8	5.6	1.1	3.5	0.5	0.1	1.6	0.0
	R50-FRCNN	0.4	2.6	0.7	0.7	6.9	5.0	0.4	0.2	1.5	0.1
	R50-DETR	0.0	0.3	0.0	0.0	0.0	6.2	0.0	0.0	0.0	0.0
	R50-MoCo	0.5	1.5	0.9	1.3	1.2	3.2	3.8	0.2	1.7	0.0
	ViT-B-MoCo	0.1	1.0	0.0	0.0	0.0	2.2	0.0	0.0	1.6	0.0
	ViT-B-MAE	0.0	0.0	0.0	0.0	0.0	1.1	0.0	0.0	0.6	0.0
	ViT-B-DINO	0.3	5.3	0.5	1.5	1.2	4.6	0.4	0.0	1.6	6.0

degrades the white-box performance and does not improve the APA transferability (see Table **??** both methods). To summarize, none of the attack methods produce an APA capable of significantly transferring to other models. The ResNet50 architecture (He et al., 2016) is most sensitive to an inhibition APA, while Swin-T architecture (Liu et al., 2021) is the less sensitive. When used to train ViT, the self-supervised methods MoCo (He et al., 2020) and MAE (He et al., 2022) enable a more robust model than DINO (Caron et al., 2021).

5.4 Conclusion

In this chapter, we studied a more challenging problem of designing an APA that hides people from being detected. We observe that the standard object detection evaluation procedure is influenced by APA's false positives and is highly dependent on the IoU threshold between box prediction and ground truth, raising a false sense of APA criticality. To mitigate these issues, we propose a new surrogate problem. This surrogate problem relies on multi-label classification and ensures that the APA works regardless of the IoU threshold. After observing that state-of-the-art image classification APA (Labarbarie et al., 2024) or object detection APAs (Thys et al., 2019; Huang et al., 2023) fail to disrupt multi-label classifiers, we proposed a new strategy to learn an APA against this task. This new inhibition attack succeeded in fooling some networks but did not generate an APA that transfers across models even when sharing the same architecture. These findings suggest that a universal pattern inhibiting multiple object detectors has yet to be discovered.

Chapter 6

Conclusion and perspectives

In this manuscript, we seek to address the following research question: "Can an attacker design an adversarial patch attack (APA) capable of disrupting a real-world system without possessing any prior knowledge of the targeted system?". In Section 6.1, we summarize our contributions and findings. Then, in Section 6.2, we discuss the limitations of our work. We contextualize our results within the broader scope of adversarial attack research and offer our perspective on the field's current state of the art. Finally, we propose future research directions in Section 6.3.

6.1 Summary of findings

Definition of a critical patch. Even though numerous patch attacks have been proposed in the literature, no work describes the prerequisites of a critical patch. To this extent, in Chapter 3, we have defined what may be a critical patch for a real-world system. Namely, a critical patch is a patch that is robust to physical transformations and transferable. The transferability measures the attack performance when a patch designed on model A targets a different model B. We have demonstrated that designing an adversarial patch attack may not be physically feasible without considering specific training techniques. Our findings indicate that the current methods cannot generate an APA capable of transferring to multiple models. While several approaches have been proposed in the APA literature to enhance patch physicality, there has been less focus on improving their transferability.

Patch transferability among image classifiers. In Chapter 4, we have investigated the transferability of APAs between image classifiers. We have categorized the different APA methods into two groups based on their learning principles: decision boundary-based (APA design relies on a classifier) or feature space-based methods (APA criterion defined in the feature space). We have shown that methods relying on classification decision boundaries tend to create an APA that overfits. The experimentation shows that the perturbation induced by the resulting APA pushes the attacked image representations into unseen regions far from source and target feature distributions. These unseen regions are network-specific, resulting in poor transferability. We have introduced a more robust and generalized feature space method that relies on optimal transport, which pushes the attacked feature distribution to match a target class distribution. This method mitigates patch overfitting to the source model and significantly enhances its transferability across CNNs and Transformers. When learned on a Swin model (A Transformer-based

model), our patch shows state-of-the-art transferability results through numerical, hybrid, and real-world experiments.

A patch to inhibit the detection of persons. In Chapter 5, we have studied the transferability of invisible cloak APAs across different object detectors. We have demonstrated that the standard evaluation procedure for object detection can mislead the perception of APA effectiveness. In fact, the evaluation procedure is influenced by APA's false positives and is highly dependent of the Intersection over Union threshold. To address this issue, we have developed a new evaluation method based on multi-label classification, which overcomes the limitations of previous approaches and provides a more accurate measure of APA's inhibition capacity. By re-evaluating the state-of-the-art APAs adapted to object detection using this new procedure, we found that they fail and the people are nevertheless detected correctly. We have introduced a novel inhibition attack designed for multi-label classification, which successfully deceives specific networks. However, this attack does not generate an APA capable of transferring across multiple models, suggesting that a universal pattern for inhibiting multiple detectors has yet to be found.

6.2 Discussion

Limitations of our work. To move towards the design of a critical APA, we have proposed two contributions regarding the transferability of APA. The first contribution is a novel method for creating more transferable APAs in image classification, and the second is the definition of a surrogate problem to evaluate the inhibition capacity of patches and, consequently, the transferability of invisible cloak APA. Although we have studied APA transferability, a significant limitation of our work is the lack of theoretical foundations underpinning this notion. In this manuscript, we have adopted the widely used definition of attack transferability: an attack is considered transferable if, when designed on a model A, it successfully targets a different model B. This notion remains vague and requires further clarification. Some attempts have been proposed for understanding adversarial noise transferability (Ilyas et al., 2019; Chen and Liu, 2024). For example, Ilyas et al. (2019) show that adversarial noises can be attributed to the presence of what they called *non-robust* features. These features are incomprehensible to humans, as they mainly rely on noisy patterns, yet networks may use them. They hypothesize that different networks trained on different datasets will likely learn similar non-robust features, thus explaining the transferability of an adversarial noise constructed by exploiting the non-robust features. However, this theory has been verified only for old networks trained in a supervised way. Li et al. (2024) show that non-robust features do not generalize across different learning strategies, suggesting that non-robust features are paradigm-wise shortcuts.

Although this widely used definition of a transferable attack is useful for quantifying transferability, it offers no insight into the underlying mechanisms that enable transferability. Moreover, none of the developed attacks in this dissertation or in the literature (to our knowledge) provide guarantees about the transferability of the attack, which is a significant limitation. From an attacker's perspective, if a company develops a new architecture or learning strategy and does not release it, we can not examine the efficacy of the attack on the targeted model. From the defender's perspective, system developers are unaware of which architectural or learning strategies may limit the transferability of attacks. This lack of knowledge prevents them from developing or selecting a safer system.

The difficulty of characterizing transferable attacks is linked to a more general difficulty in understanding deep networks. Let us consider the scenario of APA transferability among image classifiers, which, as we saw, is the simplest attack scenario where the classifiers can be decomposed into an encoder followed by a linear classifier. The encoder is responsible for extracting the relevant features, while the linear classifier aggregates these features to predict a label. The first step in defining APA transferability is to understand the role of the features extracted by the encoder. What do they encode? What is the meaning of the range of their values? Are they surjective and/or injective? However, answering these questions is very challenging. Interpretability questions are addressed under the XAI field (see Paragraph 1.2 for a reminder). However, the different XAI methods for computer vision are often post-hoc methods and give a weak understanding of the triptych (architecture, learning strategy, training dataset) defining neural networks. For example, performing a Grad-CAM (Selvaraju et al., 2017) may provide an "explanation" of the classifier decision, but what about the whole network understanding? Another potentially interesting toolkit is Microscope[1]. This visualization toolkit optimizes and retrieves the input, which maximizes the activation of a specific neuron of a given layer and network. However, similar to Grad-CAM, it does not provide an understanding of the overall network; instead, it only provides isolated explanations. For another interesting initiative, the reader can refer to Distill[2][3]. The explanations raised by these XAI methods become ineffective when the architecture, the learning strategy (learning paradigm or training recipe), or the training dataset changes.

The decision-making process is another aspect of neural networks that needs better understanding. We need to find out how these encoded features are used and aggregated to produce a decision. For example, the linear head of image classifiers is usually dense and pays attention to all the features extracted by the encoder. We do not know if the classification of an image relies on some hierarchy, e.g., first characterizing it as a dog or a cat, then which dog or which cat, and so on.

This lack of understanding of neural networks is not exclusive to this particular study, but is a common challenge faced by the entire computer vision community.

A shared weak understanding. The field of deep learning has experienced explosive growth in recent years, with an overwhelming number of new papers being published at an unprecedented rate. Since 2019-2020, it is not possible to keep tracking new papers on deep learning, even regarding the sub-field of adversarial attacks (see Fig. 6.1 from Nicholas Carlini blog[4]). While the expansion of research in this field can be exciting, it does not always conduct to new insights. We are seeing a growing number of papers in top-tier conferences that often offer incremental improvements without providing deeper insights or theoretical validation. In some cases, new problems are introduced, solved with relative success, but without sufficient consideration of their broader relevance or impact. Additionally, the current enthusiasm surrounding deep learning can sometimes overshadow valuable research in related areas, which may be overlooked or undervalued.

Whereas adversarial noise transferability has been explored in several works, APA transferability has remained less studied. Most works on APAs propose new methods to improve or evaluate their physicality. However, many of these works may need to

[1] https://microscope.openai.com/
[2] https://distill.pub/2018/building-blocks/
[3] https://distill.pub/2017/feature-visualization/
[4] https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html

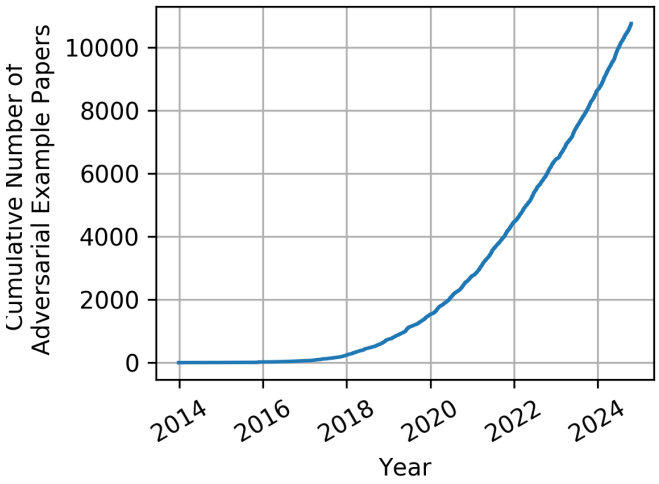


Figure 6.1: Evolution of the number of papers on adversarial noise through the years. Source: Nicholas Carlini blog.

be revisited and refined if the inhibition transferability of APAs between detectors is not proven to be feasible. Establishing this transferability is crucial, as it could significantly influence the relevance and applicability of existing research. By integrating insights gained from APA transferability studies, researchers can enhance their methodologies and ensure their findings remain valuable in the evolving landscape of APAs.

This trend of focusing on hype-driven topics shows no signs of slowing down with the rise of LLMs. Most researchers have switched to studying LLMs, including those previously working on adversarial attacks against computer vision, and now focus on attacks against such networks. However, LLMs are different but much more complex systems than models like ResNet (He et al., 2016), which remains far from fully understood despite being a simpler architecture. The fact that we have yet to comprehend why specific adversarial attacks succeed against ResNet underscores the premature shift to LLMs, where the underlying mechanisms are even less transparent. This focus on fashionable subjects risks sidelining essential work on "simpler", foundational models that could provide critical insights for the field.

The following section provides some research actions that could provide valuable insights into attack transferability.

6.3 Perspectives

As discussed, a transferable attack refers to an attack that, when designed on a source model F_s, successfully attacks a different model F_t, a model depending on the triptych: architecture, learning strategy, and training dataset.

Towards a better understanding of transferability. A first action to better understand the transferability may be to characterize the set of transferable attacks for a fixed triptych, i.e., networks sharing the same architecture, learning strategy, and training dataset, but initialized differently. To examine the existence of a transferable attack across these networks, we can optimize under a white-box scenario an attack that tries to fool simultaneously all the models. If such an attack exists, then a transferable attack may be possi-

ble. To conduct such a study, we require multiple models. A solution would be to use the checkpoints provided by Laurent et al. (2023) paper, which gives approximately 10000 ResNet18 trained on CIFAR-100 (Krizhevsky et al., 2009) and 2000 ResNet18 trained on TinyImageNet (Wu et al., 2017). Once we are convinced of the existence of such attacks, we can study the attack perturbation mechanism, i.e., which features does the attack rely on? To do so, we need to characterize, as discussed, what the features actually encode.

For each network sharing the same triptych but trained from different initializations, we get a different set of weights, but the network performance is nearly the same. Do these networks learn common features? To answer this question, (Guth et al., 2023) work tries to determine what is stable across different training runs. They show that when the width of networks goes to infinity, network activations are equal up to rotations. They propose an alignment procedure to align, at each layer, the feature activations of two networks. So, two networks are computing using the same activations up to an approximation error. A perspective may be to create an attack that only exploits the common activations in the two networks, thus enhancing its transferability to networks sharing the same triptych.

