Utilisation de l'apprentissage auto-supervisé pour améliorer la détection de drone face aux nouveaux environnements

Arthur TODER[1,2] Martin PEREZ-IZAGUIRRE[1] Adrien CHAN HON TONG[2]

[1]MBDA, 1 Avenue Réaumure, 92350 Le Plessis-Robinson, France

[2]ONERA, 6 Chemin de la Vauve aux Grange, 91123 Palaiseau, France

Résumé – Cet article propose une solution au problème de la continuité des performances de détection de drones lors d'un changement de l'environnement de détection. Pour cela il présente une combinaison d'apprentissage auto-supervisé d'un extracteur de caractéristiques, SimSiam suivi d'un apprentissage supervisé du modèle de détection, YOLO, et montre ainsi l'apport de l'apprentissage auto-supervisé dans la détection de drones inconnus sur des fonds inconnus.

Abstract – This article answers to the question about the consistency of the unmanned aerial vehicles (UAV) detection, with a strong change of the environment's detection. As a solution, it presents a self-supervised learning on a one-stage object detector, through the use of Simsiam and YOLO. Thus, it shows the improvement SSL can bring to object detection on unknown object in unknown backgrounds.

1 Introduction

L'utilisation de drones est de plus en plus répandue. Les enjeux de sécurité appellent à un besoin de détection de ces objets, indépendamment de leurs tailles, de leurs formes et surtout de leurs environnements d'utilisation. Cet article cherche des solutions pour rendre un modèle de détection robuste à des environnements différents des données d'apprentissage. Par exemple, détecter des drones en zones urbaines, en disposant seulement d'images de drones en zones rurales pour entraîner la tâche de détection. L'enjeu est alors de limiter la perte de performances lors du passage des fonds ruraux à des fonds urbains. Une solution consiste à compléter les données d'apprentissage avec des images de fonds divers, sans objets et à les introduire directement dans la base d'apprentissage. Cependant, l'article montre qu'ajouter ces images aux données d'apprentissage ne fonctionne pas sans changer le processus d'apprentissage. La contribution principale de cet article est de proposer une méthode pour rendre un tel ajout de données utile grâce à une technique d'apprentissage auto-supervisé (AAS) sur les fonds d'intérêts, avant l'apprentissage de l'architecture de détection sur les cibles d'intérêt. Notre méthode modère fortement la perte de performance due au changement d'environnement dans nos expérimentations, combinant plusieurs jeux de données classiques de détection de drones et un jeu de données de fonds urbains. Sur une dimension plus technique, l'idée est d'associer un détecteur d'objet à un étage, YOLOv5 [7], à une méthode d'apprentissage auto-supervisé, Simsiam[1], pour améliorer les performances de détections de drones lors d'un transfert de domaine.

2 État de l'art

2.1 Détection de drones

La détection d'objet correspond à un problème dans lequel un réseau de neurones apprend à encadrer les objets voulus dans une image. Bien que l'état de l'art de la détection soit foisonnant, le modèle très majoritairement retrouvé dans les articles de détection de drones est le YOLOv5. Pour cette raison, les expériences sont réalisées avec ce réseau pour permettre une comparaison des résultats avec ceux de l'état de l'art de la détection de drones. Cette architecture est divisée en 2 parties, l'extracteur de caractéristiques et la tête de détection. La première partie va, comme son nom l'indique, transformer l'image d'entrée en un vecteur descripteur. Puis la tête de détection utilise ce vecteur afin de positionner une boîte autour de chaque objet d'intérêt présent dans l'image inférée.

2.2 Changement de domaine et apprentissage auto-supervisé

Bien qu'abondant, à ce jour, l'état de l'art ne montre pas d'article associant l'AAS, la détection d'objet et le changement de domaine tel qu'il est présenté dans cet article. Il existe néanmoins quelques points de comparaison. Le changement de domaine, soit le fait d'entraîner sur une distribution différente de la distribution d'inférence, est un sujet étudié notamment pour la conduite autonome. Il se met en place à travers la recherche de descripteurs invariants au changement de distribution, ou encore à travers l'incrustation du domaine cible dans le domaine source lors de l'inférence [10]. Une solution au changement de domaine consiste à classifier le domaine correspondant à l'image, notamment à l'aide d'un classifieur de domaine, [3]. Cependant, le problème auquel ce papier s'attaque n'est pas uniquement centré autour du domaine en lui même, mais plus autour de la détection d'objets dans les domaines inconnus, nouveaux et ces méthodes correspondent plus à des problémes de classification qu'a des problèmes de régression. De son côté, l'émergence de l'apprentissage auto-supervisé se fait à travers la sortie de BYOL [4], permettant d'effectuer de l'AAS sans paires négatives. Le réseau détermine ainsi, de manière automatique, des caractéristiques permettant de rassembler, ou de différencier les éléments de la base de données. Il a déjà été montré qu'une étape de pré-entraînement avec AAS permettait

une amélioration des résultats de détection [12]. L'AAS est également utilisé pour le changement de domaine, pour des fonds adverses [8] ou encore le passage du domaine visible à infrarouge. Il existe plusieurs techniques d'apprentissage auto-supervisé tel que MoCo , SimCLR ou encore SimSiam. Dans cet article, nous utiliserons notamment SimSiam dans une architecture plus générale présentée dans la partie 3.1. L'architecture SimSiam permet, à un extracteur de caractéristiques donné, d'associer deux transformations d'une même image, voir figure 1.

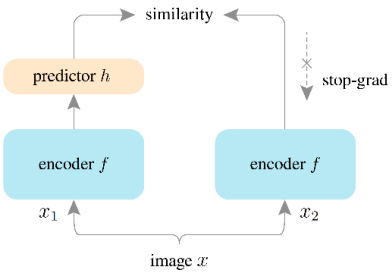

FIGURE 1 : Schéma-bloc de SimSiam

3 Méthode proposée

3.1 Formalisation du problème

Avant de présenter notre méthode, il convient de formaliser l'objectif de celle-ci, et donc comment seront évaluer les différentes approches. Notre objectif est d'évaluer une performance de transfert, dans le contexte de la détection de drones. Cependant, il n'existe aucun jeu de données répondant directement à cette problématique : les bases de données classiques de transfert sont axées sur la conduite autonome [11] ou sur un changement de style de représentation comme [5] et sont donc trop lointaines. Aussi, nous proposons le protocole suivant : les différents apprentissages et évaluations seront effectués sur les combinaisons présentées figure 4, puis chaque modèle résultant sera testé sur le jeu de données Cityscapes dans lequel des drones auront été insérés synthétiquement. Il est important de noter que les modèles n'auront jamais vu de drone incrusté, ce qui limite le risque d'apprendre à détecter les artefacts d'incrustation.

3.2 Présentation de l'architecture

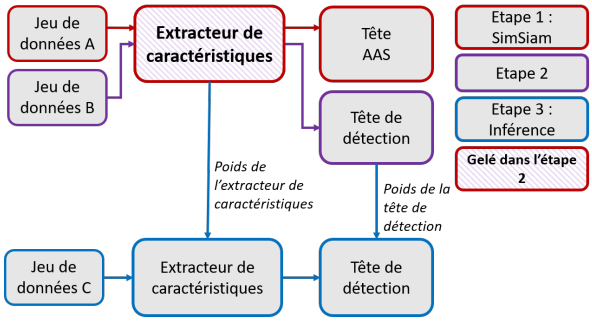

FIGURE 2 : Schéma global de l'architecture présentée

La nouvelle architecture présentée ici se découpe en 3 étapes. La première correspond à l'entraînement de l'extracteur de caractéristiques du YOLOv5 en auto-supervisé avec SimSiam sur un jeu de données (**A**) représentant l'environnement dans lequel nous voudrons détecter les drones, sans pour autant pouvoir collecter des données dans cet environnement, par exemple des images des fonds urbains, provenant du jeu de données Cityscapes. De cette étape sont sauvegardés les poids de l'extracteur qui sont gelés afin qu'ils ne soient pas modifiés lors de l'étape 2. Cette dernière correspond à un entraînement supervisé de la tête de détection du YOLOv5, à partir des poids gelés, sur un deuxième jeu de données (**B**) *source* contenant les drones. Cette étape permet d'apprendre au réseau à reconnaître des drones à partir des descripteurs de l'étape précédente. Le YOLOv5 a donc ses deux parties entraînées et peut reconnaître les drones à partir des descripteurs entraînés à caractériser majoritairement le fond des images. La troisième étape correspond à la partie inférence sur un troisième jeu de données (**C**) où des drones, similaires à ceux observés dans **B**, sont présents dans des fonds, similaires à ceux observés **A**.