Multi-label classification as a defense. Another perspective may be to use multi-label classification as an empirical defense against inhibiting APAs. The multi-label classifier would be used alongside the object detector to, for example, raise warnings if the decisions between the two networks differ. Two different encoders must be used for the object detector and the multi-label classifier to strengthen the system's robustness.

A path between inhibition and class-switch APA. In Chapter 5, we show that designing an inhibition APA is challenging even in the white-box attack scenario. A perspective may be to create an APA that changes the predicted class of the object. This type of APA is more accessible as it only targets the bounding box classification and not its objectness. A first attempt is the Shapira et al. (2022) work. However, their attack is only performed to semantically close classes (car to bus) and does not show high transferability.

Bibliography

Julia Angwin, Julia Larson, Surya Mattu, and Lauren Kirchner. Machine bias: There's software used across the country to predict future criminals. and it's biased against blacks. *www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing*, 2016.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

Anish Athalye et al. Synthesizing robust adversarial examples. In *ICML*. PMLR, 2018.

Yutong Bai, Jieru Mei, Alan L Yuille, and Cihang Xie. Are transformers more robust than cnns? *Advances in Neural Information Processing Systems*, 34:26831–26843, 2021.

Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9*, pages 404–417. Springer, 2006.

Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.

Battista et al. Biggio. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer, 2013.

Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and radon wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51: 22–45, 2015.

Tom B Brown et al. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33:9912–9924, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

Stephen Casper, Max Nadeau, Dylan Hadfield-Menell, and Gabriel Kreiman. Robust feature-level adversaries are interpretability tools. *Advances in Neural Information Processing Systems*, 35:33093–33106, 2022.

Shang Tse Chen et al. Shape shifter: Robust physical adversarial attack on faster r-cnn object detector: Recognizing outstanding. *D. Research*, 2019.

Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9640–9649, 2021.

Yanbo Chen and Weiwei Liu. A theory of transfer-based black-box attacks: explanation and implications. *Advances in Neural Information Processing Systems*, 36, 2024.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, pages 2206–2216. PMLR, 2020.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.

Jia Deng et al. Imagenet: A large-scale hierarchical image database. In *2009 IEEE CVPR*, 2009.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Bao Gia Doan, Minhui Xue, Shiqing Ma, Ehsan Abbasnejad, and Damith C Ranasinghe. Tnt attacks! universal naturalistic adversarial patches against deep neural network systems. *IEEE Transactions on Information Forensics and Security*, 17:3816–3830, 2022.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *Journal of Machine Learning Research*, 20(55):1–21, 2019.

Mark Everingham et al. The pascal visual object classes challenge 2007 (voc2007) results. 2008.

Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Dawn Song, Tadayoshi Kohno, Amir Rahmati, Atul Prakash, and Florian Tramer. Note on attacking object detectors with adversarial stickers. *arXiv preprint arXiv:1712.08062*, 2017.

Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1625–1634, 2018.

Charlie Frogner, Chiyuan Zhang, Hossein Mobahi, Mauricio Araya, and Tomaso A Poggio. Learning with a wasserstein loss. *Advances in neural information processing systems*, 28, 2015.

Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36 (4):193–202, 1980.

Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, et al. A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, 56(Suppl 1):1513–1589, 2023.

Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014a.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014b.

Jindong Gu, Xiaojun Jia, Pau de Jorge, Wenqain Yu, Xinwei Liu, Avery Ma, Yuan Xun, Anjun Hu, Ashkan Khakzar, Zhijiang Li, et al. A survey on transferability of adversarial examples across deep neural networks. *arXiv preprint arXiv:2310.17626*, 2023.

Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244, 2019.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

David Gunning, Mark Stefik, Jaesik Choi, Timothy Miller, Simone Stumpf, and Guang-Zhong Yang. Xai—explainable artificial intelligence. *Science robotics*, 4(37): eaay7120, 2019.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

Florentin Guth, Brice Ménard, Gaspar Rochette, and Stéphane Mallat. A rainbow in deep network black boxes. *arXiv preprint arXiv:2305.18512*, 2023.

Ambreen Hanif, Xuyun Zhang, and Steven Wood. A survey on explainable artificial intelligence techniques and challenges. In *2021 IEEE 25th international enterprise distributed object computing workshop (EDOCW)*, pages 81–89. IEEE, 2021.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

Kaiming He et al. Deep residual learning for image recognition. In *IEEE CVPR*, 2016.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Yu-Chih-Tuan Hu, Bo-Han Kung, Daniel Stanley Tan, Jun-Cheng Chen, Kai-Lung Hua, and Wen-Huang Cheng. Naturalistic physical adversarial patch for object detectors. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7848–7857, 2021.

Zhanhao Hu, Wenda Chu, Xiaopei Zhu, Hui Zhang, Bo Zhang, and Xiaolin Hu. Physically realizable natural-looking clothing textures evade person detectors via 3d modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16975–16984, 2023.

Zhanhao Hu et al. Adversarial texture for fooling person detectors in the physical world. In *Proceedings of the IEEE/CVF CVPR*, pages 13307–13316, 2022.

Gao Huang et al. Densely connected convolutional networks. In *IEEE CVPR*, 2017.

Hao Huang, Ziyan Chen, Huanran Chen, Yongtao Wang, and Kevin Zhang. T-sea: Transfer-based self-ensemble attack on object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20514–20523, 2023.

Lifeng Huang, Chengying Gao, Yuyin Zhou, Cihang Xie, Alan L Yuille, Changqing Zou, and Ning Liu. Universal physical camouflage attacks on object detectors. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 720–729, 2020.

Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *International Conference on Machine Learning*, pages 2137–2146. PMLR, 2018.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *Advances in neural information processing systems*, 32, 2019.

Nathan Inkawhich, Kevin Liang, Binghui Wang, Matthew Inkawhich, Lawrence Carin, and Yiran Chen. Perturbing across the feature hierarchy to improve standard and strict blackbox attack transferability. *Advances in Neural Information Processing Systems*, 33:20791–20801, 2020.

Nathan Inkawhich et al. Feature space perturbations yield more transferable adversarial examples. In *Proceedings of the IEEE/CVF CVPR*, 2019.

Kevin Jarrett, Koray Kavukcuoglu, Marc'Aurelio Ranzato, and Yann LeCun. What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th international conference on computer vision*, pages 2146–2153. IEEE, 2009.

Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, Yonghye Kwon, Kalen Michael, Jiacong Fang, Zeng Yifu, Colin Wong, Diego Montes, et al. ultralytics/yolov5: v7. 0-yolov5 sota realtime instance segmentation. *Zenodo*, 2022.

Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873): 583–589, 2021.

Danny Karmon, Daniel Zoran, and Yoav Goldberg. Lavan: Localized and visible adversarial noise. In *International Conference on Machine Learning*, pages 2507–2515. PMLR, 2018.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Alex andothers Krizhevsky. Imagenet classification with deep convolutional neural networks. 2012.

Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pages 99–112. Chapman and Hall/CRC, 2018.

Pol Labarbarie, Adrien Chan-Hon-Tong, Stéphane Herbin, and Milad Leyli-Abadi. Benchmarking and deeper analysis of adversarial patch attack on object detectors. In *Workshop Artificial Intelligence Safety-AI Safety (IJCAI-ECAI conference)*, 2022.

Pol Labarbarie, Adrien Chan-Hon-Tong, Stéphane Herbin, and Milad Leyli-Abadi. Optimal transport based adversarial patch to leverage large scale attack transferability. In *The International Conference on Learning Representations (ICLR 2024)*, 2024.

Olivier Laurent, Emanuel Aldea, and Gianni Franchi. A symmetry-aware exploration of bayesian neural network posteriors. *arXiv preprint arXiv:2310.08287*, 2023.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553): 436–444, 2015.

Mark Lee and Zico Kolter. On physical adversarial patches for object detection. *arXiv preprint arXiv:1906.11897*, 2019.

Ang Li, Yifei Wang, Yiwen Guo, and Yisen Wang. Adversarial examples are not real features. *Advances in Neural Information Processing Systems*, 36, 2024.

Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 35(1):5–22, 2022.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

Aishan Liu, Xianglong Liu, Jiaxin Fan, Yuqing Ma, Anlan Zhang, Huiyuan Xie, and Dacheng Tao. Perceptual-sensitive gan for generating adversarial patches. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 1028–1035, 2019.

Jiang Liu, Alexander Levine, Chun Pong Lau, Rama Chellappa, and Soheil Feizi. Segment and complete: Defending object detectors against adversarial patch attacks with robust patch detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14973–14982, 2022a.

Pengrui Liu, Xiangrui Xu, and Wei Wang. Threats, attacks and defenses to federated learning: issues, taxonomy and perspectives. *Cybersecurity*, 5(1):4, 2022b.

Xin Liu et al. Dpatch: An adversarial patch attack on object detectors. *SafeAI 2019 (AAAI Workshop)*, 2018.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022c.

Noel Loo, Ramin Hasani, Mathias Lechner, Alexander Amini, and Daniela Rus. Understanding reconstruction attacks with the neural tangent kernel and dataset distillation. *arXiv preprint arXiv:2302.01428*, 2023.

David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004.

Yantao Lu, Yunhan Jia, Jianyu Wang, Bai Li, Weiheng Chai, Lawrence Carin, and Senem Velipasalar. Enhancing cross-task black-box transferability of adversarial examples with dispersion reduction. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 940–949, 2020.

Aleksander Madry et al. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133, 1943.

Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38, 2019.

Seyed-Mohsen Moosavi-Dezfooli et al. Universal adversarial perturbations. In *IEEE CVPR*, 2017.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

Muhammad Muzammal Naseer, Salman H Khan, Muhammad Haris Khan, Fahad Shahbaz Khan, and Fatih Porikli. Cross-domain transferability of adversarial perturbations. *Advances in Neural Information Processing Systems*, 32, 2019a.

Muzammal Naseer, Salman Khan, and Fatih Porikli. Local gradients smoothing: Defense against localized adversarial attacks. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1300–1307. IEEE, 2019b.

Muzammal Naseer, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Fatih Porikli. On generating transferable targeted perturbations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7708–7717, 2021.

Federico Nesti et al. Evaluating the robustness of semantic segmentation for autonomous driving against real-world adversarial patch attacks. In *Proceedings of the IEEE/CVF WACV*, pages 2280–2289, 2022.

Utku Ozbulak, Hyun Jung Lee, Beril Boga, Esla Timothy Anzaku, Homin Park, Arnout Van Messem, Wesley De Neve, and Joris Vankerschaver. Know your self-supervised learning: A survey on image-based generative and discriminative training. *arXiv preprint arXiv:2305.13689*, 2023.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Julien Perolat, Bart De Vylder, Daniel Hennes, Eugene Tarassov, Florian Strub, Vincent de Boer, Paul Muller, Jerome T Connor, Neil Burch, Thomas Anthony, et al. Mastering the game of stratego with model-free multiagent reinforcement learning. *Science*, 378 (6623):990–996, 2022.

Florent Perronnin, Yan Liu, Jorge Sánchez, and Hervé Poirier. Large-scale image retrieval with compressed fisher vectors. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3384–3391. IEEE, 2010.

Gabriel Peyré, Jalal Fadili, and Julien Rabin. Wasserstein active contours. In *2012 19th IEEE International Conference on Image Processing*, pages 2541–2544. IEEE, 2012.

Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie. Generative adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4422–4431, 2018.

Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot. Wasserstein barycenter and its application to texture mixing. In *Scale Space and Variational Methods in Computer Vision: Third International Conference, SSVM 2011, Ein-Gedi, Israel, May 29–June 2, 2011, Revised Selected Papers 3*, pages 435–446. Springer, 2012.

Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *IEEE CVPR*, 2017.

Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

Dillon Reis, Jordan Kupec, Jacqueline Hong, and Ahmad Daoudi. Real-time flying object detection with yolov8. *arXiv preprint arXiv:2305.09972*, 2023.

Shaoqing Ren et al. Faster r-cnn: Towards real-time object detection with region proposal networks. *NeurIPS*, 28, 2015.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

Andras Rozsa et al. Lots about attacking deep features. In *IJCB*. IEEE, 2017.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

Aniruddha Saha et al. Role of spatial context in adversarial robustness for object detection. In *Proceedings of the IEEE/CVF CVPR Workshops*, pages 784–785, 2020.

Takami Sato, Junjie Shen, Ningfei Wang, Yunhan Jia, Xue Lin, and Qi Alfred Chen. Dirty road can attack: Security of deep learning based automated lane centering under {Physical-World} attack. In *30th USENIX security symposium (USENIX Security 21)*, pages 3309–3326, 2021.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

Avishag Shapira, Ron Bitton, Dan Avraham, Alon Zolfi, Yuval Elovici, and Asaf Shabtai. Attacking object detector using a universal targeted label-switch patch. *arXiv preprint arXiv:2211.08859*, 2022.

Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 acm sigsac conference on computer and communications security*, pages 1528–1540, 2016.

Mahmood Sharif, Lujo Bauer, and Michael K Reiter. On the suitability of lp-norms for creating and preventing adversarial examples. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 1605–1613, 2018.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Dawn Song et al. Physical adversarial examples for object detectors. In *12th USENIX workshop on offensive technologies (WOOT 18)*, 2018.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *The 2011 international joint conference on neural networks*, pages 1453–1460. IEEE, 2011.

Christian Szegedy et al. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Christian Szegedy et al. Rethinking the inception architecture for computer vision. In *IEEE CVPR*, 2016.

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification. *Advances in Neural Information Processing Systems*, 33:18583–18599, 2020.

Bilel Tarchoun, Anouar Ben Khalifa, Mohamed Ali Mahjoub, Nael Abu-Ghazaleh, and Ihsen Alouani. Jedi: Entropy-based localization and removal of adversarial patches. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4087–4095, 2023.

Simen Thys et al. Fooling automated surveillance cameras: adversarial patches to attack person detection. In *IEEE/CVF CVPR workshops*, 2019.

Eric J Topol. High-performance medicine: the convergence of human and artificial intelligence. *Nature medicine*, 25(1):44–56, 2019.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.

Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction {APIs}. In *25th USENIX security symposium (USENIX Security 16)*, pages 601–618, 2016.

Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104: 154–171, 2013.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2009.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *nature*, 575(7782):350–354, 2019.

Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. Yolov9: Learning what you want to learn using programmable gradient information. *arXiv preprint arXiv:2402.13616*, 2024.

Xingxing Wei, Jie Yu, and Yao Huang. Physically adversarial infrared patches with learnable shapes and locations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12334–12342, 2023.

Eric Wong, Frank Schmidt, and Zico Kolter. Wasserstein adversarial examples via projected sinkhorn iterations. In *International conference on machine learning*, pages 6808–6817. PMLR, 2019.

Jiayu Wu, Qixiang Zhang, and Guoxi Xu. Tiny imagenet challenge. *Technical report*, 2017.

Cihang Xie et al. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2730–2739, 2019.

Kaidi Xu, Gaoyuan Zhang, Sijia Liu, Quanfu Fan, Mengshu Sun, Hongge Chen, Pin-Yu Chen, Yanzhi Wang, and Xue Lin. Adversarial t-shirt! evading person detectors in a physical world. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 665–681. Springer, 2020.

Ke Xu, Yao Xiao, Zhaoheng Zheng, Kaijie Cai, and Ram Nevatia. Patchzero: Defending against adversarial patch attacks by detecting and zeroing the patch. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4632–4641, 2023.

Yoshihiro Yamada, Masakazu Iwamura, and Koichi Kise. Shakedrop regularization. 2018.

Koichiro Yamanaka, Ryutaroh Matsumoto, Keita Takahashi, and Toshiaki Fujii. Adversarial patch attacks on monocular depth estimation networks. *IEEE Access*, 8:179094–179104, 2020.

Cheng Yu, Jiansheng Chen, Youze Xue, Yuyang Liu, Weitao Wan, Jiayu Bao, and Huimin Ma. Defending against universal adversarial patches by clipping feature norms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16434–16442, 2021.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. *Advances in neural information processing systems*, 31, 2018.

Anqi Zhao, Tong Chu, Yahao Liu, Wen Li, Jingjing Li, and Lixin Duan. Minimizing maximum model discrepancy for transferable black-box targeted attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8153–8162, 2023.

Wen Zhou et al. Transferable adversarial perturbations. In *ECCV*, 2018.

Appendix A

Appendix of Chapter 1

A.1 A short history of neural networks

The rise and success of artificial neural networks has been a long and sinuous journey. The term neural network is inspired by the biological structure of the human brain. The foundation of neural networks, the neuron, was initially designed by McCulloch and Pitts (1943) who mathematically proposed to model the neuron activation as an aggregation of neuron inputs using binary weights followed by a threshold function. The first trainable neural network, named the perceptron, was introduced by Rosenblatt (1958) in the late 1950s. Rather than using binary weights to aggregate the neuron inputs, Rosenblatt (1958) used real-valued neuron weights and estimated these weights using a sequential updating rule. This rule is a special case of an algorithm called stochastic gradient descent (Robbins and Monro, 1951), which remains today one of the major keystones to train deep networks. Despite this success, due to the lack of theoretical foundations and the inability of the perceptron to learn the XOR function, the field declined through the years.

A.2 The Confiance.ai program

In late 2021, the Confiance.ai program was launched in France. It is a joint initiative of industry and academia to develop and promote French capabilities in trustworthy AI.

To strengthen key sectors of the economy of tomorrow (energy, automotive, IT, space, etc.), the French government has launched the France 2030 program, investing 54 billion euros in French companies, universities, and research centers. As one of the three pillars of the Grand Défi "Securing, certifying and improving the reliability of systems based on artificial intelligence," the Confiance.ai program was launched to help industrial companies integrate trustworthy AI into their critical systems. This program serves as the technological pillar for building trustworthy AI. The second pillar concerns the evaluation and validation of the AI-based system, and the third pillar concerns standardization (defining norms, standards, and regulations toward AI certification).

To this end, this program gathers industrials, diverse research organizations, and academic institutions to leverage diverse expertise to address the different challenges (robustness, ethics, explainability and transparency, safety) raised to obtain a Trustworthy AI (Fig. A.1). The different challenges were divided into projects (called EC for environment de confiance or trustworthiness environment), where each challenge is decomposed

Figure A.1: An overview of the different actors involved in the Confiance.ai program

into sub-problems to try to tackle each problem independently. To get closer to the solutions' applicability and industrial usefulness, the industrial partners provided different use cases for which the independent solutions can be evaluated.

An industrial use case is an AI application developed by a partner that can't be deployed in the company because of a lack of trust. A typical example is the welding quality inspection proposed by Renault. Renault's development team has developed a model that can determine whether a weld on the chassis is successful or not. It is a simple classification problem. The training is unbalanced because the welds are generally good, but the developed system got a very good recognition rate: more than 97%. However, the quality supervisor refused to deploy the solution in the factory because the accuracy score was insufficient. Renault brought this use case to Confiance.ai to get methods and tools to develop an AI that would meet the expectations of the quality supervisor by adding a trust dimension.

Some other ECs have been dedicated to ensuring the real-world feasibility of proposed trustworthy solutions (Fig. A.2). To facilitate the integration of AI into critical systems an "Environment of Trust" (*Environnement de Confiance*) has been developed. This environment served as a modular platform, providing methods and tools that could be seamlessly integrated into existing engineering workflows. The program's focus extended beyond the mere development of technology, emphasizing the certification and accountability of AI systems in alignment with European standards and regulations on AI. To learn more about the Confiance.ai program, see the White Book[1].

To further develop the scientific solutions, academics put forth several proposals, leading to the initiation of doctoral studies and postdoctoral research. PhD candidates were associated with a research organization (typically IRT SystemX) and an academic partner. Each candidate had one advisor at IRT and one or more advisors in the laboratory.

[1] https://www.confiance.ai/wp-content/uploads/2023/09/LivreBlanc-Confiance.ai-Octobre2022-1.pdf

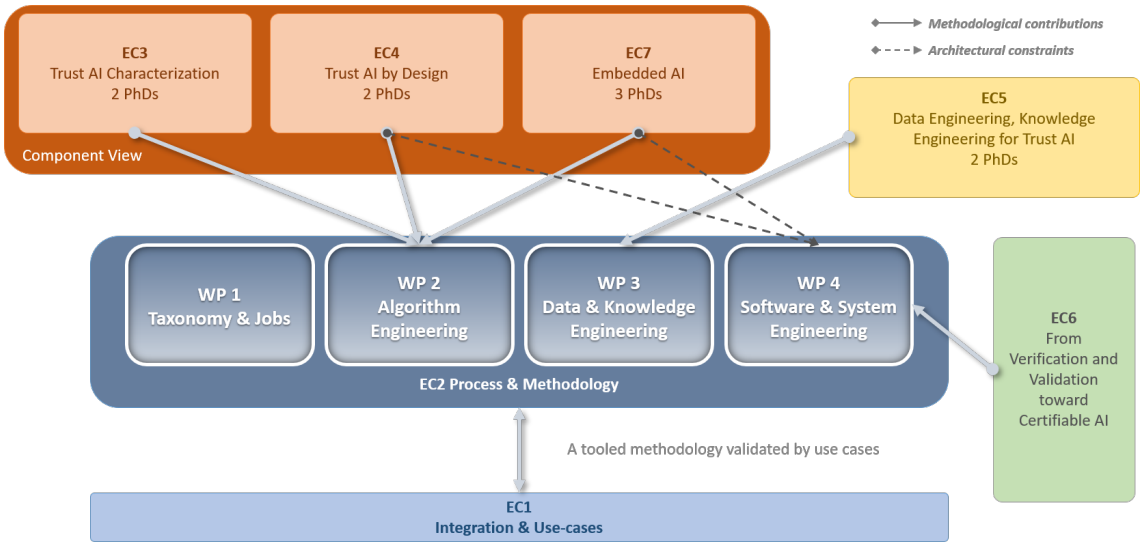


Figure A.2: Diagram of the different environment of trust (EC) of the Confiance.ai program. These ECs have been classified into two categories: those dedicated to advancing the development of trustworthy AI and those allocated to facilitating solution integrations.

Each PhD topic was associated with an EC. This thesis is associated with EC4 (related to the development of trustworthy AI by design) and was started initially to leverage the safety aspects of already deployed AI systems, especially AI components deployed in cyber-physical systems.

Appendix B

Appendix of Chapter 3

B.1 Effects of patch transformations during training

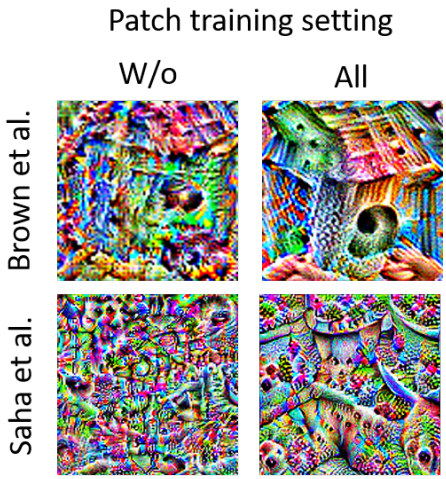


Figure B.1: Examples of the effect of the EoT method (Athalye et al., 2018) on the resulting patch style. Brown et al. (2017) patches are designed to fool ResNet 50 to output the targeted class *birdhouse*, and Saha et al. (2020) patches are contextual patches optimized to prevent YOLOv2 from detecting persons.

In this section, we qualitatively evaluate the impact of using the Expectation of Transformation (EoT) method introduced by Athalye et al. (2018) on the resulting adversarial patch attack (APA). Specifically, we compare patches generated without EoT (W/o) and with EoT applied to all transformations (All) as in Section 3.3.2.

Our observations indicate that the W/o setting tends to produce pixelated patches, which lack smooth transitions between pixels. This pixelation suggests that these patches are less likely to work effectively across different physical scenarios, as they may be sensitive to variations in real-world conditions such as lighting, perspective, and camera noise. On the other hand, the patches generated using the All setting are notably smoother, with more continuous gradients and textures. These smoother patches are not only visually more coherent but also more robust to the range of transformations that may occur in real-world deployments.

Appendix C

Appendix of Chapter 4

C.1 Implementation details

Our method training routine uses the PyTorch library (Paszke et al., 2019). For the training of each patch on medium and large models we consider a single NVIDIA V100-32G or a single NVIDIA A100 respectively. To train patches on smaller NVIDIA cards we should reduce the batch size.
When not specified, patches are designed to target one the following classes: salamander, starfish, bird house, bullfrog, pinwheel, mongoose, brown bear, accordion and common iguana.
We use Expectation over Transformations (EoT (Eykholt et al., 2018)) to obtain a more physically realizable patch, similarly to prior work on APAs (Brown et al., 2017; Lee and Kolter, 2019; Casper et al., 2022). For all the methods (GAP (Brown et al., 2017), LaVAN (Karmon et al., 2018), L2 (Inkawhich et al., 2019) and ours), during training, we randomly rotate the patch up to five degrees for the x and y-axis and up to 10 degrees for the z-axis. We also randomly scale the patch between 70×70 to 110×110 pixels, adjust patch brightness between $[-0.1, 0.1]$ and patch blur between $[0.8, 1.2]$, and apply normal noise of magnitude 0.1 on the patch. Patches are randomly translated in the image but not in the center.
To control the balance between the adversarial loss and the total variation loss, the gradient of each loss is computed individually, normalized, and combined using a weighted sum. Following Nesti et al. (2022) we choose $w_{adv} = 1$ and $w_{TV} = 0.1$ where w_{adv} is the weight for the adversarial loss and w_{TV} is the weight for the TV loss.