4 Expérimentations

4.1 Description des scénarios

Les expériences ont utilisé différents mélanges de 4 jeux de données. Trois d'entre eux sont des bases de données utilisées pour la détection de drones, Drone versus Bird (DvB) [2], Anti-UAV [6] et Det-Fly [13]. La dernière, Cityscapes [9], est, à l'origine, conçue pour la conduite autonome et ne comporte aucun drone. Les trois bases de données avec drones sont composées de plusieurs séquences vidéo pouvant se diviser en deux catégories : urbaine et rurales. L'objectif est de déterminer la capacité à transférer les performances de l'une à l'autre catégorie et principalement de réussir à passer d'un environnement rural à un environnement urbain. Ainsi, nous séparons Drone versus Bird et Anti-UAV en 4 sous parties présentées dans la figure 3.

Det-Fly n'est jamais utilisée à l'apprentissage mais est utilisée pour fournir un drone inconnu en test.

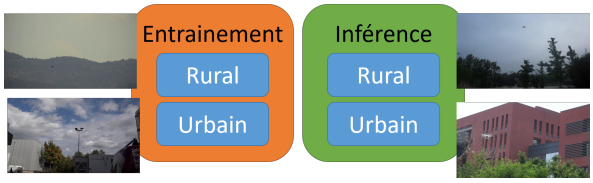

FIGURE 3 : Séparation des séquences (ex : de gauche à droite, de haut en bas, DvB rural entraînement, Anti-UAV rural test, DvB urbain entraînement, Anti-UAV urbain test)

Cette division permet dès lors d'entraîner l'AAS sur des images de drones rurales, puis d'entraîner la tête de détection sur des images de drones urbaines ou inversement. De plus, il est aussi possible d'effectuer l'AAS sur Cityscapes pour définir un espace de description à partir d'images urbaines sans drones. Nous utilisons également Cityscapes en test pour avoir des fonds dans lesquelles nous insérons des drones des trois autres bases de données. Pour chaque base de données, trois modèles de drones, chacun de taille différente, ont été détourés

puis ont été insérés, de manière aléatoire, dans les images de Cityscapes. Bien que cette technique puisse créer des artefacts sur l'image de sortie, ces images ne sont utilisées qu'à l'étape d'inférence. Les réseaux ne s'entraînent jamais sur ces images. Bien que la génération des images de test soient aléatoires, tous les modèles voient les mêmes images afin que la comparaison des performances ne soit pas bruitée. Les images correspondant à la fusion entre Cityscapes et Det-Fly ont le double avantage de correspondre à des fonds inconnus avec des drones inconnus puisque Det-Fly n'est pas présent dans les étapes précédentes. La version de référence ne possédant pas de partie AAS, elle est marquée par un rond barré sur la figure ci-dessous.

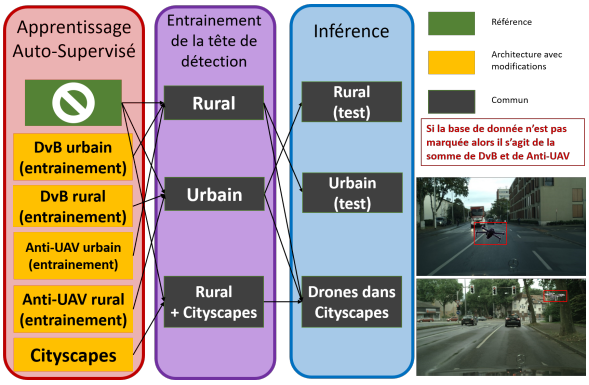

FIGURE 4 : Combinaisons des apprentissages et évaluations effectués (ex : $C_{Det-Fly}$, $C_{Anti-UAV}$)

4.2 Ajout de données simple versus ajout AAS

TABLE 1 : Performance (mAP (%), (Précision (%), Rappel (%))) d'un détecteur de drones sur deux jeux de données urbains formés par incrustation de drones dans le jeu de données urbain Cityscapes avec des drones provenant de 2 sources différentes (Anti-UAV et Det-Fly), respectivement nommés ($C_{Anti-UAV}$ et $C_{Det-Fly}$).