Computation time. We measure and report the computation time of each method in Table C.1. This Table reports the averaged computational time for the different methods. Our method has a similar computational time as other methods. This result may be counterintuitive as OT losses are known to be slow, but in our setting, the number of samples is low. The M3D method (Zhao et al., 2023) is much slower than other methods. It is coherent, this method trains alternatively the patch and two models in a min-max game.

Table C.1: Computational time of the different methods to obtain a fully optimized patch (minutes). Times are averaged over ten optimization runs. Each run is launched on the same setup composed by a single NVIDIA A100.

Method	Time
GAP (Brown et al., 2017)	20
LaVAN (Karmon et al., 2018)	30
L2 (Inkawhich et al., 2019)	20
TnT (Doan et al., 2022)	30
Casper et al. (2022)	35
TTP (Naseer et al., 2021)	30
M3D (Zhao et al., 2023)	66
Ours $(\mathbf{SW}_2^2)_{500}^{(1)}$	19
Ours $(\mathbf{W}_2^2)^{(1)}$	20

C.2 Feature point method instability

To measure the stability of the L2 method (Inkawhich et al., 2019), we launch the optimization for three randomly selected target points. Patches are designed to sway ResNet50-v1 or Swin-T to output the class Australian terrier. Figure C.1 plots the learning curves and the resulting patches for our distribution-based approach for Resnet50-v1 and Swin-T, respectively. Figure C.2 and C.3 plot the learning curves and the resulted patches of the L2 method for Resnet50-v1 and Swin-T, respectively. These four graphs show that our method is the easiest to optimize and is more robust to optimization artifacts. For the Swin-T model, the optimization for the L2 method becomes noisy. Table C.2 reports the transfer results of the obtained patches from previous figures. Although the optimization has converged for the first target of the L2 method for ResNet50-v1, the obtained patch is harmless. Even if the APA works, its attacking capacity depends on the considered target point. For example, the mean transferability on Swin-T can decreased by a factor four. In general, our distribution-oriented approach outperforms the L2 method.

Table C.2: Transfer results between categories of models (tSuc (%)) for the L2 method and for our distribution-oriented method. Three different target points are evaluated for the L2 method. Results are for the source model ResNet50-V1 and Swin-T, for the class Australian terrier and for patches of size 60×60. Patches are placed randomly in the image but not at the center of images.

Source	Method		Target CNNs-v1	CNNs-v2	ENet	CNext	DeiT	Swin	AT	mean / std
ResNet50-v1	L2 (Inkawhich et al., 2019)	Target 1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1 / 0
		Target 2	36.64	2.52	9.35	0.52	3.59	0.5	3.71	8.12 / 12
		Target 3	43.83	4.18	8.82	0.75	4.98	0.58	6.09	9.89 / 14.1
	Ours		43.34	4.76	8.75	0.92	6.46	0.63	4.68	**9.94 / 13.9**
Swin-T	L2 (Inkawhich et al., 2019)	Target 1	4.12	1.18	2.41	0.23	1.83	1.9	0.39	1.72 / 7.8
		Target 2	26.97	7.36	4.65	3.9	7.2	6.13	1.92	8.3 / 7.8
		Target 3	0.17	0.11	0.12	0.1	0.1	0.07	0.1	0.11 / 0.02
	Ours		50.77	12.54	14.2	7.08	13.64	8.19	5.94	**16.05 / 14.5**

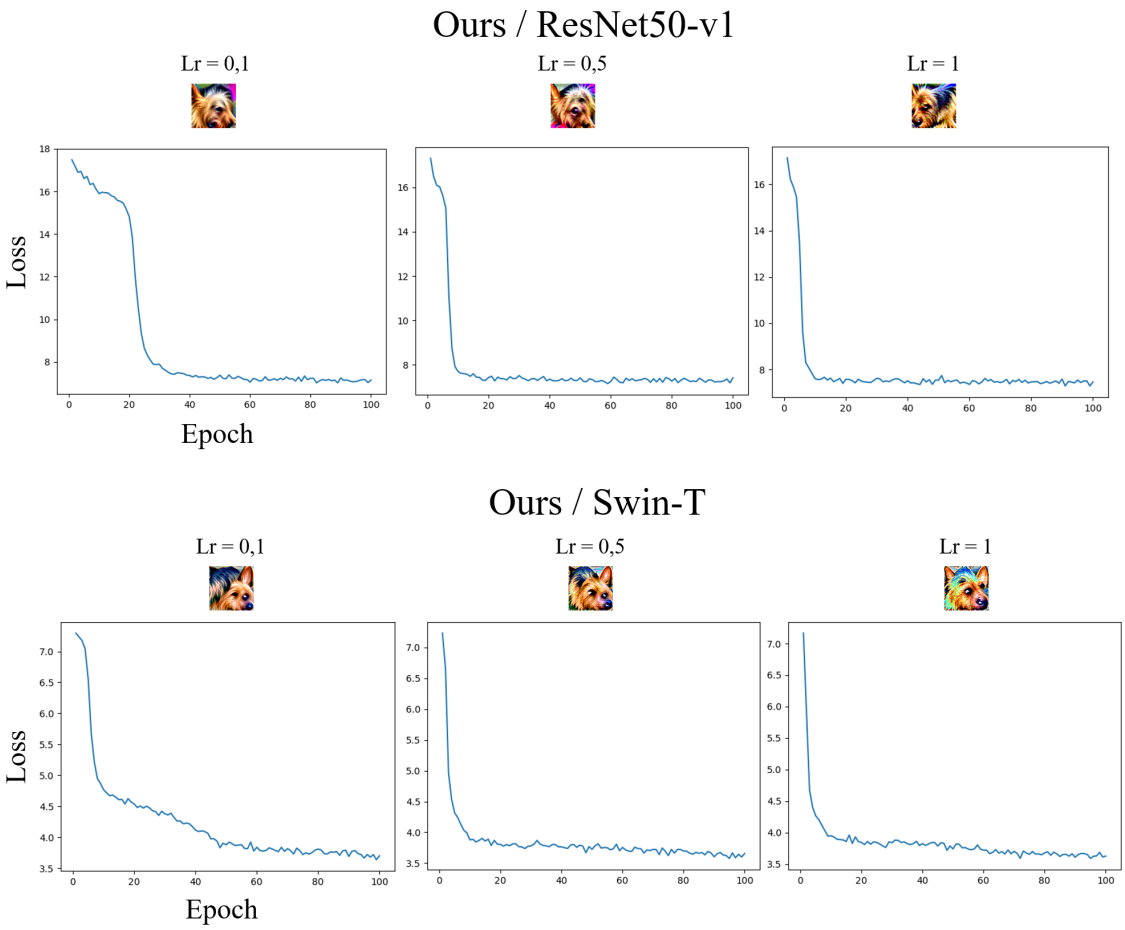


Figure C.1: Learning curves and resulted patches of our distribution-oriented method. The optimization is run for three different learning rate. The source model is ResNet50-v1 or Swin-T and the targeted class is Australian terrier.

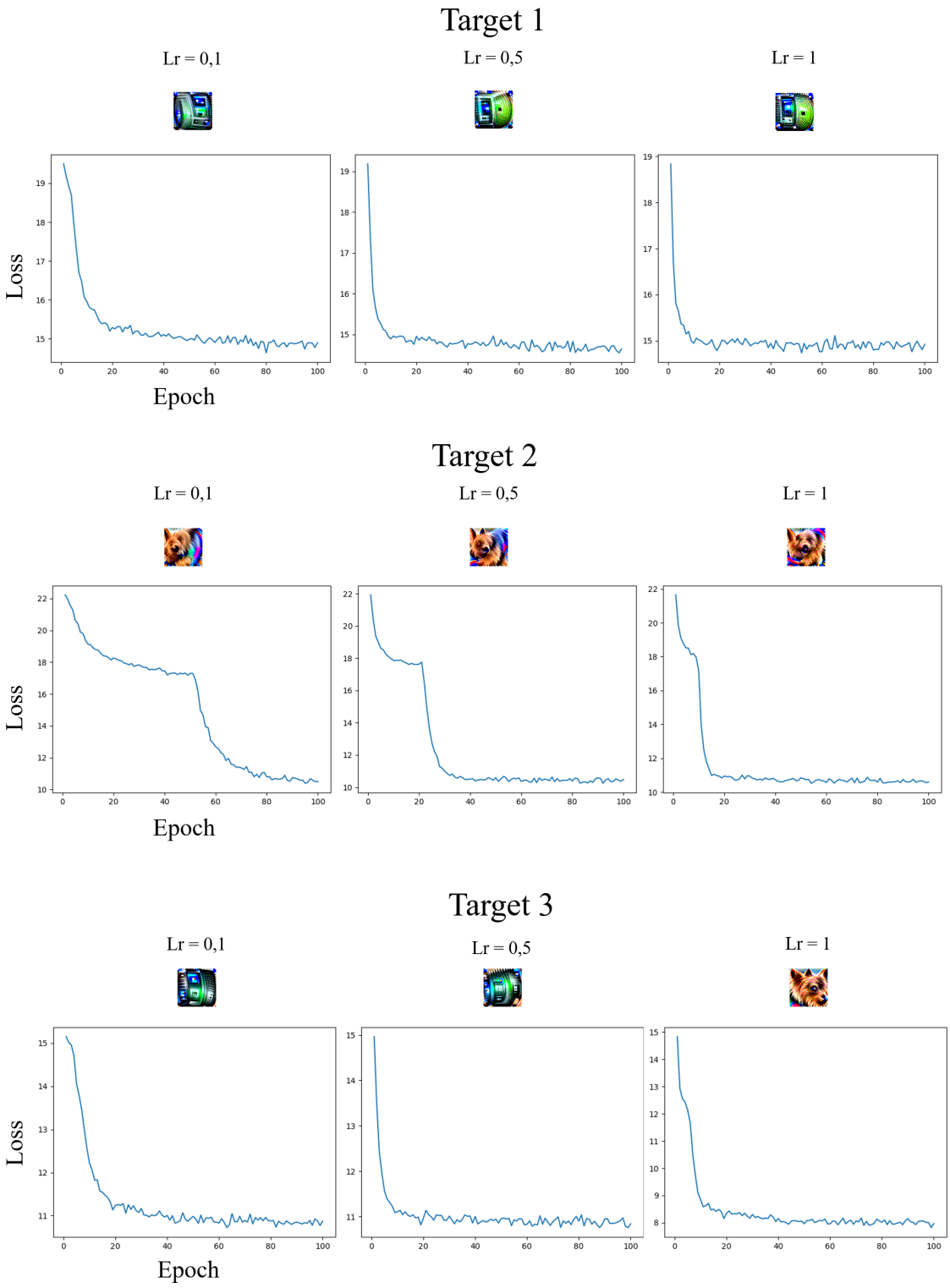


Figure C.2: Learning curves and resulted patches of the L2 method for different targeted points. For each targeted point the optimization is run for three different learning rate. The source model is ResNet50-v1 and the targeted class is Australian terrier.

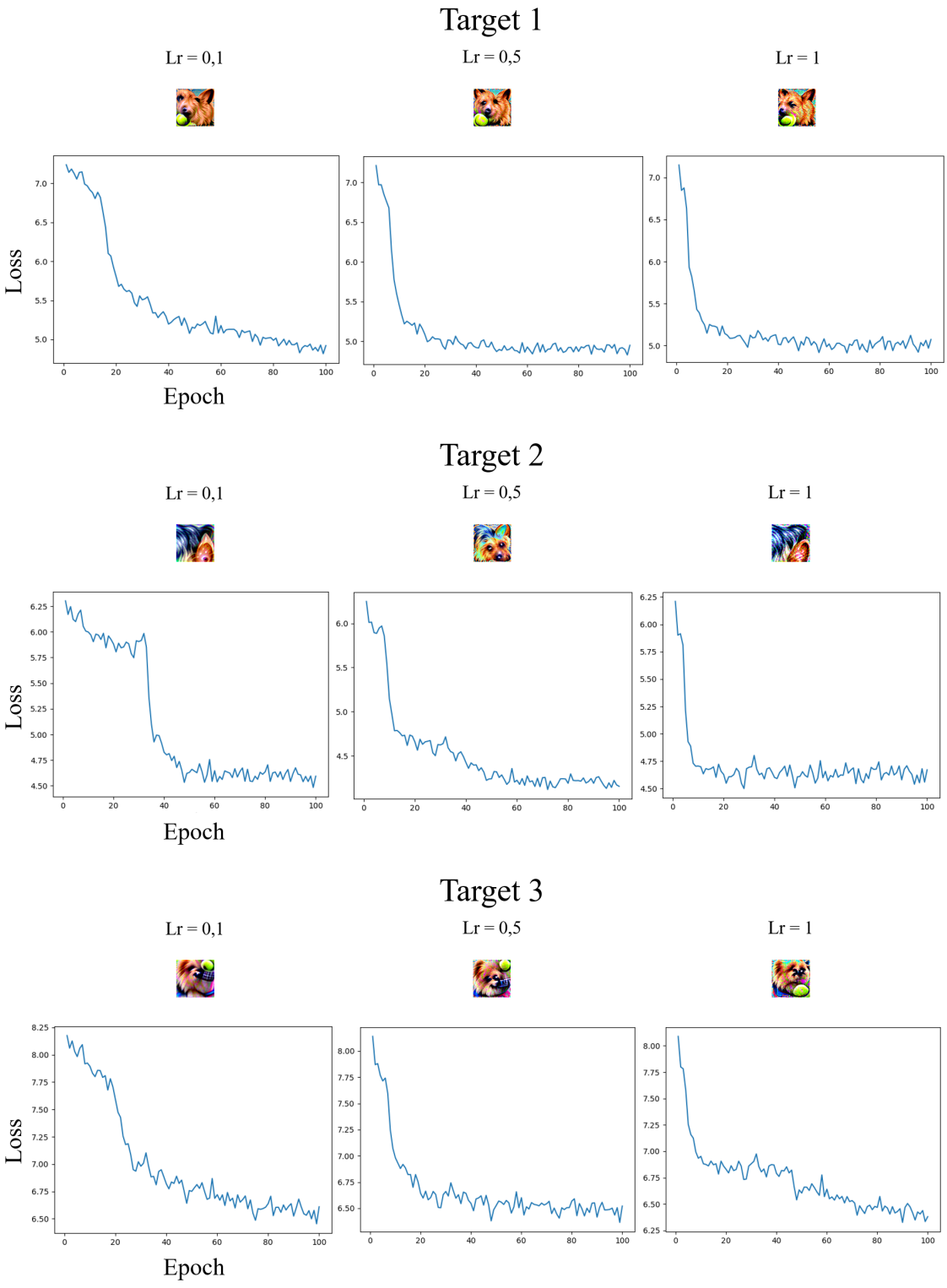


Figure C.3: Learning curves and resulted patches of the L2 method for different targeted points. For each targeted point the optimization is run for three different learning rate. The source model is Swin-T and the targeted class is Australian terrier.

C.3 Benefits of Optimal Transport

Optimal transport-based losses (both exact and sliced) has the following advantages:

- OT losses take into account the underlying metric space (through the cost matrix) on which the probability distributions are defined,
- for non-overlapping distributions such as ours, the Kullback-Leibler divergence is infinite.

To illustrate the first point, we consider the toy example shown in Figure C.4. We define four different one-dimensional distributions supported here by five points. We compute the 1-Wasserstein distance and the KL divergence between the red and the blue distributions for each column (results are shown between graphs). The blue mass has been moved near the first point from right to left. The 1-Wasserstein distance captures this mass shift, while the KL divergence does not and remains constant. This toy example highlights that OT losses capture the underlying geometry on which distributions are defined. More details concerning the advantages of OT over other methods can be found in (Arjovsky et al., 2017) (Part 2: Different Distances).

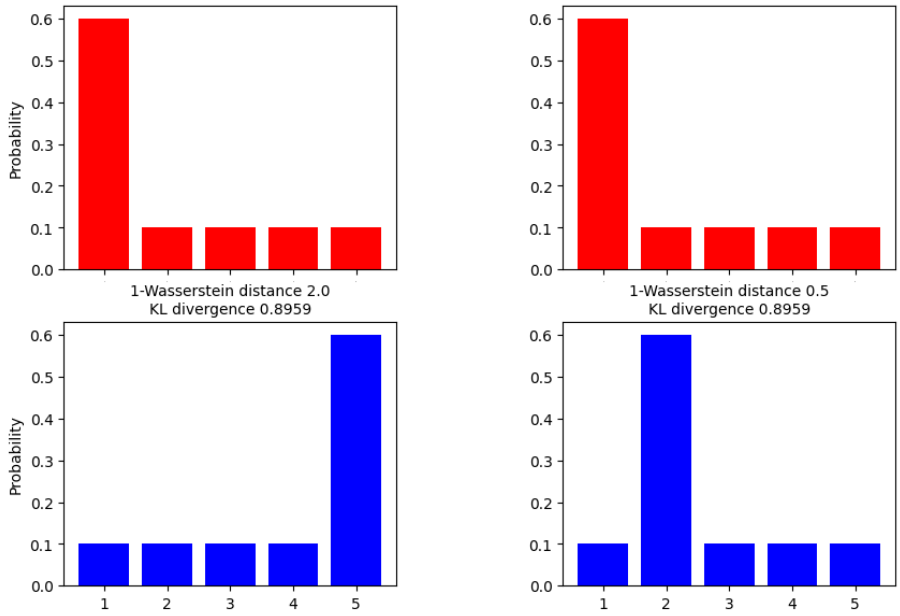


Figure C.4: Example of distributions defined on five points with different mass values. The 1-Wasserstein distance and the KL divergence is computed between the red and the blue distribution for each column.

C.4 Model robustness and patch position

In this section, we evaluate the robustness of models according to the patch position in images. We consider the same families of models as before. We define nine patch positions and measure the patch transferability when the patch is fixed at one of these positions. Figure C.5 represents the nine patch positions. We regroup these positions into three categories: Corner, Cross, and Center. We measure the patch transferability for a patch of

size 40×40 ($\approx 3\%$ image size). Results are averaged over methods (GAP (Brown et al., 2017), LaVAN (Karmon et al., 2018), L2 (Inkawhich et al., 2019) and ours), classes, and categories of patch position. Table C.3 reports the patch transferability according to its position. CNNs-v1 models are much more biased by the center of images than other network families. The accuracy of CNNs-v1 drops by a factor of 14 % when the patch is moved to corners to the center of images. This effect is not entirely due to the occluding of the object of interest since the patch is very small. Very recent families of networks (CNext and Swin models) are the more balanced networks in using context in images. For these models, the accuracy is nearly the same when the patch is placed in either corners or the center. To measure the actual efficiency of patches and to not occlude the object of interest in the case of large patches, its patches may not be placed in the center of images.

Table C.3: Transfer results according to the categories of patch position (Accuracy (%)). Results are averaged over methods, over classes, over patch positions and are for patches of size 40×40.

	Target						
	CNNs-v1	CNNs-v2	ENet	CNext	DeiT	Swin	AT
Clean / Position	74.90	77.63	80.15	82.42	77.81	82.43	65.44
Corner	71.07	76.57	78.85	81.97	76.33	81.43	64.24
Cross	67.65	75.36	77.71	81.66	75.66	81.44	62.44
Center	61.52	72.01	74.23	80.72	73.94	80.71	57.06

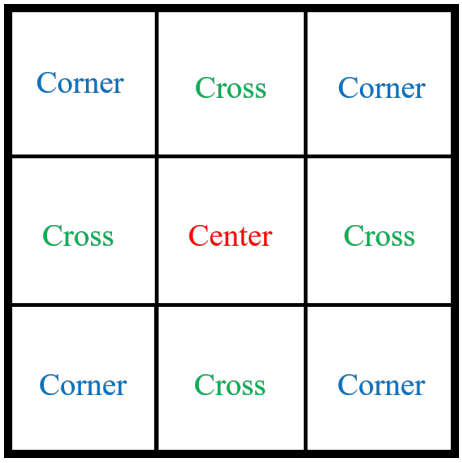


Figure C.5: Illustration of the categories of patch positions.

C.5 Transferability on adversarially trained models

In this section, we study the robustness of Adversarially Trained (AT) models. We consider two scenarios: when the patch is learned on AT models and when not. To strongly transfer on an AT model, the patch must be designed on an AT model (Table C.4). None of the other source models can show good transferability results when applied to AT models. These results suggest that AT models learn different representations than other networks. From Table C.5, we see that the GAP method (Brown et al., 2017) and our method are the best procedures to design a patch to target an AT model.

Table C.4: Transfer results between categories of models (tSuc (%)). Results are averaged over classes and over patch sizes. Patches are designed using our method $(\mathbf{W}_2^2)^{(1)}$.

	Target							
	CNNs-v1	CNNs-v2	ENet	CNext	DeiT	Swin	AT	mean / std
Source \ Clean	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1 / 0
CNNs-v1	39.65	13.01	8.27	2.44	4.89	3.16	0.82	10.32 / 12.56
CNNs-v2	19.0	11.35	3.82	4.51	3.74	4.19	0.45	6.72 / 5.86
ENet	35.12	10.45	32.0	2.27	7.8	3.79	3.49	13.56 / 12.94
CNext	3.47	12.2	0.92	25.14	2.04	15.12	0.16	8.44 / 8.69
DeiT	22.26	11.43	10.18	5.29	39.51	9.25	5.08	14.72 / 11.43
Swin	20.55	17.89	8.09	17.7	13.55	49.1	0.72	18.23 / 14.09
AT	39.75	10.69	17.35	3.51	19.87	5.31	38.95	19.35 / 13.77

Table C.5: Transfer results between categories of models (tSuc (%)). Results are averaged over classes and over patch sizes. Patches are placed randomly in the image but not at the center of images.

	Target							mean / std
	CNNs-v1	CNNs-v2	ENet	CNext	DeiT	Swin	AT	
Method \ Clean	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1 / 0
GAP (Brown et al., 2017)	43.05	11.67	16.7	3.35	20.09	5.23	39.17	19.98 / 14.51
LaVAN (Karmon et al., 2018)	37.27	10.94	14.08	3.43	18.18	5.21	29.96	17.018 / 11.64
L2 (Inkawhich et al., 2019)	6.78	1.86	2.23	0.59	4.39	1.1	8.35	3.618 / 2.77
TnT (Doan et al., 2022)	3.71	1.33	1.41	0.8	2.61	0.85	8.03	2.688 / 2.39
Casper et al. (2022)	5.83	1.38	2.74	0.54	7.55	0.97	13.91	4.78 / 4.48
TTP (Naseer et al., 2021)	35.25	9.45	13.75	2.91	17.92	4.69	35.17	17.028 / 12.43
M3D (Zhao et al., 2023)	6.24	5.61	3.45	0.82	1.82	1.12	2.53	3.088 / 1.98
Ours $(\mathbf{SW}_2^2)^{(1)}$	22.52	5.22	8.05	2.17	11.51	3.22	21.57	10.618 / 7.79
Ours $(\mathbf{W}_2^2)^{(1)}$	39.75	10.69	17.35	3.51	19.87	5.31	38.95	19.35 / 13.77

C.6 Robustness according to physical transformations

In this section, we measure the robustness of patches according to physical transformations. We evaluate the L2 (Inkawhich et al., 2019), our exact Wasserstein $(\mathbf{W}_2^2)^{(1)}$ and Sliced-Wasserstein $(\mathbf{SW}_2^2)_{500}^{(1)}$ patches as they are the only to transfer in the easiest scenario, $i.e.$, without patch rotation, medium brightness and small distance patch-camera (Section 4.4.3 of the main article). Patch transferability is measured according to z-axis rotations (rotations in the image plane), variation of light (low and high) and distance between camera and the object (the patch is placed near the object). Results are reported in Table C.6 and Figure C.6. Our patches transfer even in the worst-case scenario (far from the camera or when rotated), while other patches do not. This indicates that our patches may be critical in real-world scenarios. Globally, our method produces patches with better transferability than other methods.

C.7 Ablation studies

In this section, we study the effect of our method hyper-parameters. We solve the exact and the sliced Wasserstein distance for $p \in \{1, 2\}$ and report the results in Table C.7. This Table shows that both values of p lead to the same transferability. To penalize higher

Table C.6: Transfer results according to rotations and variation of light (tSuc %). Patches are designed to sway networks to output the class bird house. Patches are printed and placed in the real-world near a cup. Results are averaged over video frames and over all the networks.

Method	z-axis rotations			Variation of light	
	-45°	0°	45°	Low	High
L2 (Inkawhich et al., 2019)	0.8	5.7	0.23	4.4	5.7
Ours $(\mathbf{SW}_2^2)_{500}^{(1)}$	6.1	11.5	6.53	12	11.5
Ours $(\mathbf{W}_2^2)^{(1)}$	**7.1**	**14.8**	**7.05**	**12.6**	**14.8**

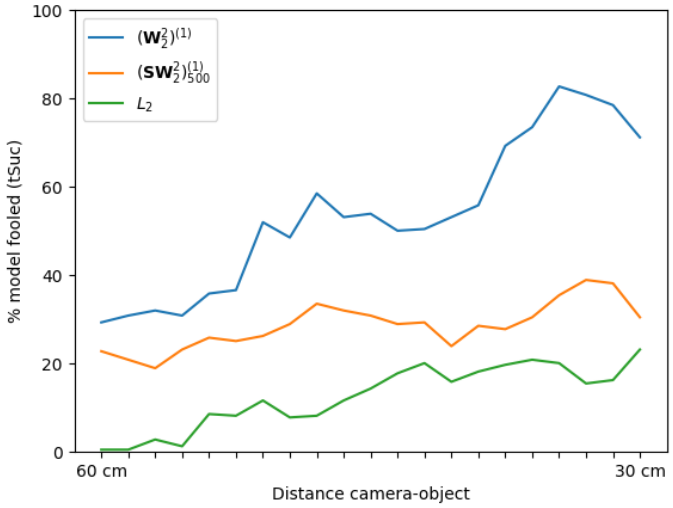


Figure C.6: Transfer results as a function of the distance camera-object. Patches are designed to sway networks to output the class bird house. Patches are printed and placed in the real-world near a cup. Results are averaged over video frames and over all the networks.

feature values, we set the value of $p = 2$.