	$C_{Anti-UAV}$ drones connus			$C_{Det-Fly}$ drones inconnus		
Rural seulement (1)	30,0	(45,5	34,4)	22,8	(45,5	23,5)
Ajout simple (2)	10,3	(49,3	8,9)	14,5	(54,7	13,5)
Ajout AAS (nous)	**34,4**	(56,5	34,3)	**39,4**	(61,5	36,7)

Le principal résultat de cet article est reporté dans le tableau 1. Il montre les résultats d'inférence sur les 2 bases de données créées en insérant des drones provenant de 2 jeux de données différents dans Cityscapes qui n'en contient pas à l'origine. Le YOLOv5 utilisé, dans le cas de la version de référence, ainsi que dans notre architecture, a été pré-entraîné sur COCO2017.
Notre méthode, section 3.2, avec l'AAS effectué sur Cityscapes, sans drone, combiné à COCO2017, est comparée à deux références : (1) un modèle entraîné uniquement sur les séquences rurales de DvB et d'Anti-UAV, marquée par la mention « rural seulement » ; (2) un modèle entraîné sur des images rurales de drones (DvB et Anti-UAV) et des images de Cityscapes sans drone. Nous observons que l'utilisation de l'AAS permet de surpasser l'ajout simple de données, avec un gain de la mAP qui atteint $\sim 24\%$ sur Anti-UAV et $\sim 25\%$ sur Det-Fly. Ces résultats montrent qu'intégrer simplement aux données d'entraînement des images d'un environnement spécifique mais sans cibles, dégrade les performances. Cela s'explique par un plus petit taux de cibles trouvées sur les nouveaux fonds, cf la Précision dans le tableau 1. L'intuition est que, les fonds de Cityscapes sont directement caractérisés comme vides même lorsqu'un drone y est incrusté. En revanche, utiliser une étape amont d'AAS sur l'environnement spécifiquement visé permet d'améliorer les performances du résultat de référence sans images du fond spécifique comme le montre le gain de performances de $\sim 4\%$ sur les drones de Anti-UAV et de $\sim 3\%$ sur ceux de Det-Fly. L'étude de la précision, dans le tableau 2, marque l'impact de l'AAS sur la stabilité lors du changement de domaine. Les précisions, en rural uniquement et sur une combinaison Cityscapes et rural, chutent lors de l'inférence sur l'incrustation des drones de Det-Fly. Cet effet, associé à un rappel qui est en faveur de notre méthode, montre que l'AAS a un impact sur le nombre de fausses alarmes qui se retrouvent éliminées par notre architecture.

4.3 Discussion de l'influence des différents jeux de données

Les résultats du tableau 2 montrent l'apport de différents jeux de données utilisés lors de l'apprentissage auto-supervisé, sur des bases de données classiques. Comme évoqué précédemment, Drone versus Bird et Anti-UAV ont été divisées entre séquences sans bâtiments et séquences avec bâtiments. Dans le cadre de l'inférence sur milieu urbain après entraînement sur milieu rural, si l'AAS est effectué sur Cityscapes (54,7%) ou sur de DvB et Anti-UAV, avec drones et bâtiments (45,9%), la mAP est supérieure à la référence (41,9%). Ces résultats montrent que la création a priori, d'un espace de description centré sur un environnement urbain cible, permet non seulement d'atteindre les performances dans un domaine rural avec 68,1%, contre 66,8% pour la référence, mais également lors de l'inférence dans un environnement urbain. Nous remarquons cependant que lorsque le changement de domaine n'est pas important entre la base de données d'entraînement et celle d'inférence, le gel des poids de l'extracteur de caractéristiques fait diminuer les performances vis-à-vis de la référence.

La version de référence, (1) présente de bonnes performances en contexte rural (66.8%) mais généralise moins bien sur de nouveaux fonds urbains (41,9%), soit une perte de 25%. Notre processus d'entraînement, incluant l'AAS, permet d'avoir de meilleures performances sur les données urbaines et également sur les données rurales, avec des augmentations respectives de $\sim 1\%$ et $\sim 3\%$ respectivement pour (2). On observe que la présence de cibles, (3) et (4), dans l'AAS peut causer une baisse de performance. Ainsi, effectuer (2) permet de surpasser (4), lors de l'inférence sur les fonds urbains,

TABLE 2 : mAP du YOLOv5 lorsqu'il est entraîné/testé sur un jeu de données sans/avec bâtiments pour plusieurs versions d'AAS (sans AAS (1) , AAS sur Cityscapes+COCO (2) , AAS drones ruraux (3) , AAS drones urbains (4))

	Rural - test	Urbain - test
Rural - entraînement	66,8%	41,9%
	71,0%	**42,6%**
	38,3%	27,9%
	66,2%	36,5%

42,6% contre 36,5% respectivement. Cette supériorité est particulièrement notable qu'elle est présente alors que les fonds qui servent à l'inférence sont issus de la même distribution que ceux de (4), qui diffère de Cityscapes.