We launch the Sliced-Wasserstein distance ($\mathbf{SW}$) for the following number of projections: $K \in \{500, 1000, 5000, 10000, 50000\}$. There is no clear advantage to considering many projections (Table C.10). The best transferability results are obtained with $K = 500$.

We now study the effect of the number of attacked layers (N). In Table C.8, we report the transferability results according to different numbers of targeted layers. We obtain better results for the exact Wasserstein distance when considering multiple layers. We observe that it helps the optimization to converge to a better local minimum, leading to stronger patches. For the Sliced-Wasserstein distance, targeting multiple layers seems counterproductive. Table C.9 details the result presented in the article on the choice of the essential layer to target. The last layer of the encoder ($l = l_J$) seems essential to model and close the gap between the two distributions and, particularly, for the Sliced-Wasserstein distance.

To evaluate the data dependency of our method, we create different targeted distributions by changing the number of points which compose it ($m = 1, 2, 10, 100, 300, 600, 900$). We launch the optimization of patches for five different sampling seeds and three different classes. We consider the Swin-T model as the source model. We evaluate patches

using the same procedure explained in the main article (Section 4.4). We report the results of the three runners-up baselines (GAP, LaVAN and TTP). As these methods do not consider distributions, they correspond to straight lines in the figure. From Figure C.8 we see that the average targeted success rate (tSuc) increases with respect to the number of target samples. When considering multiple points, our method leads to better transfer results and is more stable than the L2-based method (see **??**). Our method performs better than decision-boundary-based methods (GAP, LaVAN and TTP). However, we would like to emphasize that our method requires multiple images of the target class to overcome the limitations of the L2-based approach (see Appendix **??**). This data dependency is a practical limitation of our method. This practical limitation may be simply leveraged by considering the training data of the source model when available.

Table C.7: Transfer results according to the power p (tSuc (%)). Results are averaged over classes and over patch sizes. Patches are designed on Swin-T.

	Target						
	CNNs-v1	CNNs-v2	ENet	CNext	DeiT	Swin	mean / std
Clean \ p	0.1	0.1	0.1	0.1	0.1	0.1	0.1 / 0
$(\mathbf{W}_2^2)^{(1)}$	25.62	19.88	10.96	18.84	13.28	55.67	24.04 / 14.91
$(\mathbf{W}_1^1)^{(1)}$	26.37	20.99	10.86	19.56	13.3	56.98	24.68 / 15.31
$(\mathbf{SW}_2^2)^{(1)}_{10000}$	27.82	20.22	11.29	18.6	16.66	41.43	22.67 / 9.72
$(\mathbf{SW}_1^1)^{(1)}_{10000}$	28.74	22.72	11.24	19.89	16.07	43.13	23.63 / 10.26

Table C.8: Transfer results according to the number of targeted layers (N) (tSuc (%)). Results are averaged over classes and over patch sizes. Patches are designed on the Swin family. Layers l_{J-8} and l_{J-2} correspond to the second and third block of Swin models (which are composed by four blocks in total).

		Target						
		CNNs-v1	CNNs-v2	ENet	CNext	DeiT	Swin	mean / std
	Clean \ (N)	0.1	0.1	0.1	0.1	0.1	0.1	0.1 / 0
$(\mathbf{W}_2^2)^{(N)}$	$\{l_J\}$	20.55	17.89	8.09	17.7	13.55	49.1	21.14 / 13.12
	$\{l_{J-2}, l_J\}$	24.87	20.38	9.59	19.77	17.77	48.42	23.47 / 12.06
	$\{l_{J-8}, l_{J-2}, l_J\}$	19.14	12.45	7.52	10.56	13.55	24.75	14.66 / 14.66
$(\mathbf{SW}_2^2)^{(N)}$	$\{l_J\}$	25.26	18.7	9.19	17.27	15.32	44.11	21.64 / 11.11
	$\{l_{J-2}, l_J\}$	24.22	18.26	8.25	15.27	17.47	34.5	19.66 / 8.14
	$\{l_{J-8}, l_{J-2}, l_J\}$	15.94	10.23	6.35	8.6	12.43	17.35	11.82 / 3.89

C.8 Decision boundary-based methods overfitting

In this section, we conduct an additional experiment to support that decision boundary-based methods learn a patch that tends to overfit on the source model classifier. For this purpose, we consider the transfer not between 2 different models but between 2 models sharing the same encoder but different classifiers. We select from the different methods patches trained to attack the source model Swin-T (Liu et al., 2021). On top of this Swin-T encoder, we train a new linear classifier from scratch on the ImageNet train set (Deng

Table C.9: Transfer results according to targeted layer in the single targeted layer setting (tSuc (%)). Results are averaged over classes and over patch sizes. Patches are designed on the Swin family. Layers l_{J-8} and l_{J-2} correspond to the second and third block of Swin models (which are composed by four blocks in total).

		Target						
		CNNs-v1	CNNs-v2	ENet	CNext	DeiT	Swin	mean / std
	$\mathcal{L}$ \ Clean	0.1	0.1	0.1	0.1	0.1	0.1	0.1 / 0
$(\mathbf{W}_2^2)^{(1)}$	l_{J-2}	17.02	15.03	6.59	14.32	12.55	38.35	17.31 / 9.95
	l_J	20.55	17.89	8.09	17.7	13.55	49.1	21.14 / 13.12
$(\mathbf{SW}_2^2)^{(1)}$	l_{J-8}	0.3	0.19	0.19	0.14	0.17	0.2	0.2 / 0.05
	l_{J-2}	15.39	11.2	5.2	9.08	13.37	20.44	12.45 / 4.81
	l_J	25.26	18.7	9.19	17.27	15.32	44.11	21.64 / 11.11

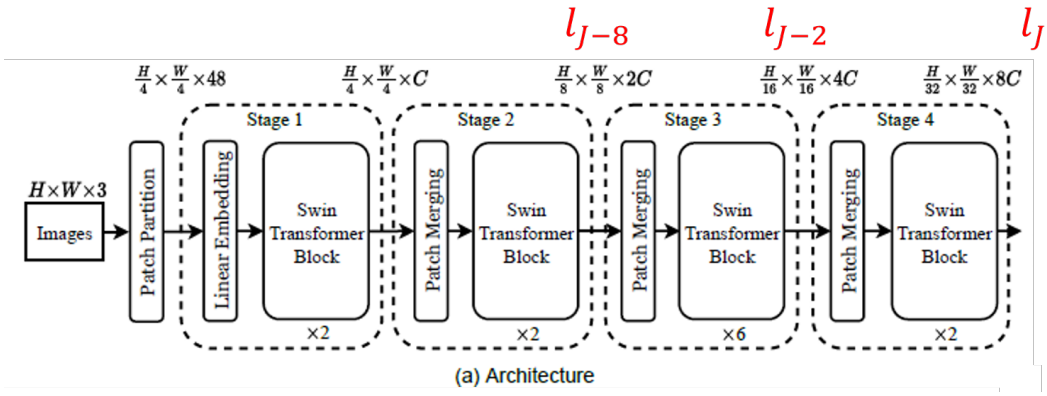


Figure C.7: Figure from (Liu et al., 2021). In red are displayed the targeted layers consider in Chapter 4.

et al., 2009). This new linear classifier reaches the same level of clean accuracy as the previous classifier (from Pytorch (Paszke et al., 2019)) while being different. We measured the patch performance when targeting this new network (same encoder, different linear classifier). As expected, the transferability of decision boundary-based patches drops drastically (nearly by half) while our patches transferability remains almost the same.

Table C.10: Transfer results (tSuc (%), higher is better attack) between categories of models. Results are averaged over classes and over patch sizes. Patches are placed randomly in the image without object overlapping. Physical transformations ($e.g.$, noise, rotations) are applied to patches. Control stands for inserting a real object of the corresponding class as a patch.

		Target						
		CNNs-v1	CNNs-v2	ENet	CNext	DeiT	Swin	mean / std
Method	Clean / Source	0.1	0.1	0.1	0.1	0.1	0.1	0.1 / 0
Control		2.85	1.59	0.86	0.54	1.57	0.93	1.39 / 0.75
$(\mathbf{SW}_2^2)^{(1)}_{500}$	CNNs-v1	25.25	6.15	4.73	1.7	5.15	2.61	7.6 / 8.04
	CNNs-v2	16.93	8.67	4.02	4.08	5.77	3.56	7.17 / 4.69
	ENet	22.53	5.83	18.8	2.07	8.49	3.03	10.13 / 7.8
	CNext	3.97	11.62	1.1	29.97	3.14	14.75	10.76 / 9.86
	DeiT	23.65	12.16	7.27	5.21	32.39	9.35	15.01 / 9.77
	Swin	25.2	20.21	8.93	19.54	16.16	45.31	22.56 / 11.3
$(\mathbf{SW}_2^2)^{(1)}_{1000}$	CNNs-v1	26.38	6.13	5.59	1.96	5.85	2.8	8.12 / 8.32
	CNNs-v2	16.88	8.82	3.97	3.89	5.8	3.53	7.15 / 4.71
	ENet	23.56	6.45	19.18	2.25	8.55	3.01	10.5 / 8.07
	CNext	4.44	12.07	1.14	33.22	3.24	15.3	11.57 / 10.89
	DeiT	22.77	11.97	7.68	5.36	35.25	9.2	15.37 / 10.48
	Swin	24.2	19.01	8.94	17.73	15.89	44.53	21.72 / 11.16
$(\mathbf{SW}_2^2)^{(1)}_{5000}$	CNNs-v1	26.4	6.11	5.37	1.83	5.2	2.65	7.93 / 8.4
	CNNs-v2	14.49	8.35	3.73	3.64	5.89	3.24	6.55 / 3.96
	ENet	27.44	7.04	19.85	2.16	8.88	3.12	11.42 / 9.2
	CNext	4.52	13.79	1.18	31.54	3.18	16.4	11.77 / 10.45
	DeiT	24.14	12.89	8.37	5.02	36.29	9.17	15.98 / 10.9
	Swin	24.02	19.69	9.53	17.97	15.06	44.74	21.83 / 11.15
$(\mathbf{SW}_2^2)^{(1)}_{10000}$	CNNs-v1	25.73	6.25	5.51	1.86	5.75	2.67	7.96 / 8.11
	CNNs-v2	18.38	10.46	4.19	4.73	6.15	4.01	7.99 / 5.14
	ENet	24.49	6.6	20.26	2.14	8.64	2.98	10.85 / 8.52
	CNext	2.92	9.34	0.92	23.33	2.9	12.18	8.6 / 7.68
	DeiT	23.87	12.22	7.57	4.89	36.3	9.34	15.7 / 11.01
	Swin	23.68	18.08	8.92	17.95	15.42	44.61	21.44 / 11.24
$(\mathbf{SW}_2^2)^{(1)}_{50000}$	CNNs-v1	26.16	6.16	5.4	1.89	5.32	2.7	7.94 / 8.29
	CNNs-v2	13.67	8.71	3.09	4.0	4.67	3.4	6.26 / 3.8
	ENet	25.66	6.06	20.4	2.11	8.73	2.99	10.99 / 8.91
	CNext	3.06	10.97	0.95	27.34	3.34	16.73	10.4 / 9.31
	DeiT	23.95	11.84	8.65	4.6	35.72	8.58	15.56 / 10.86
	Swin	25.26	18.7	9.19	17.27	15.32	44.11	21.64 / 11.11
$(\mathbf{W}_2^2)^{(1)}$	CNNs-v1	39.65	13.01	8.27	2.44	4.89	3.16	11.9 / 12.91
	CNNs-v2	19.0	11.35	3.82	4.51	3.74	4.19	7.77 / 5.69
	ENet	35.12	10.45	32.0	2.27	7.8	3.79	15.24 / 13.25
	CNext	3.47	12.2	0.92	25.14	2.04	15.12	9.82 / 8.64
	DeiT	22.26	11.43	10.18	5.29	39.51	9.25	16.32 / 11.59
	Swin	20.55	17.89	8.09	17.7	13.55	49.1	21.14 / 13.12

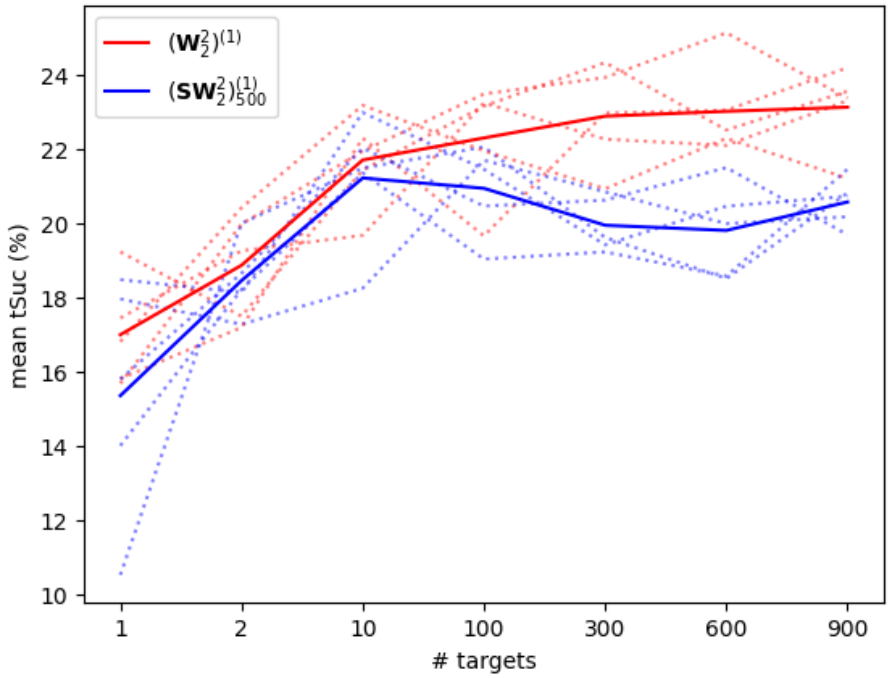


Figure C.8: Transfer results as a function of the number of targets points supported in the target distribution (mean tSuc (%)). Each dotted line correspond to a different sampling of points to create the target distribution. The solid line is the average of the five dotted lines. Patches are designed on the Swin-T source model. Results are averaged over three classes, over patch sizes and over all the targeted networks.

Table C.11: Transfer results when changing the linear classifier while the encoder remains fixed (variation of tSuc (%)). Patches are designed to fool the Swin-T model (Pytorch version, encoder and linear classifier). The transferability is measured when targeting a new network (same encoder, different linear classifier). Results are averaged over classes and over patch sizes.

Method	Variation of tSuc (%)
GAP (Brown et al., 2017)	- 61.4
LaVAN (Karmon et al., 2018)	- 42.6
TTP Naseer et al. (2021)	- 51.8
Ours $(\mathbf{SW}_2^2)^{(1)}_{500}$	**- 0.27**
Ours $(\mathbf{W}_2^2)^{(1)}$	**- 5.6**

C.9 Complementary tables

In this section, we provide additional tables. Table C.12 is the same as Table 4.7 present in the main paper but results are presented for different values of smoothing factors λ.

Table C.12: Transfer results on robustified models by LGS defense (Naseer et al., 2019b) (tSuc (%)). Patches are designed on Swin models.

		Target						
		CNNs-v1	CNNs-v2	ENet	CNext	DeiT	Swin	mean / std
$\lambda = 1.5$	Clean	0.1	0.1	0.1	0.1	0.1	0.1	0.1 / 0
	GAP (Brown et al., 2017)	0.72	0.87	0.35	0.78	1.13	2.34	1.03 / 0.63
	LaVAN (Karmon et al., 2018)	0.56	0.69	0.3	0.69	0.82	2.65	0.95 / 0.78
	L2 (Inkawhich et al., 2019)	4.79	6.44	1.72	7.79	4.79	13.85	6.56 / 3.75
	TnT (Doan et al., 2022)	0.84	0.59	0.52	0.53	0.7	0.85	0.67 / 0.13
	Casper et al. (2022)	0.37	0.4	0.2	0.32	0.25	0.59	0.36 / 0.13
	TTP (Naseer et al., 2021)	0.68	0.77	0.28	0.68	0.76	1.98	0.86 / 0.53
	M3D (Zhao et al., 2023)	0.83	0.81	0.36	0.77	1.17	1.17	0.85 / 0.27
	Ours $(\mathrm{SW}_2^2)_{500}^{(1)}$	**10.56**	**11.86**	**3.81**	**18.9**	**11.67**	**31.68**	**14.75 / 8.75**
	Ours $(\mathrm{W}_2^2)^{(1)}$	**13.23**	**13.4**	**4.37**	**21.42**	**13.84**	**32.08**	**16.39 / 8.58**
$\lambda = 1.9$	Clean	0.1	0.1	0.1	0.1	0.1	0.1	0.1 / 0
	GAP (Brown et al., 2017)	0.61	0.81	0.32	0.68	1.	1.76	0.86 / 0.45
	LaVAN (Karmon et al., 2018)	0.45	0.61	0.26	0.55	0.72	1.72	0.72 / 0.47
	L2 (Inkawhich et al., 2019)	4.05	5.72	1.53	6.6	4.27	11.26	5.57 / 2.99
	TnT (Doan et al., 2022)	0.82	0.61	0.51	0.52	0.62	0.81	0.65 / 0.12
	Casper et al. (2022)	0.32	0.33	0.19	0.25	0.23	0.5	0.3 / 0.1
	TTP (Naseer et al., 2021)	0.56	0.73	0.24	0.59	0.66	1.34	0.69 / 0.33
	M3D (Zhao et al., 2023)	0.68	0.7	0.31	0.7	1.03	1.01	0.74 / 0.24
	Ours $(\mathrm{SW}_2^2)_{500}^{(1)}$	**8.56**	**10.49**	**3.27**	**15.93**	**10.39**	**25.96**	**12.43 / 7.1**
	Ours $(\mathrm{W}_2^2)^{(1)}$	**10.95**	**11.98**	**3.78**	**18.37**	**12.35**	**27.07**	**14.08 / 7.19**
$\lambda = 2.3$	Clean	0.1	0.1	0.1	0.1	0.1	0.1	0.1 / 0
	GAP (Brown et al., 2017)	0.52	0.74	0.29	0.58	0.87	1.34	0.72 / 0.33
	LaVAN (Karmon et al., 2018)	0.38	0.55	0.24	0.47	0.64	1.19	0.58 / 0.3
	L2 (Inkawhich et al., 2019)	3.35	4.95	1.35	5.46	3.74	8.93	4.63 / 2.32
	TnT (Doan et al., 2022)	0.8	0.64	0.52	0.52	0.58	0.8	0.64 / 0.12
	Casper et al. (2022)	0.47	0.69	0.22	0.53	0.57	0.92	0.26 / 0.08
	TTP (Naseer et al., 2021)	0.47	0.69	0.22	0.53	0.57	0.92	0.57 / 0.21
	M3D (Zhao et al., 2023)	0.55	0.59	0.27	0.64	0.9	0.9	0.64 / 0.22
	Ours $(\mathrm{SW}_2^2)_{500}^{(1)}$	**6.76**	**9.16**	**2.81**	**13.1**	**9.13**	**20.69**	**10.28 / 5.59**
	Ours $(\mathrm{W}_2^2)^{(1)}$	**8.85**	**10.61**	**3.27**	**15.28**	**10.86**	**22.28**	**11.86 / 5.85**

C.10 Printable patches

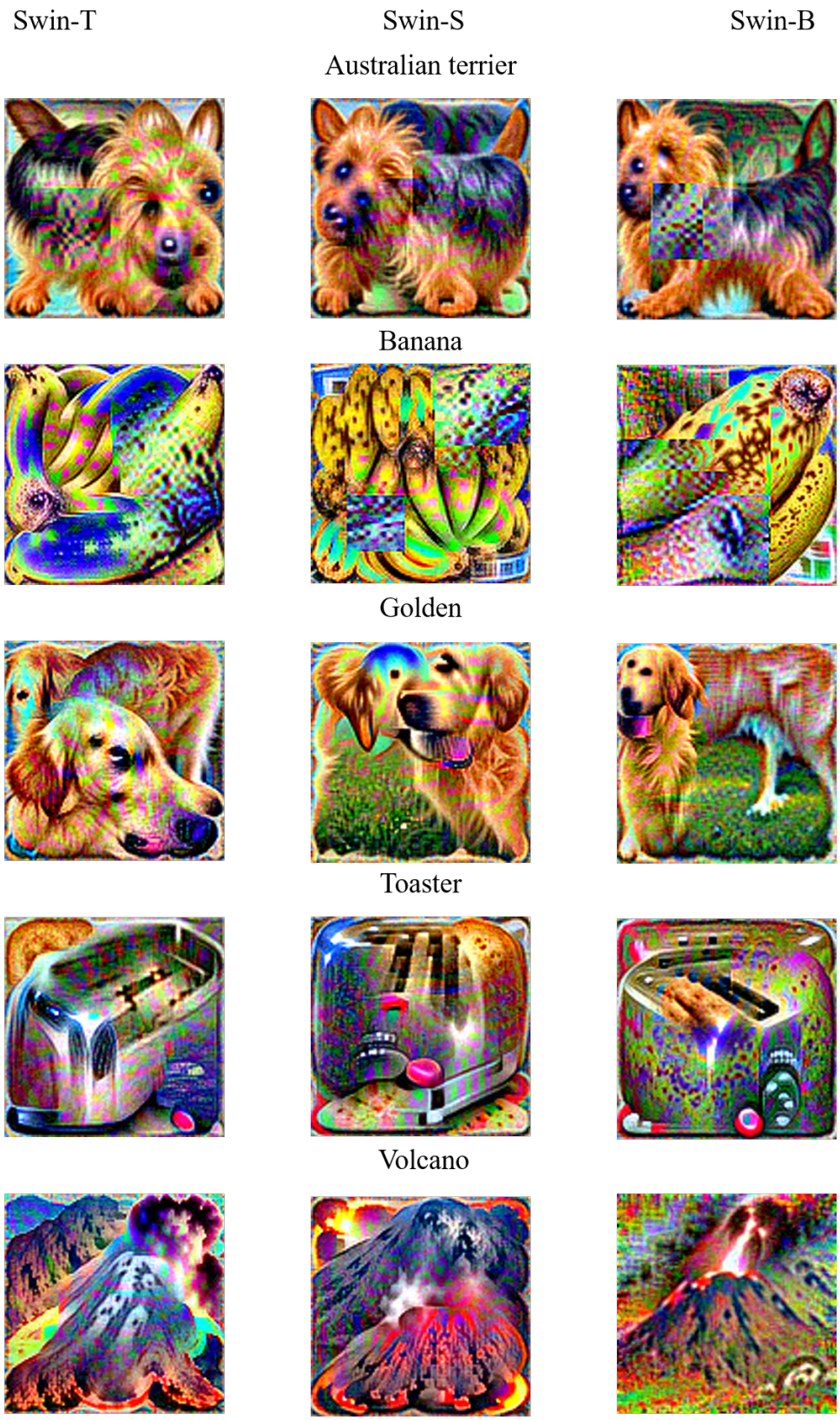


Figure C.9: Printable patches designed on Swin models with our distribution-oriented method.

Appendix D

Résumé étendu

Les réseaux de neurones profonds offrent aujourd'hui des performances inégalées notamment pour les fonctions de vision par ordinateur comme par exemple la classification d'images, la détection d'objets et la segmentation sémantique. Ces objets formels sont complexes et sujets à des instabilités. Sans précaution particulière sur l'apprentissage du réseau, il est très facile de perturber les images d'entrée pour faire en sorte que les prédictions du réseau soient erronées. On parle alors d'attaque adversaire au sens où les exemples perturbés sont optimisés pour tromper le réseau (ou la famille de réseau) cible. Une des menaces, appelée attaque par *patch*, consiste à introduire dans la scène un objet texturé pour duper le modèle (Brown et al., 2017). Par exemple, un patch placé sur un panneau stop peut amener le réseau à le classer à tort comme étant un panneau de limitation de vitesse (un autre exemple est présenté Figure D.1). Ce type d'attaque soulève d'importants problèmes de sécurité pour les systèmes de vision par ordinateur opérant dans le monde physique. Dans cette thèse, nous étudions si un tel patch peut perturber un système physique dans des conditions d'attaque réalistes, i.e., sans connaissance préalable sur le système ciblé.