5 Conclusion

5.1 Résultat principal

Nous montrons dans cet article que l'utilisation d'apprentissage auto-supervisé sur un environnement cible, permet de limiter la dégradation des performances, lors du changement de domaine en détection d'objets. Ce résultat est particulièrement important en détection de drones, où les bases de données sont souvent acquises en contexte rural pour une application visée en milieu urbain. Nous montrons également que ce résultat ne peut être simplement atteint en ajoutant simplement les fonds cibles dans la base de données d'entraînement, notamment lorsque ces derniers sont vides. Ainsi, la définition d'un espace de description au préalable définit sur des images sans drones est bénéfique.

5.2 Limites

Les deux principales limites de ce travail sont d'une part que les drones sont incrustés dans l'environnement urbain, avec cependant, le garde-fou que ces artefacts d'incrustation n'auront jamais été vus à l'apprentissage, et, d'autre part, que le paradigme présenté nécessite une grande quantité d'images lors de l'étape d'AAS, ce qui peut poser problème.

5.3 Perspectives

Cet article sera étendu dans de futurs travaux à l'aide d'un générateur d'images photo-réaliste pour incruster des drones dans des fonds spécifiques sans artefacts d'incrustation. Cela permettrait aussi de tester différents scénarios en étendant les types de domaines testables, notamment autre que ceux de Cityscapes, ou en testant un continuum rural-urbain.

Par ailleurs, il serait intéressant d'évaluer, en premier lieu d'autres techniques de l'apprentissage auto-supervisé afin d'essayer d'une part de ne plus produire de perte de performance en l'absence de transfert, et d'autre part de mieux comprendre les mécanismes de modération de la perte de performance en transfert. Il serait également intéressant de venir comparer notre méthode à des techniques dîtes de "student-teacher".

Enfin, l'autre intérêt de l'apprentissage auto-supervisé est la possibilité d'utiliser la notion de distance qu'il introduit. Il pourrait ainsi être possible de définir des images voisines dans l'espace de description pour éventuellement considérer des méthodes complémentaires d'adaptation de domaine à partir de peu d'exemples.

Remerciements

Je remercie Guillaume Quin pour son aide et ses conseils sur ces travaux. Je remercie également l'Agence Innovation Défense qui soutient et finance ces travaux de thèse.

Références

[1] Xinlei CHEN et Kaiming HE : Exploring simple siamese representation learning. *In Proceedings CVPR*, pages 15750–15758, 2021.

[2] Angelo COLUCCIA *et al.* : The drone-vs-bird detection grand challenge at icassp 2023 : A review of methods and results. *OJSP*, 5:766–779, 2024.

[3] Yaroslav GANIN et Victor LEMPITSKY : Unsupervised domain adaptation by backpropagation, 2015.

[4] Jean-Bastien GRILL *et al.* : Bootstrap your own latent-a new approach to self-supervised learning. *NeurIPS*, 33:21271–21284, 2020.

[5] Naoto INOUE *et al.* : Cross-Domain Weakly-Supervised Object Detection through Progressive Domain Adaptation, mars 2018. arXiv :1803.11365 [cs].

[6] Zhao JIE *et al.* : Vision-based anti-uav detection and tracking, 2022.

[7] Glenn JOCHER : ultralytics/yolov5 : v7.0 - YOLOv5 SOTA Realtime Instance Segmentation. `https://github.com/ultralytics/yolov5`, 2022.

[8] Seunghyeon KIM *et al.* : Self-training and adversarial background regularization for unsupervised domain adaptive one-stage object detection. *In Proceedings of ICCV*, pages 6092–6101, 2019.

[9] Cordts MARIUS *et al.* : The cityscapes dataset for semantic urban scene understanding, 2016.

[10] Giulio MATTOLIN *et al.* : Confmix : Unsupervised domain adaptation for object detection via confidence-based mixing. *In Proceedings of WACV*, pages 423–433, 2023.

[11] Christos SAKARIDIS *et al.* : Semantic Foggy Scene Understanding with Synthetic Data. *IJCV*, 126(9):973–992, septembre 2018. arXiv :1708.07819 [cs].

[12] Dang TRUNG *et al.* : A study on self-supervised object detection pretraining, 2022.

[13] Ye ZHENG *et al.* : Air-to-air visual detection of micro-uavs : An experimental evaluation of deep learning. *IEEE RA-L*, 6(2):1020–1027, 2021.