Malgré le fait que de nombreuses attaques par patch ont été proposées dans la littérature, il n'est pas encore très clair si ces patchs sont résilients à des transformations géométriques ou radiométriques ou si ils peuvent être générés sans connaissance sur l'architecture ou les poids du modèle attaqué. L'une de nos contributions est la définition de ce que serait une attaque par patch critique. Pour être qualifié de critique, une attaque par patch doit vérifier deux critères essentiels (voir Tableau D.1). Tout d'abord, le patch doit être robustes à des transformations physiques, ce qui est résumé par la notion de physicalité du patch. Ensuite, le patch doit être transférable, c'est-à-dire que le patch a la capacité de duper avec succès un réseau sans posséder aucune connaissance préalable sur celui-ci. La transférabilité de l'attaque est un facteur clé, car les systèmes physiques déployés par les entreprises sont souvent opaques ou inconnus. En évaluant la robustesse des attaques par patch proposées dans l'état-de-l'art, nous avons mis en avant que sans modification particulière de l'entraînement, l'impact de ces attaques est limité par leur sensibilité à ces variations. Cela limite le risque de leur utilisation à des fins malveillantes. En adaptant l'entraînement des attaques par patch, il est possible aujourd'hui de les rendre résilient à des transformations radiométriques et géométriques. Bien qu'avec la méthode "Expectation over Transformations" (Athalye et al., 2018) la résilience des patchs aux transformations radiométriques et géométriques semble acquise, aucun patch n'a montré de capacité de transférabilité. En effet, un patch appris sur un réseau est très souvent inefficace quand appliqué à un réseau non vu durant l'apprentissage même si celui-ci a la même archi-

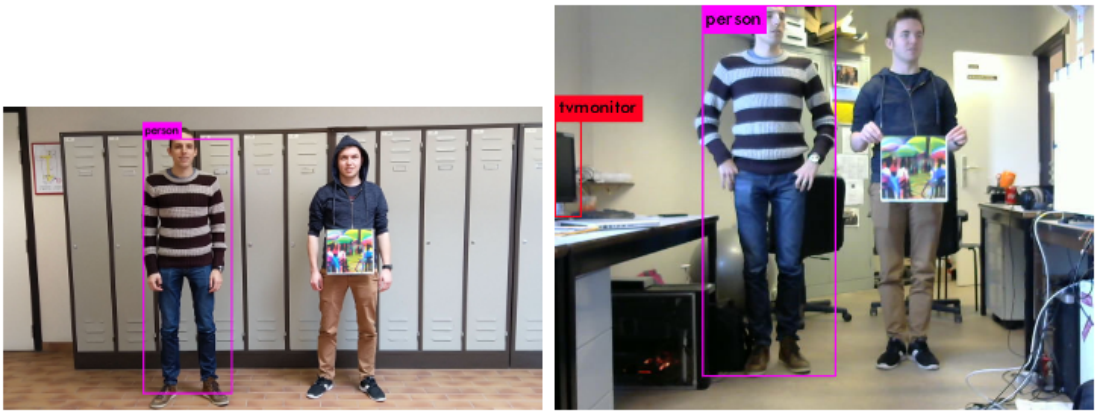


Figure D.1: Deux exemples de vidéos du monde réel perturbées par une attaque par patch. La personne portant le patch est invisible pour le détecteur d'objets, tandis que l'autre personne est détectée. Les auteurs ont fourni une démonstration de leur attaque ici. Source : Thys et al. (2019)

Propriété	Catégorie	Transformation/Processus	Exemple
Physicalité	Radiométrique	Conditions météorologiques variables	Luminosité, neige, pluie, ...
		Filtres	Transformation JPEG
	Géométrique	Redimensionnement	* * *
		Recadrage	* * *
		Transformations affines	Rotations
		Distance par rapport à la position d'apprentissage	Déplacement de la position d'apprentissage
Transférabilité	Détecteur Sensibilité	Paradigme d'apprentissage	Supervisé, contrastif, ...
		Recette d'entraînement	Augmentations de données, optimisation, ...
		Initialisation	Différentes graines
	Détecteur généralisation	Variation d'architecture	e.g. ResNet à ViT

Table D.1: Paramètres d'évaluation par catégorie et leur brève description.

tecture mais une initialisation des poids différentes que le réseau sur lequel le patch est appris.

Afin d'améliorer la capacité de transfert des attaques invisibles, de nombreux travaux ont proposé de considérer l'espace des caractéristiques pour construire leur attaque (Inkawhich et al., 2019). Par exemple, Inkawhich et al. (2019) proposent d'optimiser leur attaque invisible afin que les représentations des images corrompues se rapprochent des représentations d'une image préalablement choisie. Cependant cette stratégie présente plusieurs inconvénients. Tout d'abord, lors de l'optimisation, pousser plusieurs points vers un unique point est plus susceptible d'échouer. Même si l'optimisation converge, la performance de l'attaque dépend du choix de l'image cible choisie. De plus, l'image cible choisie peut être bien classée par un classifieur et mal classée par un autre. Afin de créer une attaque par patch transférable pour une grande variété de classifieurs d'images, nous proposons une nouvelle méthode de conception des patchs. Cette méthode repose sur l'utilisation de la distance de Wasserstein, distance définie entre deux mesures de probabilité. La figure D.2 schématise les différentes approches ainsi que celle proposée. Notre patch est appris en minimisant la distance de Wasserstein entre la distribution des caractéristiques des images corrompues par notre patch et la distribution des caractéristiques d'images d'une classe cible préalablement choisie. Une fois appris et placé dans la scène, notre patch induit plusieurs réseaux à prédire la classe de la distribution ciblée. Nous montrons qu'un tel patch est transférable et peut être implémenté dans le monde physique afin de perturber des classifieurs d'images sans aucune connaissance sur ceux-ci. Le tableau D.2 donne un

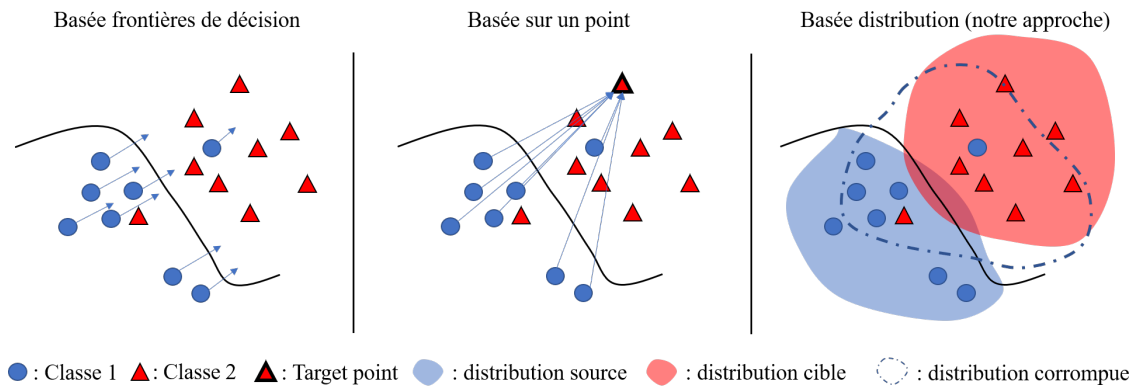


Figure D.2: Trois stratégies différentes pour concevoir des attaques par patch. Gauche : L'attaque pousse plusieurs échantillons de l'autre côté d'une frontière de décision définie pour un modèle particulier. Milieu : L'attaque fait correspondre un point donné dans l'espace des caractéristiques qui est censé représenter un échantillon d'une classe différente. Droite : Notre stratégie réduit l'écart de distribution entre les échantillons corrompus par le patch et une autre distribution trompeuse dans l'espace des caractéristiques. Elle ne dépend ni des frontières de décision, ni du choix d'un point cible spécifique dans l'espace des caractéristiques.

Method	min	moyenne	max
GAP (Brown et al., 2017)	2.22	15.46	37.33
LaVAN (Karmon et al., 2018)	2.26	8.67	31.4
L2 (Inkawhich et al., 2019)	4.44	13.6	32.78
TnT (Doan et al., 2022)	0.67	2.11	5.84
Casper et al. (2022)	0.33	3.81	14.85
TTP Naseer et al. (2021)	2.33	13.77	31.87
M3D Zhao et al. (2023)	0.84	5.19	17.11
Ours $(\mathrm{SW}_2^2)_{500}^{(1)}$	**8.93**	**22.56**	**45.31**
Ours $(\mathrm{W}_2^2)^{(1)}$	**8.09**	**21.14**	**49.1**

Table D.2: Meilleurs résultats de transfert d'un seul modèle vers tous les autres obtenus pour chaque méthode (tSuc (%) plus élevé est meilleur pour une attaque).

aperçu de nos résultats.

Afin d'avantage caractériser la potentielle menace des attaques par patch, nous proposons d'étudier leur transférabilité quand ceux-ci sont développer pour duper des détecteurs d'objets. Les détecteurs d'objets sont des modèles plus complexes que les classifieurs d'objets et sont souvent plus utilisés dans les systèmes opérant dans le monde physique. Nous étudions plus particulièrement les attaques par patch dites *cape d'invisibilité*, un type particulier de patchs conçus pour inhiber la détection d'objets lorsqu'ils leur sont appliqués dessus. Nos résultats révèlent que le protocole dévaluation utilisé dans la littérature comporte plusieurs problèmes rendant l'évaluation de ces patchs incorrecte. Pour y remédier, nous introduisons un problème de substitution qui garantit que le patch produit supprime bien la bonne détection de l'objet que nous souhaitons attaquer. En utilisant ce nouveau processus d'évaluation, nous montrons que les attaques par patch de la littérature ne parviennent pas à inhiber la détection d'objets limitant ainsi leur criticité (voir Figure D.3.

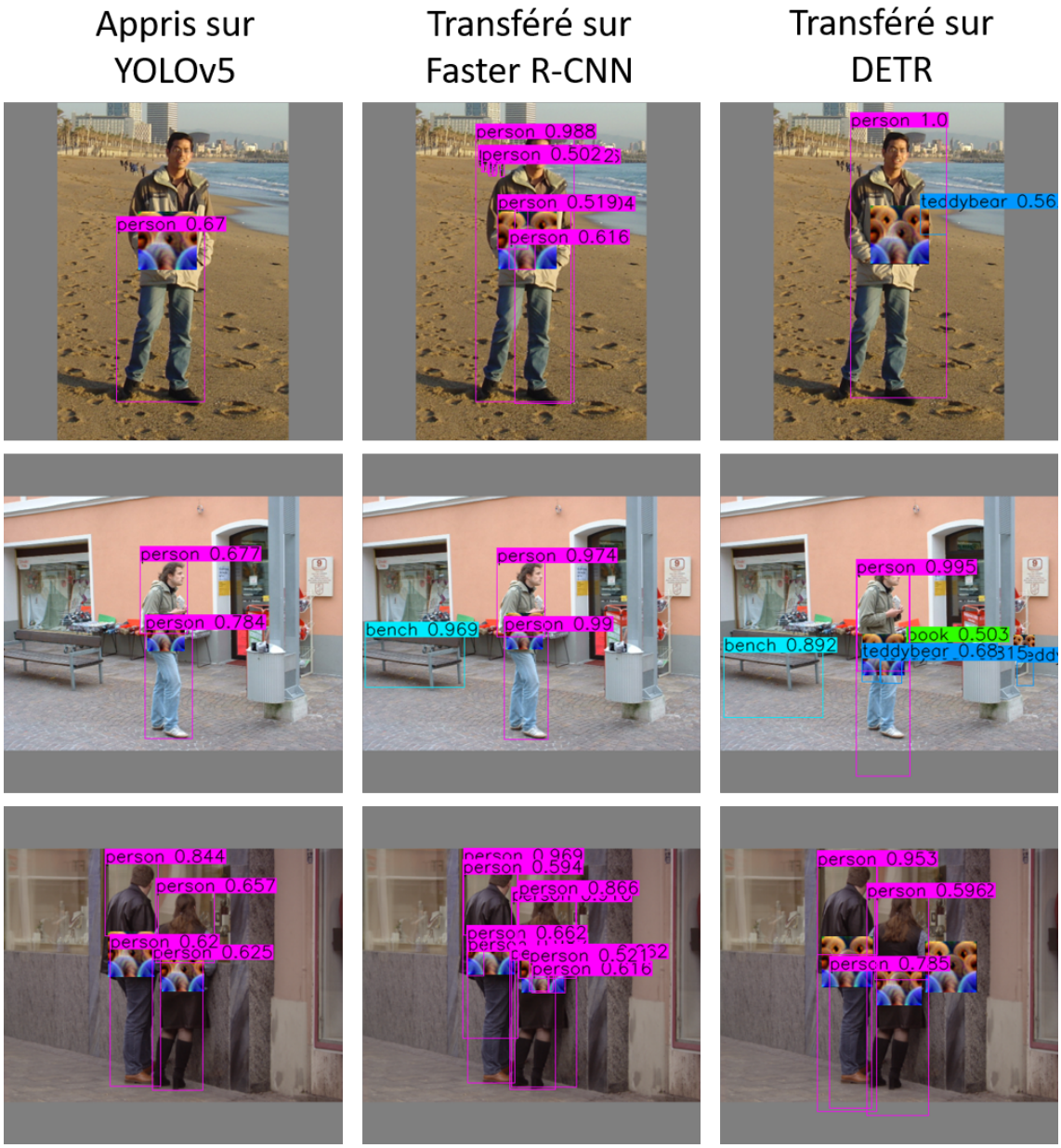


Figure D.3: Exemples d'attaque de cape invisible en white-box (conçue sur YOLOv5 et appliquée à celui-ci) et de transfert en black-box. Le patch est conçu sur YOLOv5 et provient du travail de Huang et al. (2023). Les patchs sont conçus pour empêcher la détection des personnes.